

1

2

Decentralized Cybersecure Privacy Network

3

For Cloud Communication, Computing And Global e-Commerce

4

CROSS-REFERENCE TO RELATED APPLICATIONS

5

This application claims the priority of U.S. Provisional Application No.

6

62/696,160, filed July 10, 2018.

7

This application is a continuation-in-part of U.S. Application No. 15/946,863,

8

filed April 6, 2018, which is a division of U.S. Application No. 14/803,869, filed July 20,

9

2015, now U.S. Patent No. 9,998,434, issued June 12, 2018 (sometimes referred to herein

10

as the "SDNP patent"), which claimed the priority of U.S. Provisional Application No.

11

62/107,650, filed January 26, 2015.

12

This application is also a continuation-in-part of U.S. Application No.

13

15/943,418, filed April 2, 2018, which is a continuation-in-part of the above referenced

14

U.S. Application No. 14/803,869, filed July 20, 2015, now U.S. Patent No. 9,998,434,

15

issued June 12, 2018, and which claimed the priority of U.S. Provisional Application No.

16

62/480,696, filed April 3, 2017.

17

Each of the foregoing applications is hereby incorporated herein by reference in

18

its entirety.

19

20

BACKGROUND OF THE INVENTION

21

The advent of global telecommunications, computer networking, and the Internet

22

has had a profound impact on human society, touching every aspect of daily life

23

including home, family, work, entertainment, travel, and socializing. Today's Internet

24

also plays a crucial role in a country's infrastructure controlling energy production and

25

power distribution, facilitating communication, transportation systems, shipping,

26

manufacturing (including robotics), surveillance and supporting law enforcement.

27

Electronic banking, ATMs, wire transfers, online transactions, and point-of-sale (POS)

28

transactions illustrate the pervasive role of the Internet in fintech and e-commerce.

29

Network connectivity of devices, the so-called 'Internet-of-Things' (IoT), and of

30

automobiles in vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) networks

31

expand the reach of the Internet to ever corner of the globe. More recently, the advent of

1 digital identity, distributed ledgers, blockchains, and cryptocurrency represent the rise of
2 new forms of e-commerce having no corresponding counterpart in traditional brick-and-
3 mortar based business.

4 Because the Internet was not designed with security or privacy in mind,
5 transactions and data transfers processed over the Internet and the World Wide Web
6 employ the extensive use of cryptography, ciphers and codes in an attempt to maintain
7 privacy, security, and data integrity. Other efforts to improve privacy involve
8 decentralization— a method eliminating reliance on a central control, network operator, or
9 trusted authority. These methods include decentralized applications (dApps), applications
10 operating over peer-to-peer (P2P) networks lacking any network operator, and in the use
11 of smart contracts— cryptographic and blockchain based contracts employed to digitally
12 facilitate, verify, and/or enforce the negotiation or performance of a contract without third
13 party involvement.

14 Privacy advocates purport crypto-based e-commerce represents a transformative
15 technology promising transactional integrity of credible business contracts and personal
16 agreements without requiring a legal authority or a governmental agency to participate in
17 an exchange. The potential application of crypto-contracts are diverse and may include
18 electronic notary services; electronic purchase agreements and supply chain management,
19 bank-less money transfers; blockchain recording of property deeds; and lawyer-less
20 execution of wills, trusts, and estates.

21 By removing the need for a central bank or certification authority to achieve
22 ‘trusted’ transactions and contracts, transaction speed and efficiency improves while
23 contractor expense is reduced (or eliminated altogether), driving down transactional costs
24 while improving business profitability. Specifically, decentralizing legal and financial
25 authority and processes offers the potential benefits of fostering competition, improving
26 service, lowering fees, and inviting innovation in risk adverse industries.

27 In financial services for example, cryptographic blockchain technology offers a
28 compelling option to replace monopolistic, autocratic, arcane, and even obsolete
29 federated practices of big banks with flexible decentralized alternatives. By eliminating
30 reliance on a central authority, blockchain technology can improve the integrity and
31 transparency of financial transactions involving secure payments, money transfers, e-

1 commerce, and insurance.

2 Aside from its potential benefit to the financial services sector, blockchain
3 technology is already proving useful for tech startups, facilitating both a channel for
4 marketing and a flexible means for fundraising. Decentralized blockchain based contracts
5 represent a potentially powerful disruptive market force, enabling small and medium
6 sized businesses (SMBs) to effectively compete against much larger corporations by
7 facilitating supply chain management and contracts; audit-ready records; process
8 automation and by connecting entrepreneurs to clients and capital funding.
9 Decentralization also addresses concerns that large corporations could control access of
10 big data, insuring that no one company has exclusive access to or control of market data.
11 As such, blockchains offer the potential for *democratizing* business.

12 The combined application of cryptography, decentralization, smart contracts,
13 blockchain technology and cryptocurrency, properly deployed over the Internet should
14 allegedly protect individuals and businesses against hacking, circumvent control of
15 corporate oligopolies or criminal cartels, and stop cyber-criminals from perpetrating
16 fraud and identity theft on unsuspecting consumers. The same methods have been
17 advertised as a means to prevent unauthorized surveillance from governments and to
18 thwart personal profiling from today's mega-network operators (Google, Amazon,
19 Facebook, Microsoft, etc.). In their exuberance some blockchain enthusiasts even assert
20 (since a blockchain comprises an immutable ledger) that blockchain technology should
21 not only combat fraud and protect privacy, but it should even be able to protect the
22 Internet. The implication of this premise is that the blockchain itself is secure and
23 immune to hacking. But is this accolade meritorious, or is the blockchain just replacing
24 today's vulnerabilities with newer ones?

25 Moreover, do facts support the premise that today's pervasive use of cryptography
26 has successfully curtailed or reduced online fraud, hacking, identity theft, privacy
27 invasions, banking and wire fraud, and infrastructure attacks, or is the perceived security
28 benefits of cryptographic transactions an illusion?

I. Network Vulnerabilities & Deficiencies

Amid an incessant barrage of news reports of data breaches, cyber-attacks, and surveillance reports, the Internet's lack of security and deficient privacy provisions are notorious. It is estimated in 2017 cybercrime damages exceeded \$445 billion dollars. In 2018 reported cybercrime broke the unprecedented record of \$600 billion dollars. Other cyber attacks remain unreported or even undiscovered. Cybercrime in 2019 is expected to jump another 50% sequentially. Clearly then, encryption is not thwarting cyberattacks or delivering the level of security experts claim it can or should.

Privacy attacks are even more pervasive, but difficult to quantify as they may include the theft of personal information without any record of how the stolen identity data was used. Identity theft, confounded by the cavalier treatment of client personal information by social media, merchants, credit bureaus, insurance agencies, and financial institutions, not only makes the Internet a perfect platform for cyber criminals to hone their trade but also represents a convenient environ for 'profiling' targets, i.e. collecting information in order to maximize cyber attack damage.

With a proverbial plethora of 'experts' and vendors claiming to have the secret keys to mastering security and insuring privacy, one must question why the number, frequency, and magnitude of cyber-attacks are growing, not diminishing. The answer is at least in part, network vulnerability is a multi-factor problem with varied root causes including reliance on antiquated systems, unsecured communication links (intrusion points), the willful release and promotion of personal data and private information on and by social media, naive behavior of cloud users, and in general a pervasive (if not religious) over-reliance on encryption as the sole means for securing data and transactions.

Moreover, the naive assertion that blockchain technology can protect the Internet is also a false premise. Because blockchain transactions occur over the Internet, attacks on Internet's communication protocol TCP/IP expose all blockchain transactions to attack, risking data theft and the irrevocable corruption of a blockchain's data. Blockchains cannot protect the Internet because they rely on it to execute their transactions.

1 **Network Connectivity**

2 To better understand the nature of cyberattacks and what makes a network
3 connected device vulnerable, we must first consider a connected device's architecture and
4 how it operates. The basic construction of a network-connected device comprising a
5 hardware platform hosting operating system firmware and application software is
6 illustrated in **FIG. 1**. The block diagram description applies equally to high-speed
7 servers; cell phones; notebook computers; tablets; routers for WiFi, Ethernet, satellite,
8 and DOCSIS3 (cable) systems; IoT devices such as home appliances, factory automation,
9 and security systems; and to the communication and control modules in automobiles and
10 other vehicles.

11 As shown, hardware of a generic device comprises a computing core 3 such as a
12 microcontroller or microprocessor; I/O 4 comprising input-output connections to a
13 display, keypad or touch panel, ports, peripherals, sensors, and other interfaces; and data
14 storage 2 comprising non-volatile memory such as flash and scratch-pad memory such as
15 DRAM and SRAM. The hardware components are controlled by hardware specific
16 device drivers 1, which together with operating system kernel 6 form a host operating
17 system or OS for the device. While device drivers 1 convert manufacturer-specific
18 hardware components control into a hardware-agnostic generic instruction set, operating
19 system kernel 6 provides scheduling and resource management of these hardware
20 elements to facilitate all necessary communication and application operations performed
21 by the device.

22 This functionality includes management of the OSI communication-stack
23 comprising network PHY and MAC communication block 5 used to facilitate physical
24 signal communication link 16 to nearby (local) devices in a communication network;
25 TCP/IP block 7 used to manage TCP/IP datagram communication 15 beyond local
26 routers and across the entire network or Internet; and OS application VM 8 used as a
27 virtual machine (i.e. an application domain) used to host application software. The
28 software app domain may support any number of software application types including
29 host OS native applications 11, online apps 12, local blockchain processing BCP 9, and
30 block applications BC apps 10. Application specific driver and app UI/UX 13 controls
31 the user interface (UI) between the application and the user and manages the device's

1 response to a user's actions (the user experience or UX).

2 Together, the integrated system as illustrated is able to receiver commands from a
3 user, execute tasks, and communicate across a communication network with other
4 devices compliant with transmission-control-protocol/Internet-protocol (TCP/IP)
5 communications. By adhering to seven hierarchical layers of abstraction consistent with
6 the 7-layer open systems interconnection (OSI) standard, devices of completely
7 dissimilar technologies and manufacturing are able to achieve interoperability through
8 TCP/IP. Interoperability is the key feature enabling the Internet to achieve the
9 transformative impact on human society and commerce it enjoys today. By stipulating
10 every aspect of network communications, the OSI open systems standard also represents
11 the Internet's greatest weakness– its pervasive and inescapable vulnerability to cyber-
12 attacks.

13 Because security and privacy attacks come in many forms, no unified taxonomy
14 can be employed to arrange or classify the subject matter. That said, it is convenient to
15 group attacks into several classes of attack vectors (vulnerabilities) namely (A) network
16 attacks, (B) trust attacks, (C) data breaches, and (D) blockchain attacks. These attacks can
17 be perpetrated by intercepting data in transit, corrupting transactions, or by infecting a
18 network-connected device with malware.

20 **Network Attacks**

21 Network attacks describe cyberattacks perpetrated through network connectivity
22 and communication. Network attacks represent the *unauthorized access* or surveillance of
23 communication and computer networks to gain information; redirect packet traffic;
24 interfere with (or impede) bona fide business, or to commit fraud, theft, and malfeasance.
25 Denial-of-service (DOS) attacks may be considered a type of network attack. Network
26 attacks also frequently play a role in trust attacks including wire fraud, transaction fraud,
27 CA-certificate fraud, and malware diffusion. Network sniffing, snooping, and spying may
28 participate in profiling, privacy attacks, and identity theft.

29 One aspect of network communication making cyberattacks possible is the means
30 by which packet routing occurs in TCP/IP. As shown in **FIG. 2**, in packet switched data
31 networks like the Internet data is arranged into digital data packets (data frames) for

forwarding across the network. Read from left to right, the first portion of data packet 37 contains information used to establish a local connection of the device to the network followed by a datagram containing two IP addresses and a payload. Referred to as Layer-1 header, the L1 data defines timing needed to connect a user's device to the electrical, optical, or radio link (the physical signals or PHY layer) of the network's gateway device. The layer 2 data L2 describes the data protocol and address needed to interpret the media carrying the data, the media access control (MAC) address. For example tablet 30 establishes a radio link to WiFi router 31 by exchanging MAC addresses in accordance with a particular WiFi standard (such as 802.11ac) over a bi-directional microwave radio link on a defined frequency (such as 1.8 GHz) using data modulated in a specific format, namely OFDM (orthogonal frequency division multiplexing).

Following the L1/L2 header, the source IP address defines the digital address of the sending devices (so the recipient knows who to reply to) and the destination IP address, which defines the ultimate recipient of the data. As illustrated, the IP address of tablet 30 is IP_{TB} constitutes the source IP address while the address IP_{CP} constitutes the destination IP address of cell phone 36. During data transport, data packet 37 directed from tablet 30 to router 31 is forwarded to either server 32a in cloud 41 or to server 33 depending on the instructions of routing table 40 contained within WiFi router 31. If the routing table selects server 32a, the data packet is then routed over intra-cloud connection 39 to server 32b and then server 32c, before being directed out of cloud 41 to mobile network cell tower 35 and onto cell phone 36. Conversely if routing table 40 selects server 33, then routing ensues through wire link 38 to server 34, then to mobile network cell tower 35 and onto cell phone 36.

A routing table (similar to table 40) existing within every server 32a, 32b, 32c, 33, and 34 determines the routing after WiFi router 31 makes the initial path determination. As such, tablet 30 has no authority in choosing the path data packet 37 traverses through the Internet. If router 31, for example, were to choose a path through a server compromised by malware, then the security and payload integrity of data packet 37 may be compromised and all subsequent packet routing are at risk of being hijacked or misdirected for nefarious purpose.

Internet packet routing is therefore at risk of network attacks because (i) the two

communicating parties are identified by their IP addresses, (ii) the data packet remains unaltered as it traverses the network, and (iii) the sending device has no influence on the route the data takes through the network. Moreover, because the content of the TCP/IP datagram follows the standardized format made in accordance with the 7-layer OSI model standardized in a 1984 ISO publication entitled the 'Open Systems Interconnection Reference Model,' hackers can easily analyze a packet's contents to identify vulnerabilities.

The 7-layer OSI communication stacks 50a and 50b and data packet construction are illustrated in **FIG. 3**. In operation, each layer in stacks 50a and 50b relies on processes performed by the layers below it, and performs services for the layers above it. As such, a particular layer doesn't care how lower layers execute their tasks so long that data is exchanged with the layer directly below it in accordance with its protocol. Similarly the same layer is not concerned with how upper layers utilize or create data so long that it supports them, delivering and receiving data in accordance with the protocol. In this manner, a predetermined division of labor and functional communication is realized for each layer without requiring detailed knowledge of any other layer.

Using abstraction layers in an open architecture promotes fair competition, giving SMBs unbridled commercial access to the burgeoning Internet and World Wide Web (WWW or Web) while thwarting any one company, technology, or government from dictating policy or usurping control. No registration or central authority approval is required to connect to the Internet. Simply by adhering to agreed abstraction layers in accordance with the OSI standard, a device can reliably negotiate and subsequently communicate with other network-connected devices with no knowledge of the other devices. In detail, the seven OSI layers collectively comprise a 'protocol stack' representing the physical interface, either electrical signals, electromagnetic waves, or light, along with data processing hardware and software used to interpret and use the signals. In operation, data is passed to and from a network-connected device, which in turn may utilize its own separate and unique abstraction layers dedicated to realizing applications in computing, databases, robotics, IoT, security, or as a general hardware abstraction layers (HAL). The Internet's protocol stack can also be linked to business services, or to other non-tech industries, financial transactions, banking, shipping, and

1 more.

2 As described in the table below, the 7-Layer OSI model includes two lower layers
3 for connecting a device to a network over a physical medium, two middle layers for
4 controlling packet routing over the Internet, and three top layers for managing network
5 applications.

6

Layer Name	Function / Feature
7 Application	APIs, BC, PKIs, login Telnet, file transfer FTP, trust CA-cert/L7, email IMAP, SMTP, computing DCOM, networking DNS, DHCP, NTP, TLS/SSL
6 Presentation	Cryptographic encapsulation, compression, trust CA-cert/L6, encoding, translation, images EBCDIC/ASCII, PDF, MPEG, document security
5 Session	Session initiating, authentication, trust CA-cert/L5, authorization, full/half duplex, session restoration, SOCKS, tunneling PPTP
4 Transport	Transport reliability and handshaking (TCP/UDP), port addressing, transport security selector (SSL/TLS)
3 Network	IPv4/IPv6, IP routing, IP addressing, traffic control, time to live, ICMP, PIM multicast
2 Data Link	Media Access Control (MAC) connectivity in accordance with Ethernet, WiFi, 3G/LTE, 4G, 5G, DOCSIS3 protocols, security WEP, WPA2
1 Physical (PHY)	Signal transmission as symbols (or bits) including timing control, synchronization, digital (electronic), radio, microwaves, light

7

8 During communication between two devices, data from the application layer is
9 optionally encrypted then passed down the stack, encapsulated into a IP datagram with
10 transport instructions and IP address routing, then transmitted over the PHY layer to the
11 second device using the Data Link Layer-2 specific protocol. Once delivered to the
12 packet's destination IP address and port, the packet is validated, decrypted then passed up
13 the stack to the application layer for execution. Although the 7-layer abstraction model is

generic, the Internet was adopted using the TCP/IP protocol and network stack. TCP/IP is an acronym for transmission control protocol / Internet protocol.

As shown in **FIG. 3**, even though the only physical connection occurs on the PHY Layer-1 (shown as a solid line), each communicating device pair operates virtually on a layer-by-layer basis (depicted by dashed lines), where transport Layer-4 communicates to the other device's transport Layer-4, session Layer-5 communicates to it corresponding Layer-5, and so on. As such each data layer exhibits its own security vulnerabilities, especially Layer-7 data comprising a packet's payload, the contents of which may include user ID information, passwords, login files, executable code, and blockchain data, or cryptocurrency. Examples of vulnerabilities of the TCP/IP data packet of **FIG. 3** include

- PHY data 51 Layer-1 data frame vulnerabilities such as signal intercepts and jamming
- MAC data 52 Layer-2 data frame vulnerabilities including ID sniffing, packet sniffing, profiling, denial-of-service (DoS) attacks, WPA/WPA2 (i.e. WiFi Protected Access) hacking
- Network data 53 Layer-3 datagram vulnerabilities in IP routing, DNS name server, and static addressing including man-in-the-middle (MiM) attacks, packet hijacking and rerouting, sniffing and packet recording, spoofing (imposter attack), and denial-of-service (DoS) attacks
- Transport data 54 Layer-4 datagram vulnerabilities including port banging of static and pre-assigned (fixed) port numbers, TCP protocol exploits, SSL/TLS exploits and brute force code breaking, metadata collection and user profiling, traffic monitoring, and denial-of-service (DoS) attacks
- Session data 55 Layer-5 session vulnerabilities in identity, trust and the establishment of ad hoc networks including certificate authority (CA) fraud, cryptographic key theft, malware installations, privacy attacks, man-in-the-middle (MiM) attacks and denial-of-service (DoS) attacks
- Presentation data 56 Layer-6 payload vulnerabilities involving encryption and trust including unencrypted content, weak encryption, cryptographic key theft, key analysis by profiling, man-in-the-middle attacks, triggered key theft, imposter

1 exploits, malware attacks including backdoors, keyloggers, and Trojans, privacy
2 attacks, and denial-of-service (DoS) attacks

- 3 • Application data 57 Layer-7 payload vulnerabilities of data and content including
4 malware installation, man-in-the-middle attacks, asset theft, cryptocurrency theft,
5 wallet theft, login exploits and redirects, OS coup, privacy and trust attacks,
6 blockchain attacks, blockchain corruption, database corruption, identity theft,
7 account blocking, account theft, blockchain surround and DoS attacks, application
8 denial-of-service (DoS) attacks. Layer 7 vulnerable payload data 57a includes
9 software code, blockchains, and cryptocurrency including smart contracts 57b.

10

11 *Metadata & Routing Vulnerabilities:*

12 An inescapable vulnerability of TCP/IP communication is the data packet
13 necessarily carries meaningful unencrypted data that can be observed during transport
14 because the data is needed to route the datagram. This data necessarily includes the
15 content of Layer 1 through Layer 6 carrying blocks (sub-packets) of unencrypted data 51
16 to data 56 because the Internet's packet routing relies on this information.

17 The observable data contained in intercepted IP packet includes the Layer-2 MAC
18 addresses of the sender's device (with the sender's subnet); the Layer-3 source and
19 destination IP addresses of both communicating parties (essentially the identities or the
20 persons communicating); the data transport protocol (UDP, TCP) employed; the Layer-4
21 port number of the sending and receiving devices describing the type of service being
22 requested (e.g. email, VoIP, etc.); and Layer-5 data involved in authenticating a party to
23 open a repeating dialog referred to as a 'session'.

24 Collectively, this data is referred to as 'metadata'. Governments and state actors
25 routinely monitor, collect, and store metadata to profile or analyze the behavior of a
26 person of interest. If a state can monitor network traffic then Mafioso, gangs, and
27 criminal cartels can do the same thing to profile an enemy. The real danger in packet
28 metadata is the IP source address contains information traceable to a specific device and
29 ultimately to the identity of the person using the device. Once an individual's identity is
30 known their data traffic can be monitored, their behavior patterns analyzed, their social
31 media trolled for additional useful information, and ultimately their passwords, identity,

1 and their accounts stolen.

2 If an attacker illicitly gains access to the cellular network, the location of the
3 caller's cell phone can also be triangulated through the relative signal strength of the cell
4 phone towers communicating with the device, meaning a person's device can be located
5 and their movements tracked, even with GPS turned off. This feature gives criminals,
6 gangs, kidnappers, human traffickers, and sexual predators enormous advantages over the
7 police in selecting the place and time to attack their next victim. By hacking IoT clouds,
8 criminals can concurrently use surveillance cameras to know where police are and to help
9 choose when and where to best perpetrate a crime.

10 Even without profiling a user, a hacker can inspect unencrypted Layer-6 data
11 related to the coding of the payload's encryption, using the information to either to launch
12 a brute force attack to break the encryption key, to use password guessing (like a person's
13 birthdate or common names for dogs and cats), or to devise a means to infect the target's
14 device with spyware. Although pure brute force attacks have a lower chance of success
15 than profiled attacks (where the attacker knows something personal about their target),
16 recent brute force hacker strategies employ cloud computing, harnessing the power of
17 millions of computers to guess passwords or to break a code's encryption key. Using
18 cloud computing with the collective computing power of today's fastest supercomputers
19 'unbreakable' cyphers have been broken in surprisingly short times. The advent of
20 quantum computing will only make matters worse, likely favoring hackers to expedite
21 their attacks.

22 23 *Session Layer-5 Vulnerabilities*

24 One especially intrinsic weakness in Internet communication is its necessary use
25 of a Layer-5 session to establish a dialog between two parties. Without TCP/IP's 'session'
26 feature, every packet sent over a network would have to be checked for its authenticity
27 one-by-one, a time consuming process that would prohibitively slow down Internet
28 communication, create unmanageably data traffic congestion, and prevent real time
29 communication or video. Instead, when two devices begin communicating, the establish
30 'trust' to open a channel by exchanging information either using an encryption key, or by
31 exchanging a signed digital certificate issued by a trust certificate authority (CA).

1 Once the trust is established, the two devices are connected by a virtual channel,
2 meaning any data exchanged from that point forward having the proper digital credential
3 is taken "on faith" as being authentic and accepted without question. One example of the
4 use of a session is when a client opens an online session with a bank to access their bank
5 account. During the login process, the account holder's identity is verified and the TCP/IP
6 session is established. Transactions performed while the session remains open do not
7 require further verification and no additional identity authorization is required.

8 In essence, a session is a device-to-device communication channel that is easily
9 identifiable by the IP addresses of the two communicating parties and by the session
10 security credentials exchanged at the beginning of the session. Should a hacker find a
11 means to insert their device into the session when it first commences, they will have
12 access to all information exchanged between the two parties. Session payloads may
13 include bank account numbers, passwords, private assets, contacts, photographs, videos,
14 chat histories, company confidential data, credit histories, medical data, tax information,
15 social security payments, insurance files, and more. Such attacks, referred to as man-in-
16 the-middle (MiM) attacks or packet hijacking are especially nefarious because the
17 unsuspecting parties are wholly unaware that their communiqués are being hacked. In
18 some MiM attacks, a fake CA-certificate is used to authenticate the attacker's identity
19 and to open a valid session using a *fraudulent session* token thereby bypassing the Layer-
20 5 security provisions altogether. Such a method was employed to disseminate the
21 notorious virus Stuxnet, malware that ultimately infected the entire worlds' computers,
22 even those of its perpetrators.

23 Another certificate theft method employs a Trojan horse to manipulate calls
24 between the main application's executable (ex: the browser) and its security mechanisms
25 or libraries. In a SSH downgrade exploit, the attacker tricks the client and server to use a
26 less secure protocol before continuing with their attack. The malicious session can then
27 be used to gather information or to commit fraudulent transactions. Opening a fake
28 session may also be employed to execute a *Session layer malware attack*, delivering
29 system malware including zero day exploits, time bombs, viruses, or worms.

30
31 *Presentation Layer-6 Attacks:*

1 Layer-6 attacks generally involve stolen security credentials and encryption keys
2 often using the same mechanisms of fake CA-certificate in order to fool Layer-5
3 authentication. Because virtually every Internet data packet relies on encryption to ensure
4 security and privacy, an attack on Layer-6 and Layer-7 delivered cryptographic keys
5 renders most communications exposed to spying and criminality. Aside from defeating
6 security and disabling all privacy protections, the exploit may also involve Presentation
7 Layer-6 malware attacks, installing malicious code in the form of innocuous looking
8 utilities including PDF readers, media players, ad-blockers, disk defrag utilities, etc.

9 One particular strategy for crypto key theft involves apprehending the distribution of
10 one or more crypto keys from a third party crypto key server when a connection is first
11 made. For example some allegedly ‘secure’ personal messengers distribute keys openly
12 over the Internet. If the keys are intercepted, the security of “end-to-end” (E2E)
13 encryption is compromised. For example, despite claiming unbreakable E2E encryption,
14 all VoIP personal messengers Line, KakaoTalk, WhatsApp, WeChat and even Telegram
15 have been reportedly been hacked and in some cases the contents of private
16 communications unmasked.

17 18 *Application Layer-7 Attacks:*

19 Application layer attacks employ a diverse range of stratagems involving fake
20 identities (CA-certificate fraud and trust attacks), malicious code (malware and spyware),
21 and a variety of denial-of-service attacks. Most Layer-7 attacks start with deception– using
22 a digital signature, fake SSH keys or fake CA-certificate to gain access and system
23 privileges. Once a cyber attacker uses fraudulent security credentials to pass
24 authentication and gain access to the system or cryptographic keys, the only protection
25 remaining for Layer-7 applications are the security provisions built-in to these apps.
26 Many apps however, offer limited or no security provisions, instead relying wholly on the
27 TCP/IP protocol stack to protect their content and integrity.

28 Layer-7 malware attacks can be used to subvert, cripple, or destroy a system with
29 viruses or worms; to gather information using spyware, phishing, key loggers, and
30 Trojans; to bypass security by installing backdoors; to overtly take control of a system
31 such as ransomware; or to surreptitiously gain control of files and processes. Other

1 attacks involve zero day exploits (such as Stuxnet) or to use fileless malware infections.
2 Storage drive and database attacks can also be used to steal personal information, steal
3 credit card and banking data, steal login files, or perform theft of cryptocurrency from
4 accounts or crypto-wallets.

5 Downloading of personal photographs and private documents may also be used to
6 perform extortion or blackmail. The attack can also involve installing content or software
7 of unknown content or purpose, generally activated through some active process or
8 application. In some cases, cyber criminals may utilize special software called ‘crypters’
9 to protect their malware from antivirus utilities. Similar cyber attack methodologies are
10 adaptable to cell phones including back doors, ransomware, botnets, and spyware. Attack
11 vectors include downloads from malicious websites; encrypted malicious payload
12 downloads; and stealth malware designed to circumvent detection including anti-security,
13 anti-sandbox, and anti-analyst techniques.

14 Another means by which cyber criminals are able launch an effective application
15 layer attack is through 'root access', to gain access to system administration rights of a
16 device, server, or network. Root access can be gained through covert means such as
17 Trojans or by injecting malicious adware to infect a large population, to steal
18 information, and to earn money as credit for fraudulently installing apps. So rather than
19 gaining unauthorized access to one user’s account, by hacking the system administrator’s
20 login, significant access and privileges become available to the cyber pirate without the
21 knowledge of those using the system. Since the system administrator acts as a system’s
22 police, there is no one to catch their criminal activity – in essence for systems or
23 networks with corrupted administration there is no one able to police the police.

24 Such attacks on personal computers, servers, and on mobile phones are referred to
25 as pirate administration or infiltration attacks. The task for cyber criminals is made easier
26 by the practice of jail breaking or ‘rooting’, where a user modifies the operating system
27 of a mobile phone to give themselves administrative privileges. The phone, once rooted,
28 loses its defensive abilities against malware. In extreme cases, the attacker can usurp
29 complete control of the device. This scenario is especially worrisome in IoT and V2X
30 transportation applications, where an effective cyberattack could take control of an
31 autonomous vehicle, intentionally or inadvertently causing life threatening conditions or

1 accidents.

2 Denial-of-Service (DoS) attacks, while possible to execute on any layer, are most
3 commonly executed on Application Layer-7 because of the myriad of diverse
4 applications on which the attack can be executed including HTTP, FTP, IMAP, Telnet,
5 SMPT/POP, IRC, XMPP, SSH, etc. Especially popular vectors include HTTP attacks on
6 web server processes and web application attacks on CPU processes.

8 **II. Identity Fraud & Trust Attacks**

9 Trust attacks can be considered *an attack of imposters*, where perpetrators (or
10 their devices) pretend to be someone they are not, usurping the identity, authority, and
11 access privileges of their target to engage in illicit transactions or to install malware into
12 devices disguised as valid applications or utilities. Oftentimes trust attacks are performed
13 immediately following network and communication attacks in order to capitalize on
14 stolen information before anyone notices. Spying and personal profiling are also often
15 used to gather information as a prelude to imposter exploits including the use of network
16 attacks and packet sniffing, or through physical device interventions using malware
17 including spyware, key loggers, login exploits, etc. The monetization of identity theft also
18 represents another type of trust attack, using fake credentials to divert funds (wire fraud)
19 or fraudulently pay for purchases (transactional fraud).

20 Trust attacks are frequently executed as man-in-the middle (MiM) attacks where
21 neither party is aware of the intrusion. For example, in **FIG. 4**, an imposter 41c inserts a
22 node 44 into the communication network between communicating parties, namely
23 browser 41b and secure HTTPS server 41a and attempts to subvert the certificate
24 authentication procedure by introducing false credentials into the exchange. The session
25 opens normally with browser 41b making an introduction (client hello 42) to HTTPS
26 server 41a. When the server 41b replies by sending a valid SSL certificate 43a as a
27 response imposter 41c intercepts the message *en route* by pirate node 44 which instead
28 forwards a stolen or falsified SSL CA certificate 43b to client browser 41b. The CA
29 certificate describes how information is to be encoded by client browser when
30 communicating with the HTTPS server.

31 The client then replies with a cryptographic key 45 used to codify a cypher unique

1 to the particular session. The changed cipher 46a granting access to HTTPS server 41a
2 communication is then intercepted by pirate node 44 establishing an authorized secure
3 communication link 47a between pirate node 44 and HTTPS server 41a. Concurrently
4 pirate node 44 sends a corrupted cipher 46b to client browser 41b establishing an
5 authorized secure communication link 47a between pirate node 44 and browser 41b.
6 Once completing the ruse, client browser 41b communicates with HTTPS server 41a
7 through a secure but corrupted channel comprising links 47a and 47b without knowing
8 pirate node 44 has inserted itself into all communication in the channel. In essence,
9 TCP/IP's session protocol helps enable a pirate to safely and securely commit a crime.

10 Trusting an untrustworthy certificate authority or accepting a corrupted or
11 fraudulent CA certificate is a common means by which the integrity and security of
12 Internet communication is compromised and the means by which network connected
13 devices trust and accept files containing malware and spyware. The administration of CA
14 certificates in Internet based online transaction processing is illustrated in **FIG. 5**. In the
15 example shown client 80 intends to securely execute a commercial transaction with
16 online merchant 89. In order to validate purchase order PO 88 for merchant 89, client 80
17 first sends private information 82 to a registration authority RA 83 who confirms the
18 client's identity 84 and inform certificate authority 85. Certificate authority 85 then issues
19 a copy of the CA certificate 86a to client 80 and CA certificates 87 (including a digitally
20 signed copy of CA certificates 86a) to verification authority VA 92. To execute a
21 transaction client 81 presents PO 88 to merchant 89 along with CA certificate 86a.
22 Merchant 89 then verifies the authenticity of CA certificate 86a in a dialog 91a and 91b
23 with verification authority VA 92 to complete the transaction.

24 Should the certificate authority CA 85 or the validation authority VA 92 be
25 corrupted or a stolen CA certificate employed, however, a fraudulent process may be
26 wrongfully approved thereby wrongfully endorsing illicit financial transactions and
27 enabling malware infections. The client and merchant must therefore 'trust' the certificate
28 and validation authorities are earnest and honest. Worse yet, many CA certificates are
29 derived from previously generated certificates to create a family tree of 'root',
30 intermediate, and 'leaf' certificates whereby many leaf certificate originate from a
31 common ancestral root CA certificate. This ancestral heritage forms a 'trust chain' where

1 a leaf certificate accepted in a transaction is assumed valid because its heritage traces
2 back to a trusted root certificate issued by a reputable certificate authority. But how can
3 the validity of the leaf certificate be truly trusted if the intermediate certificates are issued
4 by unknown parties?

5 Once a fake or corrupted CA certificate is used to open a session or validate a
6 transaction, transactional and data fraud can occur virtually undetected and untraceable.
7 Fraudulent processes improperly verified by corrupt CA certificates may include money
8 wire fraud, illicit online and point-of-sale (PoS) transactions, account theft, login
9 exploits, login usurpations, and malware distribution. It can also be used to allow an
10 unauthorized device, drive or USB key to access a server or personal computer thereby
11 bypassing network security protective provisions.

12 CA-certificate theft can occur in numerous ways— through spying, network
13 attacks, or by malware. If a backdoor Trojan infects a computer, the attacker may gain
14 full access to the compromised computer and will be able to control it. The attacker will
15 therefore be able to steal any information found on the computer.

16 As an alternative to theft, fake code-signing CA-certificates can be created
17 through counterfeiting and purchased online over the dark web, obviously for nefarious
18 campaigns. To create a counterfeit certificate at the behest of a client, unscrupulous shop
19 owners use digital identities stolen from a legitimate company (or its employees) to order
20 a valid CA certificate from a respected CA-certificate issuer. In most cases, duped
21 business owners and CAs are completely unaware that their data was or is being used in
22 these illicit activities. Regardless of how fraudulent CA-certificates are obtained, cyber
23 criminals use them to diffuse malware to commit illicit transactions over the Internet,
24 sometimes even disguised and openly advertised as beneficial antivirus and disk cleaning
25 software.

26 Some of the most insidious uses of fraudulent CA certificates involve the
27 distribution of malware. Malware infections deliver and install malicious code into target
28 devices in order to commit crime, gather information, or deny services. In the early days
29 of computing, memory devices such as floppy disks, CDs, and USB-drives carried
30 viruses. Ever since the 1988 Morris worm attack, the first recorded malware exploit
31 delivered over the Internet, the cloud has become the medium of choice to infect

1 computing and mobile devices. Not only did it result in the first felony conviction for
2 cybercrime, it exposed the intrinsic vulnerability of the Internet and email to attack—
3 alarming users while inspiring hackers.

4 While fairly easy to detect and expunge, the Morris worm revealed the potential
5 chaos an effective denial-of-service attack could render. Thirty years later, network
6 infectious agents have evolved significantly in capability and in stealth, employing a
7 wide variety of attack vectors including email; web browsers (HTTP-exploits); file (FTP)
8 downloads; ad-blockers; system cleanup software; software updates and installers; java
9 scripts; Acrobat and PDF readers; media files and Flash players; and personal
10 messengers. Most network delivered malware exploits also employ fraudulent CA-
11 certificates (described previously) in order to establish trust, avoid detection, and gain
12 access. Other attack vectors gain system access through adware, when a user connects to
13 a malicious URL, or in 'typosquatting' when a user mistypes the name of a valid URL and
14 is unknowingly diverted to a hostile site.

15 Installed malware operates in various ways. In denial-of-service attacks, fork
16 bombs, ransomware, lethal viruses, and many zero-day exploits, the target knows
17 immediately of the infection because of overt system failures or messages. In interactive
18 exploits like phishing, login exploits, keyloggers, and scareware, a user is tricked to
19 willingly enter personal private information, unknowingly revealing it to a nefarious
20 party. In spyware, rootkit, eavesdropping, data scrapers, and backdoor attacks, the
21 malware surreptitiously invades the target using evasion methods to 'cover its tracks',
22 avoiding detection and erasing all evidence as to its presence and its origin. Still another
23 class of time-delayed malware including time bombs and logic bombs waits unnoticed till
24 the conditions arise to release its payload, either to damage the target device or to launch
25 a pandemic against other devices. More advanced attack vectors called Frankenstein
26 malware exploits deliver hostile components in a series of innocuous-looking 'benign
27 binaries' disguising their true malicious intent. After delivery, the component pieces are
28 collected and stitched together to activate the attack.

29 With the rapid growth of mobile devices and apps, cyber criminals are turning
30 their attention to attacking smartphones and tablets, in part because they offer less-
31 sophisticated means to detect or prevent intrusion, and because they tend to contain

1 significantly more personal information than the data commonly stored in PCs. For
2 example, researchers recently identified an Android Trojan in the wild (dubbed
3 KevDroid) disguised as anti-virus application. The most recent instantiation of the
4 malware features severally disturbingly sophisticated capabilities, including the ability to:

- 5
- 6 • Record phone calls & audio
- 7 • Steal web history and files
- 8 • Gain root access (take control)
- 9 • Steal call logs, SMS, emails
- 10 • Collect device' location (every 10 seconds)
- 11 • Collect a list of installed applications
- 12

13 These capabilities allow a criminal, gang, or crime syndicate to track a person's
14 whereabouts; monitor their voice, text and email communications, engage in fraudulent
15 bank transactions, and commit extortion or blackmail. As such the loss of privacy
16 through a security breach could seriously endanger a target, reaching far beyond the
17 realms of cybercrime. As an open source platform, Android suffers the lion's share of
18 malware attacks for any mobile operating system. While the vast majority of these attacks
19 are delivered through network connectivity, either WiFi or wireless communication
20 networks, as a multisource market, mobile phone makers can also introduce OEM
21 manufacturer unique vulnerabilities such as back doors, and pre-loaded malware.
22 Although iOS and iPhones are less susceptible to unwanted incursion, a variety of attack
23 vectors and iPhone attack stratagems have been reported.

24 Assuming that Internet-based attacks cannot be prevented, virus checkers and
25 firewalls are sometimes used to combat cyber insurgencies. For global businesses today,
26 however, there is no realistic means to facilitate a firewall to cover an international
27 footprint without employing cumbersome high-latency virtual private networks (VPNs).
28 Even worse, virus checkers are often only able to detect an attack after the infection has
29 occurred. Advanced attacks like Frankenstein binary-fragmented malware avoid
30 detection altogether. As such, preventing cyber attacks over an open public network such
31 as the Internet remains the focus of numerous research efforts. Network carried malware

1 constitutes a serious impediment to trusted commerce and growing risk to personal
2 privacy and safety.

4 **III. Data Breaches**

5 According to the United States Department of Health and Human Services, a data
6 breach is “a security incident in which sensitive, protected or confidential data is copied,
7 transmitted, viewed, stolen or used by an individual unauthorized to do so.” While a data
8 beach can occur by hacking network communication (described previously), most
9 breaches occur using methods designed to *steal or corrupt stored data files*. Stored in
10 corporate or government databases, online in cloud storage, or in massive storage farms
11 or server farms, such data may involve active records, regular backup files, archival data,
12 or catastrophic recovery files.

13 Data breaches impact every aspect of modern life, both personal and public.
14 Attacks may be made on financial records, business transactions, trade secrets and
15 intellectual property, client lists, personal information, social security and tax records,
16 government employee records, active military personnel records, veteran associates
17 records, insurance records, credit reports, personal health information, files for social
18 media platforms, and personal cloud storage or containing, pictures, and other private
19 information. Cloud-based backup storage files containing virtually all of the foregoing
20 private data are especially valuable targets for hackers. The motive of data breeches can
21 be for financial gain, for espionage, or for “FIG” (fun, ideology, grudge). In identity theft,
22 for example, social security information, drivers licenses, passports, addresses, email
23 addresses, phone numbers, etc. are stolen and then used to create fake IDs to commit
24 fraud or theft, to circumvent homeland security authorities, or are sold to spammers for
25 marketing campaigns. Incalculable personal and commercial harm has resulted from
26 security breaches involving access to or theft of commercial and personal private data.

27 While many of the data breach and data-storage attacks involved theft of large
28 blocks of memory *en masse*, others attacks were more targeted, suggesting spying and
29 profiling were employed to maximize the attack’s value or impact. A particularly
30 nefarious database attack is *identity usurpation* where perpetrators corrupt or erase a
31 target’s identity in a database altogether as though they never existed and usurp their

1 personal identifiable information (PII). Although hardcopy, backup storage, and unrelated
2 data bases can be used to reestablish a person's identity, the recovery process could be
3 arduous and the financial impact to a person or business devastating.

4 In another class of data breach, *transactional record attacks* are also possible. For
5 example, an effective transactional attack on a bank's database could transfer or misdirect
6 funds from one bank account to an offshore account then erase all records of the illicit
7 transfer. Without hardcopy backup, a victim would have no means by which to prove the
8 theft occurred or that they ever even owned the money. The same class of database attack
9 could, at least theoretically, be launched against insurance databases, against
10 corporations, or against government agencies such as veteran's administration, the social
11 security agency, or federal banks. Blockchains, may reduce transactional record fraud by
12 producing indelible ledger chronicling a sequence of transactions, but only if the
13 blockchain's integrity can be guaranteed as secure.

14 Since numerous commercial databases involve proprietary interfaces and
15 allegedly 'secure' protocols, many database companies promote a false narrative that
16 their database is incorruptible despite overwhelming evidence to the contrary. Like trust
17 attacks and network attacks, data breaches are made possible because of an overreliance
18 on cryptography. To break into a database, there is no need for a perpetrator to crack the
19 encryption; just steal the account password or circumnavigate the security gates. For
20 example, in one database exploit called *SQL injection*, an attacker adds Structured Query
21 Language (SQL) code to a web form input box to gain access to resources or make
22 changes to data. Through automation, it is believed professional hackers, so called 'black
23 hats', are developing freeware hacker tools for SQL injection to steal passwords, inject
24 worms, and access data, potentially exposing 60% of all web applications using dynamic
25 content. This vulnerability cannot be avoided because user access to virtually all
26 databases and storage files today occurs over the Internet, an intrinsically insecure
27 communication medium.

28 29 **IV. Blockchain Attacks**

30 Blockchain and cryptocurrency-based e-commerce comprise a decentralized
31 transactional process using cryptographic digital-ledger technology (DLT) recordkeeping

1 and distributed consensus validation. Using cryptography to protect blockchain content,
2 transactions are broadly purported to constitute reliable, secure, un-hackable processes
3 applicable for commercial, legal, and personal transactions. Moreover, because they
4 employ immutable digital ledgers not controlled by a central authority, blockchain
5 records are not subject to backdating, record tampering, or post revision.

6 As shown in **FIG. 6**, blockchain transactional processing involves a blockchain
7 application 65 executing a validated transaction 70 using blockchain ancestry 60 to
8 validate and indelibly record and chronicle the transactional process. A blockchain is a
9 digital ledger arranged sequentially in time from the past to the present and stored in
10 multiple instances in a cloud, network, or system. Topologically, a blockchain represents
11 a one-dimensional directed acyclic graph (or 1D DAG) having a single *unitary* chain of
12 main blocks 61. The main chain may include forks 62 leading to orphan blocks 63 and
13 extinct side chains, all arranged sequentially and time stamped. The term acyclic means
14 the chain does not form loops whereby a side chain rejoins the main spine of the chain
15 forming a cyclic loop. The formation of a loop or cycle is prohibited as it can create a
16 temporal sequencing paradox where a new block appended to the end of the chain has a
17 time earlier than the prior block.

18 In unitary blockchains, only one chain remains active and extant. Sidechains are
19 simply dead ends generally used to record supplemental information not endemic to the
20 main chain's function and purpose. The blockchain and its blockchain ancestry 61 may be
21 published, i.e. made public, or kept private. Blockchain transactions can be validated by a
22 central authority 71 (like a bank or government) or confirmed using decentralized
23 methods involving a jury of peers 72.

24 Since the blockchain contains both cryptographic and unencrypted data,
25 publication over public clouds does not necessarily constitute a privacy breach or public
26 disclosure. It depends how the data is represented in the appended blocks. If a block in
27 unencrypted, all parties with access to the blockchain ancestry can retrieve the data. If the
28 block is encrypted, only parties with the corresponding decryption key are able to view
29 its contents. The most common method is to synthesize new blocks using a cryptographic
30 hash, a unidirectional cipher producing a fixed length ciphertext output unique to its
31 input. Even though a cryptographic hash has no decryption key, it has unique value in

1 confirming authenticity in a public venue without actually disclosing the actual content of
2 the hash. Without ever revealing the content of file X, the authenticity of file Y can be
3 validated by comparing the hash value of the two files, namely is $H(X) = H(Y)$ or not? If
4 the two hashes are equal then the contents file Y is confirmed to be the same as the
5 contents of file X and the transaction can proceed. If the hash values do not equal, i.e. if
6 $H(X) \neq H(Y)$ then file X is not the same as file Y, but the holder of file X never had to
7 reveal its content or trust another party not to leak confidential information.

8 Referring again to the blockchain transaction process of **FIG. 6**, application 65
9 produces confidential data file DF 56 it wishes to append onto blockchain BA 60.
10 Blockchain processor 67 then combines file DF 56 with BA 60 and creates hash 68
11 comprising ciphertext file $H(DF+BA)$. It also produces an unencrypted plaintext header
12 file 69. In transaction validation 70 the value of $H(BA)$ and $H(DF+BA)$ are compared to
13 confirm that application 65 is truly the source and owner of file DF 66. The validation
14 may be performed by authorization by a central authority 71 or in a decentralized system
15 through consensus of a jury of peers 72. Once validated, pending transaction block 64 is
16 appended onto the end of the blockchain BA 60.

17 All the blockchain processing is performed as an application in the OS application
18 VM 8 of **FIG. 1**. Communication 15 among parties engaging in the blockchain
19 transaction also is executed using the Application Layer-7, the top layer of TCP/IP 7. The
20 populist tech articles describing the blockchain processing and its execution as a
21 'blockchain network' is misleading since the entire process is hosted by the device
22 application VM and executed across the Internet using TCP/IP. All vulnerabilities of
23 TCP/IP apply equally to blockchain processing. Blockchains are therefore not immune to
24 network security risks as is often purported.

25 One use of blockchain technology is in the generation and use of cryptocurrency.
26 In cryptocurrency generation using proof-of-work (PoW), a process known as 'mining',
27 digital processors execute computationally intensive brute force methods to solve a
28 mathematical challenge, e.g. discovering the next prime number or guessing the solution
29 to a hash-nonce puzzle. A solution once discovered, is validated as authenticate and
30 correct by a jury of peers, and only then is the result appended onto the end of the
31 blockchain and the miner paid for their result with a newly generated coin. Such a

1 process represents an energy intensive and arduous process with no guarantee of any
2 financial return whatsoever. Another problem with unitary blockchains is the longer they
3 become the more memory they consume and the slower they are to use in transactional
4 commerce.

5 Moreover, the entire process of remunerating miners and storing digital coins in
6 wallets or in currency exchanges is fraught with risks of theft and transactional fraud,
7 enabled by the security vulnerabilities intrinsic to the Internet and its absolute reliance of
8 the TCP/IP communication protocol. Even so, many decentralized cryptocurrency
9 advocates believe that the risk of criminality is still preferable over the potential abuses of
10 world governments' invasive control of international fiat currency and monetary policy.

11 Decentralized blockchain and cryptocurrency transactions are often characterized
12 as operating in “trustless” systems– a somewhat confusing reference to the lack of a
13 central authority needed to validate transactions. More accurately, decentralized
14 blockchains don’t eliminate trust– they shift reliance from a single party to a group of
15 interconnected computer nodes acting as a jury-of-peers distributing control and
16 eliminating single point system failure risks. Despite these beneficial features, in practice
17 numerous attack strategies have been devised to corrupt blockchain transactions and steal
18 cryptocurrency, to launch security and privacy attacks, or to engage in criminality or
19 other malicious online activities.

20 Other attacks seek to exploit blockchain’s decentralized trustless consensus as a
21 weakness. Cyber assaults on blockchains and cryptocurrency primarily involve financial
22 fraud, security breaches, and privacy attacks. The attacks can be perpetrated on the
23 network Layer-3 using sniffing and other means to capture cryptographic keys, using a
24 DoS attack, or directly corrupting the blockchain transaction itself. Blockchain attacks
25 can be grouped into several broad classes including blockchain fraud, cryptocurrency
26 theft, malware attacks, privacy leakage, blockchain illegality, and smart contract fraud.

27 28 **Blockchain Fraud**

29 Generally for economic gain, blockchain fraud comprises any method obfuscating
30 illicit cryptocurrency activity while intentionally subverting timely validation of bonafide
31 transactions. These blockchain attacks typically utilize two specific types of online

malicious activities: double-spending and record hacking. In *double-spending* fraud, cryptocurrency is intentionally spent twice where only one transaction is valid. To complete the illicit transaction, the perpetrator must avoid detection using various means such as system disruption, misdirection, or disguise.

In a 51% attack, for example, a group of miners controlling a majority of the network's mining hashrate or computing power intentionally impede the peer consensus process preventing confirmations of bonafide transactions in lieu of preferentially enabling illicit ones. The 51% majority attack vulnerability highlights a significant weakness of proof-of-work (PoW) consensus protocols used to validate transactions in decentralized processes. Specifically because parties with the most computing power can unapologetically usurp control with no vested interest in protecting the integrity of the cryptocurrency against fraudulent transactions or out-of-sequence spending, perpetrators controlling peer consensus are able to commit fraud with impunity.

Blockchain fraud enables double spending through a variety of mechanisms including race attacks, Finney attacks, Sybil attacks, timejacking, and variants thereof. Even without majority control, mining monopolies can constrain rapid resolution of transactions improving their odds of launching undetected frauds, especially when the number of newly generated coins rewarded to miners declines as a particular currency such as Bitcoin matures and traders become more desperate (the so-called Tragedy of Commons). Considering the benefits of mining monopolies, it is not surprising that PoW based cryptocurrencies are less decentralized than previously believed.

In blockchain *record hacking*, a perpetrator corrupts a blockchain by either inserting unverified fraudulent blocks into the blockchain, or by creating a hardfork in a blockchain for nefarious purposes. Once the blockchain is corrupted, unless it is rejected before the next transaction, the damage is nearly irreversible. The design-around of a corrupted block is to launch a hardfork prior to the offending event while rolling back (cancelling) the main blockchain branch and revoking all subsequent transactions. Such a remedy is unpopular, extremely problematic, and possibly illegal in specific jurisdictions because the cryptocurrency, once spent, is unrecoverable.

Advocates of the hardfork sanction hold the position that a perpetrator's benefit should be expunged, i.e. to unwind the theft, but in so doing it penalizes coin holders

1 having executed legally valid transactions subsequent to the fraud. Opponents of the fork
2 include those penalized by the action and philosophical blockchain purists adhering to the
3 principle of *caveat emptor*, that blockchains should remain irrevocable even when fraud
4 occurs.

6 **Cryptocurrency Theft**

7 One of the present-day risks of e-commerce based on decentralized currency is
8 the possibility of theft without any recourse to recover stolen assets. The thefts, totaling
9 hundreds of millions of dollars, have occurred by attacks on robbing cryptocurrency
10 mining companies, mobile wallets, end-points (devices), and even over WiFi. Other
11 attacks focus on digital currency exchanges and web hosts. Many cryptocurrency thefts
12 involve simple password hacking through malware and spyware (see next section),
13 misplaced trust through fake CA-certificates, cryptographic key theft, packet sniffing,
14 reliance on unsecure third parties, use of fake currency exchanges, or engaging in
15 unsecured online transactions.

16 In short, an unsecure Internet is not able to protect cryptocurrency from online
17 theft. One method to execute online theft employs phishing for cryptocurrency wallets
18 using a login exploit. In this attack, a cybercriminal redirects the login window to a fake
19 website where the victim willingly enters their password and login information,
20 unknowingly passing it to the hacker who subsequently uses it to log into the real site to
21 steal their funds. Such exploits depend on a fake SSL certificate. While preventing login
22 exploits over the Internet is not certain, the risks can in part be mitigated by employing
23 site-specific unique passwords, multi-factor authentication, and by carefully checking all
24 site's SSL certificate for signature authenticity. More importantly, the majority of funds
25 should be stored offline in 'cold storage'.

27 **Blockchain Malware Attacks**

28 Malware attacks on blockchains represent another risk to digital currencies. A
29 recent paper presented at the RSA conference revealed 146 different types of malware
30 designed to steal Bitcoins. These techniques include Trojans, viruses, and spyware
31 designed to log personal keystrokes, steal cryptocurrency wallet passwords, capture

1 screen shots, or even stream video-screen images live to a hacker. If a target's computer
2 is infected, their CPU can be commandeered to mine new cryptocurrencies with the high
3 electrical power consumption and utility bills paid for by the unaware victim. Malware
4 infections can also sequester devices for use in botnets— massive networks of malware-
5 infected computers used to execute attacks on blockchains, cryptocurrency wallets, and
6 the devices storing them.

7 Other attacks involve viruses designed to search out the wallet.dat files
8 containing the private cryptographic keys used to protect the wallet and its contents.
9 Although the wallet can be encrypted, if the infection includes a key logger, typing the
10 password even once gives the hacker the ability to open the wallet, steal (relocate) the
11 cryptocurrency to their own accounts, or to change the password to remove owner access.
12 Once transferred, coin traceability is altogether lost. Another form of malware intercepts
13 cryptocurrency transfers. The virus surreptitiously infects a computer waiting silently
14 until the infected device copies a Bitcoin address whereupon the malware immediately
15 becomes active to launch its attack by changing the IP address redirecting the coin
16 transfer to the hacker's account. A particularly aggressive attack involves ransomware,
17 malware demanding payment in Bitcoin or other cryptocurrency in exchange for
18 unlocking an infected computer or its files.

19 An endpoint attack uses malware specifically focused on interfering with devices
20 participating in a cryptocurrency exchange such as the purchaser, the merchant, and the
21 cryptocurrency wallet holding the coins to be exchanged. As such the attack is performed
22 on the terminus devices, not the nodes carrying the transaction through the network. Best
23 practice for mitigation of malware, while imperfect, involves using firewalls, antivirus
24 software, and offline operation of a dedicated personal computer.

26 **Blockchain Privacy Leakage**

27 Although originally believed to deliver transactional pseudonymity protected by
28 cryptographic hash methods, in 2013 blockchainologists confirmed their ability to extract
29 the private information from an encrypted blockchain including names and account
30 numbers. This process, referred to as 'deanonymisation' executes a detailed analysis of a
31 cryptocurrency's blockchain using a combination of data-driven pattern recognition (to

1 extract commonalities in blocks), and analysis of test transactions (to uncover addresses).
2 Since a full blockchain contains blocks detailing every prior transaction, there are risks
3 not only that digital assets may be stolen but also that personal privacy and private
4 financial information may inadvertently be leaked. Private information leakage through a
5 blockchain can be leveraged by astute cybercriminals to profile targets, steal
6 cryptocurrency, engage in identity theft, or to launch personal attacks. Blockchain records
7 can also be employed to identify and selectively target more successful cryptocurrency
8 traders.

9 The vulnerability of a blockchain to deanonymisation depends on management
10 policies and personal behavior, especially in address reuse and web purchases using
11 cryptocurrency. For example, in the case of online purchases using cryptocurrency, third-
12 party trackers providing online analytics and advertising possess sufficient information to
13 “uniquely identify the transaction on the blockchain, link it to the user’s cookie, and
14 further to the user’s real identity.” Moreover, if a purchaser executes multiple online
15 transactions on the same blockchain, the user’s entire cluster of addresses can be exposed
16 including all transactions even if the user employs blockchain anonymity techniques. The
17 analytical risk is indelible and perpetual, meaning an attack can be mounted retroactively.

18 As industries migrate to blockchain records, the risk of personal privacy attacks
19 through blockchain exploits becomes increasingly a matter of concern, especially as it
20 might expose consumers to identity theft who are wholly unaware that their personal
21 information is being stored in a blockchain. This conundrum has inspired intense research
22 to address the issue of blockchain privacy leakage and how to mitigate it, including
23 proposals to bind physical entities to virtual identities as proxies in an effort to improve
24 accountability while preserving anonymity. Present proposals, despite their creativity, are
25 unconvincing, lacking any credible implementation, testing, or real-world deployment.
26 Transaction resolution speeds of these privacy-protected blockchains are expected to be
27 unusably slow.

29 **Blockchain Illegality**

30 Because blockchain can embed any type of data into its blocks, a blockchain
31 could be contaminated with illegal or objectionable material that may be illegal in

1 specific countries or jurisdictions. In a decentralized system, *arbitrary content* files
2 embedded into a blockchain are not reviewed or approved by any administrator prior to
3 inclusion. As such, there exists no means by which to manage a blockchain's content, to
4 decide what is appropriate, or to reject what is not. A number of risks result from
5 blockchain's ability to indefinitely store arbitrary content, including the risk of copyright
6 violations, stolen intellectual property, malware, privacy violations, politically sensitive
7 content, religiously offensive material, as well as illegal and condemned content.

8 Copyright violations involve the distribution, illegal downloading and
9 unauthorized use of copyrighted material involving original works of authorship,
10 including musical, dramatic, literary, artistic, and other intellectual works. Similarly,
11 stolen intellectual property involves the unauthorized disclosure, distribution, or use of
12 intangible creations of the human intellect not already public, including unpublished
13 pending patents, trade secrets, confidential work product, business plans, private
14 contracts, and other private works of creativity. In both cases, since it is impossible to
15 recall publically distributed blockchains it is difficult to ascertain the economic damage
16 caused by the unauthorized release of IP and creative works on the blockchain. Since
17 most users are unaware of illegal material contained within a blockchain, some countries
18 have begun to prosecute infractions based on the download and use of the unauthorized
19 material rather than seeking remedy from the perpetrator who uploaded the stolen
20 material.

21 Another risk of the arbitrary content field in a blockchain is the introduction of
22 malware. According to INTERPOL, "the design of the blockchain means there is the
23 possibility of malware being injected and permanently hosted with no methods currently
24 available to wipe this data," permanently impacting global *cyber-hygiene*. Malware
25 infected blockchains may involve zero-day exploits, time bombs, Trojans, or difficult-to-
26 detect molecular viruses. Once infected, blockchain malware is impossible to expunge,
27 representing an ever-present transactional risk and a continuing annoyance of triggering
28 anti-viral software alerts. The injection of politically sensitive or religiously objectionable
29 material into a blockchain strongly depends on the country or community affected by the
30 material. Political or religious views held sacrosanct in one country may be considered
31 sacrilege in another. Illegal content relating to religiously offensive content or

1 pornography also vary country-to-country. Since there is no arbitrator to ensure the
2 cyber-hygiene of a blockchain, the unknowing import of cryptocurrency containing
3 illegal or banned material into a country may result in unexpected or severe legal
4 consequences. The illegal use of blockchains may also include blackmail, extortion,
5 trafficking, or comprise threats to a sovereign nation's national security and stability.

6 7 **Smart-Contract Fraud**

8 Smart contracts represent a significant potential for both beneficial and malicious
9 use of blockchain technology. Smart contracts comprise digital code comprising an
10 executable computer program indelibly stored in a blockchain. Operating as a sequential
11 state machine, the smart contract executes a sequence of verifiable tasks and distributes
12 cryptocurrency rewards to a pool of miners based on a negotiated value for each job.
13 Although the concept of smart contracts dates back to 1996, it was nearly twenty years
14 before Ethereum offered the first smart-contract based blockchain-as-a-service (BaaS).

15 By enabling other companies to utilize its platform and blockchain, Ethereum has
16 differentiated itself from conventional cryptocurrencies focused on trading. Although
17 BaaS adoption is slow and market penetration limited, in part due to distrust in new
18 technology and bad press from reported cases of fraud, a number of exciting potential use
19 cases have emerged, mostly in financial technology (fintech) including securities, trade
20 financing, derivatives trading, financial data recording, insurance, and mortgages. Other
21 possible non-financial applications include digital identity, record keeping, supply chain
22 management, land title recording, clinical trial management, and medical research.

23 Proponents of BaaS suggest that smart contracts can be used to prevent fraud in
24 business while opponents are quick point out that smart contracts have been shown to be
25 susceptible to Ponzi schemes and other fraudulent exploits. In practice, smart contracts
26 today remain illusively problematic, facing a myriad of issues including their intrinsic
27 lack of privacy, the inability to expeditiously repel attacks, and a propensity to duplicate
28 errors using flawed drafting techniques and error-filled code propagating vulnerabilities,
29 reportedly in 44% of 19,000 Ethereum smart contracts studied. Ironically, blockchains
30 promoted as a solution to preventing distributed-denial-of-service attacks are unable to
31 combat DDoS attacks on Bitcoin exchanges relying on blockchain technology.

V. Privacy Concerns

Aside from the described security vulnerabilities of the Internet and TCP/IP, another major topic of concern today is the topic of privacy, i.e. who owns personal and business data and can the data's owner limit its availability, control its dissemination, and restrict its use. The rapid rise of social media such as Facebook, Instagram, and Twitter has created new avenues for people to share experiences or to promote their business. More recently, however, it has been revealed that the platforms hosting social media gather information about their clients and sell that information to advertisers and data analytics companies. In the world of big data, an individual may not own their own personal data nor have the right to limit other's access to it.

In a similar manner large computer networks such as AWS, GWS and Azure carry a disproportionate fraction of the world's data traffic, having annual business expected to approach \$100B USD in network revenues in 2020. The networks collect significant information about their clients including what searches a user makes, what products they buy, where they shop, and with whom they socialize. This *en masse* collection of personal data is being executed without the approval of their clients and without any disclosure as to how the information is being used or to whom it is sold. It has been alleged that the unauthorized sale of contact information such as email and phone numbers to unscrupulous telemarketers and email spanners in contributing to the junk mail and robocall epidemic now plaguing virtually everyone.

During the events surrounding Edward Snowden and his revelation that the NSA was spying on Americans and monitoring (in old school parlance "wire tapping") people's calls without a warrant or probable cause, the role of the state in practicing privacy invasion became well documented. Still other countries such as the UK and China outwardly proclaim any call may be monitored and data analyzed without warning. In recent weeks, this fact became politicized when it was revealed that Huawei routers and cell phones are capable of sending messages and call data back to China.

Corporate privacy is equally a topic of heated debate today. Merchants and corporations are reticent to share client information with data analytics companies in fear that their client contact and sales information will end up in the hands of their

competitors. Luxury brands although motivated to more thoroughly engage with existing clients while attracting new ones, are concerned that big data companies once obtaining client contact information will either inundate the VIP clients with spam and promotions or share the information with competing brands.

Despite laws demanding privacy protection, e.g. securing personal medical records in accordance with HIPAA (the Health Insurance Portability and Accountability Act of 1996) in the USA, and prohibiting data mining of a individuals actions as specified in the General Data Protection Regulation (GDPR) in Europe, there is no clear way to ensure these laws can be implemented. In reality, in order to sort and use most data, the data is stored in unencrypted form, rendering the information readily available to anyone with system access.

Because there is no such thing as "identity" in the Internet then it is nearly impossible to protect privacy. Unable to ascertain who legally owns personal or corporate data and communication metadata, it is impossible to regulate who should have access to it. It is paradoxical that privacy protection demands the need for identity. In the Internet anyone can pretend to be anyone, so identity means nothing. In short, personal and private information is misused because the Internet allows it to happen, with no provisions to stop it.

V. Decentralization

Wikipedia defines *technological decentralization* as a shift from concentrated to distributed-modes of production and consumption of goods and services. In the Internet the term applies to the network's operation including cloud computing, to communication networks, and to e-commerce. Economic motivation for decentralization includes optimizing economies of scale, matching supply and demand, fostering competition, and recruiting underutilized resources.

One means to economically enlist unused assets is through a concept referred to as 'the sharing economy'. First popularized by carpooling then later expanded by Uber and Lyft rideshare services, sharing economics seeks to employ underutilized resources by making them available to those who need them, i.e. dynamically matching supply and demand. Investopedia defines a sharing economy as "an economic model defined as a

peer-to-peer (P2P) based activity of acquiring, providing, or sharing access to goods and services that is often facilitated by a community-based on-line platform." The sharing economy is environmentally beneficial in that it reduces waste of natural resources while bringing underutilized resources to bear on solving a challenge or delivering a service. Sharing economies apply equally to distributed computing, cloud communication, disaggregated storage, blockchain services, and decentralized currencies.

Distributed Computing

Distributed computing involves dividing and spreading a task across a group of networked computers. Although it is possible to employ a private cloud of servers, one benefit of distributed computing is that a user can access and utilize computers without owning them, employing unlimited computing capacities. For example, **FIG. 7A** and **7B** illustrates the world's utilized and unused capacity in computer and servers throughout the week and within any given day. In **FIG. 7A**, the daily use curve 50 shows heavy usage from 8:00 AM to 5:00 PM but enormous unused computing capacity 51 for the other fifteen hours of the day. If we divide the world into three large time zones, it means two thirds of the world's computing resources remains unused at any time of the day. As illustrated in **FIG. 7B**, average daily use 52 of computing resources consumes nine hours every day but drops below three hours per day on the weekend, meaning the weekly total of unused computing capacity is actually 73% of all computers in a given time zone. The above analysis ignores the enormous computing capacity of cryptocurrency miners, gamers, home computers, and IoT devices (like refrigerators, air conditioners, routers, HDTVs, and the omnipresent cable set-top boxes).

If this unused computing capacity could somehow be harnessed for addressing massive tasks like weather forecasting, storm tracking, climate modeling, cancer research, searching for near earth objects (asteroids risking collision with the earth), or studying the cosmos, or supporting the search for extraterrestrial intelligence (SETI), sustainable progress could be achieved across the spectrum of topics with a minimal cost to society. One example employing a sharing economy over the Internet today involves the Berkeley Open Infrastructure for Network Computing (BOINC), which as of June 2018 averages 20 PFLOPS (the prefix P stands for peta or 10^{15}) using only volunteer

resources. BOINC hosted projects include SETI@Home and MilkyWay@Home, each consuming 4% of BOINC's total compute cycles, and Einstein@Home a study of pulsars consuming roughly 15% of BOINC's available cloud computing capacity (<https://en.wikipedia.org/wiki/FLOPS>). Once key factor in effective cloud computing is uptime whereby the host devices remain always on, meaning they are AC powered and available even when they are not being used otherwise.

Enormous computing resources are also available in mobile devices including notebooks, tablets, and smartphones. By 2020, the number of smartphones worldwide is expected to exceed three billion (<https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>). Considering the most recent generation of iPhone contains a NPU neural processor with a capacity of six hundred billion floating point operations per second or 600 GFLOPs, the total worldwide distributed computing capability of smartphones will soon reach a capacity of $(600 \times 10^9 \text{ FLOPs/device})(3 \times 10^9 \text{ devices}) = 1.8 \times 10^{21} \text{ FLOPs}$, or roughly 2 ZFLOPs (Z is prefix zeta meaning 10^{21}), five-orders-of-magnitude beyond BOINC's volunteer-based distributed computing network.

The problem today is the Internet offers no such means to identify what resources are available or underutilized or to make them accessible to users looking for such resources. And although philanthropy is admirable, it is highly doubtful that the masses will donate their compute cycles to a noble cause when they are looking for how to pay next month's utility bills. Even if a merchant is willing to compensate people to provide resources to them, no such system or marketplace exists to match merchants looking for resources to resource providers (people willing to offer their unused compute cycles) or to facilitate a means of fair remuneration for their efforts. As such, today's Internet is unable to support a computing sharing economy.

Moreover, many computing and IoT devices remain on at all times, constantly consuming power without providing any useful work or benefit to their owner or to society. In 2017, for example, the power wasted by set top boxes in the US alone was 21 TWh (terawatt-hours), roughly the power generated by five nuclear power plants, even though the majority of the time the device is doing nothing. For environmental sustainability, the set top box could be redesigned to use the same power management

1 methods used in smartphones, and ideally in a sharing economy to do useful computing
2 when it's on.

4 **Cloud Communication**

5 The opportunity of the sharing economy extends to cloud communication. A few
6 decades ago, a handful of telecommunication companies dominated the world's
7 communication traffic. With the advent of the Internet and mobile networks, the number
8 of Internet service providers (ISPs) and cellular carriers exploded. In recent years,
9 however, the growth of Infrastructure-as-a-Service (IaaS), Software-as-a-Service (SaaS),
10 and Platform-as-a-Service (PaaS) service markets has seen a rapid consolidation of the
11 market to only a few global suppliers (primarily AWS, Azure, GWS and IBM clouds).

12 This emerging cloud oligopoly reverses the decentralization trend spawned by the
13 Internet, concentrating the exclusive control of network traffic, data storage, and data
14 analytics into a group of corporations not answering to any electorate or government.
15 Although regulators can attempt to manage these mega-corporations' growth, the real
16 issue is as a fragmented platform the Internet is unable to offer a competitive open source
17 alternative to privately owned clouds, let alone deal with the security issues of an open
18 source environ. If it were possible to realize a global communication network as a sharing
19 economy, however, the prospects of creating real and viable competition to the private
20 cloud oligopoly is compelling, especially if cell phones could somehow comprise
21 portions of the network.

22 To implement a global communication network able to compete with large server
23 clouds, the network would have to operate using peer-to-peer communication (P2P) to
24 reduce network traffic and minimize reliance on pay-per-byte cloud services. Although
25 the prospect of P2P communication is impractical in rural areas, in highly populated
26 cities a peer-to-peer network has the potential for achieving good quality-of-service
27 (QoS) metrics. In fact, in 2019 it has been reported that China with 1.6B phones
28 ([https://www.thatsmags.com/china/post/27097/china-has-more-active-mobile-phones-](https://www.thatsmags.com/china/post/27097/china-has-more-active-mobile-phones-than-people)
29 [than-people](https://www.thatsmags.com/china/post/27097/china-has-more-active-mobile-phones-than-people)), has more active mobile phones than it has people.

30 "According to several working professionals in Guangzhou, they use one phone
31 solely for work while the other is kept for personal use." So although no mechanism

exists today to harness this dense mobile telephony population into a P2P network, the nodal density is sufficiently high to support services. Unfortunately TCP/IP is ill equipped to support ad hoc networks especially of disparate components based on different hardware platforms and operating systems.

Another opportunity for deploying a P2P communication network is in vehicle-to-vehicle (V2V) communications. Cities such as Los Angeles have 6M cars in an urban area of 12,561 km² or roughly 500 cars for every square kilometer. Shanghai has 3.3M vehicles compressed into an area 6,341 km², also roughly 500 cars for every square kilometer. Such densities are sufficient to host a viable P2P network. Unfortunately, as stated previously TCP/IP does not have the provision to manage ad hoc networks or carry P2P data without exposing packet content to the peer nodes carrying the data.

Decentralized Currency

As stated by Investopedia, fiat money "is government-issued currency that isn't backed by a commodity such as gold. Fiat money gives governments' central banks greater control over the economy because they control how much currency is printed." Rather than being set by the true market dynamics of supply and demand, governments and central banks are able to manipulate currency valuations by controlling supply, arbitrarily printing or retiring currency in circulation based on arbitrary, geopolitical, and self serving interests.

Early attempts to develop a decentralized currency led to the creation of Bitcoin, the world's first cryptocurrency. Using a mathematical challenge to create new tokens in a process called mining, new Bitcoins become increasingly difficult to produce thereby limiting the number of Bitcoins in circulation to some predefined quantity. The artificial scarcity imposed by arbitrarily limiting the supply of new Bitcoins combined with unpredictable demand resulted in a highly volatile currency valuations rendering Bitcoin useless for real commerce. Merchants and service providers cannot engage in commerce using a currency with an uncertain value fluctuating daily.

Moreover today's cryptocurrency is ecologically unsustainable. Environmentalists consider cryptocurrencies as realized today as a flagrant waste of our planet's natural resources. Electrical power consumption in 2018 for mining the two largest

cryptocurrencies Bitcoin and Ethereum reached 83 TWh, greater than the annual energy consumption of all but the top-40 biggest energy consuming countries on planet Earth. This consumption has spawned a firestorm of controversy about energy waste balanced against the potential yet unproven benefits of cryptocurrency. The issue of energy waste cannot be resolved by present day cryptocurrencies since they all rely on a mining and consensus protocol referred to as Proof-of-Work (PoW). PoW is by design intentionally energy efficient as it was originally developed to economically discourage hackers from attacking a network.

Another feature of cryptocurrency is its inescapable reliance on blockchain technology to ensure a trusted pedigree required to prevent fraud and double spending in a decentralized currency system. To enable verification of the Bitcoin family tree, traceability extends to its origins including every mining event producing new coins, every coin transfer, and every hardfork and softfork stemming from the main blockchain. The resulting impact of this thorough record keeping is three-fold, namely (a) the blockchains become excessively long, (ii) the resolution speed (time needed to confirm the coin's veracity) becomes slow, and (iii) if the trade takes too long, its payee will not do a thorough job on confirming the coin's validity.

Incomplete checking invites fraud and double spending exploits. The longer a blockchain exists in the wild, the longer it becomes and the longer it takes to check. Today, the size of each new Bitcoin is 156.4 GB long and growing. The memory requirement for each Bitcoin is now becoming prohibitive, in that it is too large to carry or conveniently use. With each global transaction the length of the blockchain grows and the size of the memory needed to store a coin increase. The incremental increase in memory size for each new blockchain entry typically varies between 0.5 to 1.0 MB depending on the type of transaction executed.

The other major concern with cryptocurrency today is one of scalability. Using Proof-of-Work, the more people who use a PoW cryptocurrency the longer the unitary blockchain gets and the more difficult it becomes to use. For example, hypothetically should Bitcoin become a global currency studies reveal it would become nearly useless adding hundreds of gigabytes to the blockchain every day. For the sake of argument, assuming it requires 150 minutes to fully validate each 8-MB block, if the Bitcoin

1 became a global dominant currency its block size would necessarily swell to 2.4 GB,
2 taking over 51,000 minutes (over 2 years) to validate. While alternative consensus
3 protocols such as Proof-of-Stake have been proposed, they primarily address issues
4 regarding blockchain attacks rather than improving speed performance. Even so, such
5 consensus methods remain exclusively the topic of whitepapers and conferences.

6 In essence today's realization of cryptocurrency represents a financial instrument
7 to cumbersome and to volatile to use in real commerce.

9 **VI. Conclusion**

10 In conclusion, despite it ubiquitous global reach, the Internet was never designed
11 as a platform to ensure security, protect privacy, host e-commerce, or to support a sharing
12 economy. The Internet's greatest weaknesses are now fully evident, namely

- 14 • No security
- 15 • No privacy
- 16 • No trust

17
18 These issues remain untenably problematic yet unresolvable. Attempts to ameliorate
19 these deficiencies using cryptography and certificate-authority based digital signatures
20 have failed wholly and absolutely as evidenced by the relentless growth in the severity
21 and magnitude of cybercrime.

22 Clearly, the Internet is not well suited as a global platform for securely and
23 privately facilitating real-time communication, decentralized cloud computing, cloud data
24 storage, cloud connected devices, electronic transactions, and e-commerce. Since
25 virtually all-modern network-communication, whether realized globally as the Internet,
26 or implemented locally as a subnet employ TCP/IP as its foundational communication
27 protocol, then all communication and e-commerce is exposed to vulnerabilities intrinsic
28 to TCP/IP. The vulnerability is omnipresent regardless of the physical medium used to
29 carry the data be it optical fiber, a wired Ethernet network, a wireless WiFi LAN; cellular
30 networks; cable distribution networks; satellite communication networks, or other peer-
31 to-peer networks.

1 Aside from TCP/IP vulnerabilities, additional security and privacy risks are
2 manifest in device hardware, and in physical and data link communication layers,
3 allowing the interception or manipulation of data packet communication contents and
4 metadata, corruption of memory contents, the overwriting of system and application
5 software, the undetected installation of malware, backdoors, spyware, and phishing
6 software, and the possible usurpation of an operating system.

7 No present day communication system or network securely executes
8 cryptocurrency transactions, provides for the safe storage of cryptocurrency, prevents for
9 the unauthorized use of fake or stolen cryptocurrency, or ensures that decentralized
10 transactional verification has not been commandeered by cyberbots programmed to
11 validate fraudulent transactions while thwarting or denying the timely execution of valid
12 transactions. No present day communication system or network is able to avoid denial-of-
13 services attacks perpetrated at one or multiple OSI communication layers.

14 No present day communication system or network is able to ensure minimal
15 propagation times and network latency needed for reliable realtime communication, to
16 guarantee real-time QoS, or to adjust data packet transport efficiency, redundancy and
17 urgency to match the nature of the data packets or to a user's demands. No e-commerce
18 system is able to ensure stability in the cost of digital transactions for e-commerce using
19 utility tokens while still supporting investor desire for market-demand based
20 cryptocurrency volatility.

21 No present day communication system or network is able to effectively reallocate
22 global communication, computing, and data storage capacity to maximize commercial,
23 research, and philanthropic utility in accordance with market dynamics to avoid the
24 underutilization of resources, the wasteful consumption of energy, or the egalitarian
25 control of global network assets and compute capacity. No present day communication
26 system or network is able to effectively to block massive data breeches risking the
27 privacy of every person on earth.

28 A more thorough description and analysis of present day network and
29 transactional vulnerabilities are specified in provisional patent application 62/696,160
30 entitled "The HyperSphere– a Real-time Cybersecure Privacy Network with Embedded

1 DyDAG Dual Cryptocurrency for Global e-Commerce," filed 10 Jul 2018 and included
2 herein by reference.

3 What is needed is a completely new approach to cloud-based communication,
4 networking, computing, data storage, and decentralized cryptocurrency that addresses the
5 aforementioned issues of privacy, security, performance, social responsibility, and
6 sustainability while discouraging criminality, fraud, and hacking of the network and the
7 devices connected to it. Even the Internet's visionary founders Tim Berners Lee (creator
8 of the World Wide Web) and Steve Wozniak (inventor of the personal computer) concur,
9 publically professing that fixing this myriad of problems will require a complete rebuild
10 of the Internet.

11 12 **SUMMARY OF THE INVENTION**

13 In accordance with this invention, data is transmitted between client devices such
14 as cell phones or laptop computers over a network or "cloud." The cloud includes a
15 plurality of nodes that are individually hosted on servers or other types of computers or
16 digital equipment. While in transit between the media nodes in the SDNP cloud, the data
17 is in the form of packets, discrete strings of digital bits that may be of fixed or variable
18 length.

19 Software installed in the nodes allows them to perform three functions, referred to
20 herein as "name server," "authority" and "task" functions. The "name server" function
21 entails the management of a dynamic list of the client devices that are connected to the
22 cloud. The "task" function entails the receipt and transmission of the packets as they
23 proceed from node to node through the cloud. The "authority" function entails the
24 determination of the respective routes of the packets through the cloud, e.g., from node A
25 to node B to node C, and the transmission of "command and control" packets to each of
26 the nodes on the route, instructing the node where to send the packet on the "next hop"
27 through the cloud. The packets may be fragmented, i.e., broken up into subpackets that
28 travel over different routes and are united in the destination client device. The packets
29 and subpackets may be scrambled and/or encrypted in accordance with different, state-
30 based algorithms as they pass through a node. An aspect of the task function is to
31 determine by which algorithm an incoming packet should be unscrambled or decrypted

1 and by which algorithm an outgoing packet should be encrypted or scrambled.

2 The nodes are “metamorphic,” which refers to the fact that each node is capable
3 of performing the name server, authority and task functions, but it does not perform more
4 than one function simultaneously. A node that is not performing a function is referred to
5 as “undifferentiated.” The software installed in a node enables it to determine from the
6 nature of a request for it to act whether it is being requested to perform the name server,
7 authority or task function. After completing its designated job, a node reverts to its
8 “undifferentiated” state awaiting its next performance request, and it forgets all
9 information regarding its last action.

11 **BRIEF DESCRIPTION OF THE DRAWINGS**

12 **Fig. 1** is a block diagram of computing device with TCP/IP communication stack.

13 **Fig. 2** is a TCP/IP datagram of packet routing.

14 **Fig. 3** is an illustration of TCP/IP communication stack and datagram
15 construction.

16 **Fig. 4** illustrates Internet packet hijacking through man-in-the-middle (MiM)
17 attack.

18 **Fig. 5** shows Internet certificate authority.

19 **Fig. 6** shows Internet blockchain processing.

20 **Fig. 7A** illustrates daily server and computer capacity.

21 **Fig. 7B** illustrates weekly server and computer capacity.

22 **Fig. 8** shows an example of a Secure Dynamic Communication Network And
23 Protocol (SDNP) meshed network.

24 **Fig. 9** illustrates the elements of an SDNP meshed network.

25 **Fig. 10** illustrates a flowchart of SDNP routing over meshed network.

26 **Fig. 11A** illustrates SDNP registration operation.

27 **Fig. 11B** illustrates SDNP name server query operation.

28 **Fig. 11C** illustrates SDNP routing request operation.

29 **Fig. 11D** illustrates SDNP network node request operation.

30 **Fig. 11E** illustrates the SDNP command and control (C&C) routing instruction
31 dispatcher operation.

1 **Fig. 11F** illustrates the SDNP first packet transport operation.

2 **Fig. 11G** illustrates the SDNP second packet transport operation.

3 **Fig. 11H** illustrates the SDNP third packet transport operation.

4 **Fig. 11I** illustrates the SDNP fourth packet transport operation.

5 **Fig. 12A** illustrates an example of the decentralized SDNP meshed network.

6 **Fig. 12B** is a representation of an alternative decentralized SDNP meshed

7 network.

8 **Fig. 12C** shows elements of decentralized SDNP meshed network.

9 **Fig. 12D** is a flowchart of decentralized SDNP routing over a DyDAG meshed

10 network (part 1).

11 **Fig. 12E** is a flowchart of decentralized SDNP routing over a DyDAG meshed

12 network (part 2).

13 **Fig. 13A** illustrates the operation of a decentralized SDNP registration and name

14 server query.

15 **Fig. 13B** illustrates the operation of a decentralized SDNP routing request.

16 **Fig. 13C** illustrates the operation of a decentralized SDNP network node request.

17 **Fig. 13D** illustrates the operation of a decentralized SDNP C&C routing

18 instruction dispatcher.

19 **Fig. 13E** illustrates the operation of first decentralized SDNP datagram packet

20 transport.

21 **Fig. 13F** illustrates the operation of second decentralized SDNP datagram packet

22 transport operation.

23 **Fig. 13G** illustrates the operation of third decentralized SDNP datagram packet

24 transport operation.

25 **Fig. 13H** illustrates the operation of fourth decentralized SDNP datagram packet

26 transport operation.

27 **Fig. 14** illustrates the selection criteria for HyperNode access to a d'SDNP diffuse

28 data cloud.

29 **Fig. 15A** illustrates the transformation of a metamorphic HyperNode into an |NS|

30 node for call initiation.

31 **Fig. 15B** illustrates the download of caller ID information from a name server's

diffuse cloud data into a differentiated $|NS|$ node.

Fig. 16A illustrates the transformation of a metamorphic HyperNode into an $|A|$ node for route planning.

Fig. 16B illustrates the download of network hop times from an authority node's diffuse cloud data into a differentiated $|A|$ node.

Fig. 16C illustrates a d'SDNP authority node's processing of available task node, rank sorted by network hop times.

Fig. 17A illustrates the transformation of a metamorphic HyperNode into a $|A|$ node for route dispatching.

Fig. 17B illustrates the download of a task node translation table listing current dynamic IP addresses corresponding to selected SDNP addresses.

Fig. 18A illustrates the transformation of a metamorphic HyperNode into a $|T|$ node for routing in response to $|A|$ node's command and control (C&C) routing command.

Fig. 18B illustrates the download of d'SDNP task node shared secrets comprising dynamic concealment algorithms (including a single-hop encryption cipher).

Fig. 18C illustrates a d'SDNP task node processing datagrams using dynamic concealment methods for incoming and outgoing packets.

Fig. 19 is a schematic representation of a multi-layer diffuse data cloud.

Fig. 20 is a schematic representation of various methods for realizing and accessing the HyperSphere's diffuse-data cloud.

Fig. 21 is a schematic representation of realizing diffuse cloud storage using disaggregated data.

Fig. 22 is a schematic representation of the redundant file management of a d'SDNP name server's diffuse-data cloud storage and access keys.

Fig. 23 is a flow chart for updating name server's diffuse-data cloud for HyperNode dynamics.

Fig. 24 is a flow chart of dynamic meshed data routing in a HyperSphere's decentralized cloud of task nodes.

Fig. 25 illustrates packet transport of d'SDNP data utilizing hop-by-hop state-based dynamic concealment algorithms.

Fig. 26 shows d'SDNP datagrams illustrating the division of labor between [NS], [A], and [T] differentiated HyperNodes.

Fig. 27 is a hierarchical representation of a HyperSphere's 7-layer OSI communication stack and corresponding d'SDNP datagram.

Fig. 28 is a graph of heterogeneous cloud hosting homogeneous HyperNodes.

Fig. 29 illustrates the types of HyperSphere portals comprising software performing communication, computing, data storage, and cloud-connected devices.

Fig. 30 shows an example of a HyperSphere communication application showing operation of personal messenger.

Fig. 31 shows an example of the dynamic concealment of a messenger payload during d'SDNP datagram transport.

Fig. 32A is a graph of a vehicle-to-infrastructure (V2I) based network over a mobile (cellular) carrier network.

Fig. 32B is a graph of an autonomous *ad hoc* vehicle-to-vehicle (V2V) network maintaining indirect mobile network access.

Fig. 32C is a graph of a fully autonomous *ad hoc* vehicle-to-vehicle (V2V) network (micro-cloud) isolated from mobile network access.

Fig. 32D is a graph of a fully autonomous *ad hoc* vehicle-to-vehicle (V2V) network isolated from mobile network access, dynamically reconfigured to access available HyperNodes.

Fig. 32E is an updated graph of a fully autonomous *ad hoc* vehicle-to-vehicle (V2V) network isolated from mobile network access, dynamically reconfigured to access available HyperNodes.

Fig. 33A is a schematic representation of a HyperSphere HyFi wireless router using multiple (802.11 compatible) OFDM modulated microwave carriers.

Fig. 33B shows an example of HyFi router d'SDNP multi-PHY communication using fragmented data transport over multiple microwave radio channels and multiple WiFi protocols.

Fig. 34A is a schematic representation of a HyperSphere Ethyrnet wireline router capable of multiple channels of IEEE 802.3 compatible (multi-PHY) communications.

Fig. 34B is an example of Ethyrnet router multi-PHY communication using

d'SDNP fragmented data transport over multiple 802.3 compatible Ethernet mediums and protocols.

Fig. 35A is a schematic representation of a HyperSphere cable system comprising d'SDNP enabled communication stacks for cable modem termination system (head unit) and multiple cable modem (or STB) downstream devices.

Fig. 35B is an example of d'SDNP fragmented data transport employing trellis coded multi-channel DOCSIS3 cable communication.

Fig. 35C is an example of the trellis coding of multiple d'SDNP channels and HDTV content during DOCSIS3 compatible cable modem operation.

Fig. 36A is a schematic representation of HyperSphere enabled mobile network communication using d'SDNP transport of fragmented data packets over multiple carrier frequencies and cellular protocols (4G and 5G shown by example).

Fig. 36B is a schematic representation of edge device (mobile phone or tablet) access to HyperSphere enabled mobile network, illustrating d'SDNP transport of fragmented data packets over multiple carrier frequencies and cellular protocols (4G and 5G shown by example).

Fig. 37A is a schematic representation of HyperSphere enabled IoT device over 801.11 protocol compatible microwave communication.

Fig. 37B is a schematic representation of a HyperSphere IoT device illustrating d'SDNP network to IoT datagram download and upload content.

Fig. 37C is a schematic representation of HyperSphere HyFi network control of IoT device cloud.

Fig. 38A is a schematic representation of a hypersecure cloud for disaggregated data storage of client-owned files with redundant storage elements and recovery mechanisms.

Fig. 38B is a schematic representation of the access control of user-owned diffuse-data cloud including both private data and collaborative files.

Fig. 39A is a flow chart showing a HyperContract job specification of a HyperSphere cloud computing application.

Fig. 39B is a flow chart showing the hypersecure job execution of a HyperSphere cloud computing application.

1 **Fig. 40** illustrates the account opening process.

2 **Fig. 41** illustrates how an offline root certificate is used to sign an online CA-
3 certificate.

4 **Fig. 42** illustrates the process of forming a trust chain by using trusted verified
5 certificate to sign antecedent certificates.

6 **Fig. 43** illustrates the process of forming a trust chain by using trusted verified
7 certificate to sign antecedent certificates.

8 **Fig. 44** illustrates personalized and multi-tree DyDAG blockchains.

9 **Fig. 45** depicts a comparison of cyclic and multi-dimensional directed graphs.

10 **Fig. 46** shows 2D and 3D representations of dynamic acyclic graphs (DyDAGs).

11 **Fig. 47** illustrates HyperContract task-based cryptocurrency remuneration of task
12 HyperNodes.

13 **Fig. 48** illustrates a HyperSphere merchant API generated HyperContract and
14 pledge.

15 **Fig. 49** illustrates the synthesis of HyperCoins through minting and recycling.

16 **Fig. 50** is a flow chart of HyperCoin minting and juror validation through
17 HyperContract execution of transient and irrevocable blockchains.

18 **Fig. 51** illustrates conjunctive HyperNode hop code (HHC) generation during
19 data packet transport.

20 **Fig. 52** is a flow chart of HyperCoin recycling and juror validation through
21 HyperContract execution of transient and irrevocable blockchains.

22 **Fig. 53** is a block diagram depicting SDNP stack and device host with Layer-3
23 blockchain generation and processing.

24 **Fig. 54** illustrates a HyperSphere process sequence for executing blockchain
25 defragmentation.

26 **Fig. 55** illustrates a HyperSphere blockchain process for the creation and
27 validation of auxiliary sidechains.

28 **Fig. 56** is a schematic representation of HyperSphere payment processing using
29 one-time transaction tokens (OT³) facilitating preauthorized payment proxies.

30 **Fig. 57** is a schematic representation of HyperSphere rapid transaction validation
31 and privacy protection using replicant blockchain observer segments (RBOS).

1 **Fig. 58** is a schematic representation of the use of HyperNode tunnels to prevent
2 consensus surround attacks of jurors.

3 **Fig. 59** illustrates the cryptoeconomics of HyperSphere e-commerce including the
4 minting and recycling of HyperCoins through HyperContract execution via HyperMetal
5 and HyperCoin pledges.

6 **Fig. 60** is a flow chart showing an economic amplification representation of a
7 value-added transaction, including positive and negative economic feedback mechanisms.

8 9 **DESCRIPTION OF THE INVENTION**

10 To overcome the deficiencies of the Internet in securing communication, ensuring
11 privacy, and supporting trusted business and e-commerce, we introduce a new,
12 innovative, and highly-advanced decentralized cybersecure ‘privacy’ network for global
13 e-commerce supporting real-time communication, data storage, cloud computing, cloud-
14 connected devices, and e-services– the HyperSphere. As described in the above-
15 referenced U.S. Provisional Application No. 62/696,160, filed July 10, 2018, entitled
16 "The HyperSphere– A Real-time Cybersecure Privacy Network with Embedded DyDAG
17 Dual Cryptocurrency for Global e-Commerce," the HyperSphere comprises an open-
18 source business platform and technology for enabling decentralized communication and
19 global e-commerce employing functions comprising:

- 20
21 • Decentralized hypersecure communications
22 • Network native certificate authority
23 • HyperSphere blockchain processing
24 • Network native cryptocurrency

25
26 These features are facilitated using inventive matter specific to each function being
27 performed. To understand the operation of hypersecure communication over a
28 decentralized network, we must first consider the functions executed by dedicated single-
29 function nodes in a network as described in the above-referenced U.S. Application No.
30 14/803,869, entitled "Secure Dynamic Communication Network and
31 Protocol," hereinafter referred to as an “SDNP network” or “SDNP cloud.”

Communication outside the SDNP cloud is described in the above-referenced U.S. Application No. 15/943,418 entitled "Methods and Apparatus for HyperSecure Last Mile Communication."

A. SDNP Routing Over Dedicated Nodes

FIG. 8 illustrates an SDNP network comprising edge client devices cell phone 95 and notebook computer 96 communicating over an SDNP cloud 100 comprising dedicated single-function nodes comprising SDNP media nodes 106, SDNP signal server nodes 107, and SDNP name server nodes 108 using high-speed physical links 110. Last mile communication between cell phone 95 and SDNP cloud 100 occurs over a mobile network including cell towers 103a and 103b and radio channels 112a and 112b. Cell phone 95 has access to SDNP cloud 100 via an installed SDNP application 101. Last mile communication between notebook 96 and SDNP cloud 100 occurs over a microwave last link 112c to a WiFi router 105, which in turn completes its access to SDNP cloud 100 over last mile wireline connections 111a and 111b via routers 104a and 104b. Notebook 96 has access to SDNP cloud 100 via installed SDNP application 102.

FIG. 9 illustrates the resources comprising the SDNP network including SDNP name server node 108, signaling server node 107, and media node 106 along with non-SDNP-enabled devices including Ethernet router 104 and WiFi router 105 and that employ TCP/IP routing tables, and cellular wireless network 103 that manages routing by the mobile network carrier.

SDNP name server nodes comprise dedicated single-function software designed to perform the network's name server role of matching phone numbers, account names, email addresses, or other identification to a device's IP address. The IP address stored in the SDNP name server can be dynamic and may comprise a NAT address, not a true Internet DNS recognized address. The SDNP name server files will be updated through a re-registration process whenever a device's dynamic IP address changes. As shown, SDNP name server node 108 is a virtual device, comprising SDNP name server software 133 installed in any lightweight server 130 and operating within the OS application VM environment of server 130.

As such, SDNP name server node 108 is not the same thing as server 130 hardware that hosts it. SDNP name server node 108 communicates with other nodes in the SDNP cloud using SDNP routing. Moreover, SDNP name server node 108 contains dynamic data not stored in the Internet's DNS servers. SDNP name server node 108 is limited in function to its lookup and address translation tasks and does not perform packet routing or carry media files (content) in SDNP cloud 100. SDNP name server node 108 has no access to the contents of any media packet traversing the SDNP network and no knowledge as to the encryption or concealment methods applied to media packets therein. SDNP name server node 108 is partially (but not purely) stateless, in that it retains no record of the inquiries it receives, the data it provides to signal server nodes, or to which signal server nodes. It is not purely stateless in the sense that it necessarily retains a translation table between SDNP network names or zip codes, phone numbers, and dynamic IP addresses. That said, the translation table is dynamic, meaning it contains constantly changing dynamic IP addresses and port numbers, updated each time they change or a device logs into or out of the SDNP cloud. The translation table is also stirred, i.e. randomly revised at regular intervals, changing the line item correspondence between SDNP addresses and/or zip codes and TCP/IP dynamic IP addresses.

SDNP signal server nodes (also referred to as signaling server nodes) comprise dedicated single-function software designed to perform the task of directing packet-routing and ensuring minimum propagation delay transport of packets. A SDNP signal server node issues routing commands to SDNP media nodes using lightweight command and control packets (C&C) either unilaterally or in conjunction with other SDNP signal server nodes working on the same communication's routing. The SDNP signal server nodes use addresses obtained from an SDNP name server node with no knowledge as to the identity or ownership of any host device. As shown, SDNP signal server node 107 is a virtual device, comprising SDNP signal server software 134 installed in any lightweight server 130 and operating within the OS application VM environment of server 130.

As such, SDNP signal server node 107 is not the same thing as server 130 hardware that hosts it. SDNP signal server node 107 communicates with other nodes in the SDNP cloud using SDNP routing. SDNP signal server node 107 has no idea as to the true owner of any device in the SDNP network and no knowledge as to the identity of a

1 caller, callee, or any other edge device. SDNP signal server node 107 has no knowledge
2 as to the content of any SDNP media packet or how the payload inside is concealed or
3 encrypted. SDNP signal server node 107 is purely stateless, meaning the node is launched
4 whenever it is needed; it collects the information it requires to perform routing (alone or
5 in conjunction with other signal server nodes); it distributes command and control (C&C)
6 packets to its selected media nodes, and then it forgets everything it just did. Stateless
7 operation of the signal server is the same thing as instantaneous amnesia – it cannot
8 remember any IP addresses it requested from the name server, it does not recall the last
9 instructions it gave or to what media nodes it sent its C&C packets.

10 SDNP media nodes comprise dedicated single-function software designed to
11 perform the task of carrying data packets of content (media) through the SDNP network.
12 Media packet payloads may contain fragments of audio, video, software, blockchain,
13 cryptocurrency, or software files. SDNP media nodes receive signal routing instructions
14 directing a packet from one SDNP media node to another, on a hop-by-hop basis with no
15 idea where a given packet originally came from or to where it is going. SDNP media
16 nodes receive instructions in C&C packets from signaling servers, which may include
17 numeric seeds and cryptographic keys used to dynamically change the concealment and
18 content of any packet traversing the SDNP network on a hop-by-hop basis.

19 As shown, SDNP media node 106 is a two-piece virtual device, comprising SDNP
20 media node software 135 installed in any high bandwidth server 131 operating within the
21 OS application VM environment of high bandwidth server 131, and comprising a second
22 component that includes DMZ software 136 installed in any non-network (air-gapped)
23 server 130a operating within the OS application VM environment of air-gapped server
24 130a. Air-gapped computer server 130a cannot be addressed directly over the Internet
25 and is separated from the media node and its host by a firewall or demilitarized zone
26 (DMZ) barrier 140 through which media node software 135 can only request certain
27 dynamically changing state-based information such as algorithms and concealment
28 methods.

29 As such, SDNP media node 106 is not the same thing as server 131 or air-gapped
30 DMZ server 130a or the hardware that hosts them. SDNP media node 106 communicates
31 with other nodes in the SDNP cloud using SDNP routing. SDNP media node 106 has no

1 idea as to the true owner of any device in the SDNP network and no knowledge as to the
2 identity of a caller, callee, or any other edge device. Except for the next hop and its last
3 hop, SDNP media node 106 has no knowledge as to the ultimate destination of a packet
4 or from whence it came. Even within media node 106, knowledge is limited. DMZ
5 software 136 and air-gapped server and 130a have no knowledge or access to the content
6 of the media packets being processed or carried by SDNP media node software 135 and
7 by high bandwidth server 131. Conversely, SDNP media node software 135 has no idea
8 as to how DMZ software 136 selects algorithms or chooses the file processing
9 instructions for concealing the payload contents of the data packets transported by media
10 node 106.

11 In the aggregate, media node 106 operates in a stateless manner, as it does not
12 retain the content of any media packet it carries, where the packet came from, where it
13 was sent, or how its contents were revised in executing the dynamic concealment process,
14 a complex dynamic editing of a packet's payload. DMZ software is quasi-stateless in that
15 it must remember a table of algorithms for concealment operations including content
16 mixing, splitting, scrambling, unscrambling, junk data insertions and deletions,
17 encryption and decryption, but it never possesses or sees the states, numeric seeds, and
18 cryptographic keys used in selecting and executing the algorithm. In cybersecurity, the
19 relationship across the DMZ barrier 140 is described as “zero knowledge.” So even if the
20 algorithm stored as a shared secret in the DMZ server software 136 is somehow
21 discovered (e.g. by a Mission Impossible type on-site invasion of a secure server room),
22 because the DMZ server software 136 is unaware of the state variables, numeric seeds,
23 and cryptographic keys used to process the media packet's payload, knowledge of the
24 algorithm doesn't help break the code. Moreover because the incoming packet contains
25 only a data fragment of scrambled encrypted information, there is no way to decrypt its
26 original contents.

27 Transport of media packets also may occur across non-SDNP enabled devices like
28 WiFi router 105, or Ethernet router 104, or over wireless network 103. This case is
29 especially likely in a “last mile communication” between a caller (edge device) and the
30 SDNP cloud. Because the SDNP packet follows the 7-OSI format, the routers see the IP
31 addresses as regular TCP/IP addresses for the Internet, the subnet, or the NAT address,

1 and route the packet the same as if it were a regular TCP/IP packet. Routing through
2 WiFi router 105 therefore employs its subnet or ISP provider's routing table 137 before
3 each reaches an SDNP node enabled device, possibly traversing no more than one or two
4 intermediary routers.

5 Similarly, routing through Ethernet router 104 employs the local ISP provider's
6 routing table 138 until each reaches an SDNP node enabled device. In the case of
7 transport of a SDNP packet over wireless network 103 or similarly over a cable or
8 satellite network, routing occurs using the network's proprietary network routing table
9 139 and custom packet format, e.g. 3G, 5G, satellite, or DOCSIS3 protocols. Network
10 address translator (NAT) 132 is required to convert the proprietary packet format back to
11 Ethernet compatible TCP/IP packet formatting.

12 In this manner, a SDNP media packet can be transported across a network of
13 Internet routers and SDNP nodes, with no need for custom hardware or dedicated
14 privately owned clouds, meaning an SDNP cloud can be installed on any TCP/IP
15 network. Transport of fragmented concealed packets over non-SDNP enable devices does
16 not risk communication security, but may degrade call quality-of-service QoS by
17 increasing a packet's propagation delay by routing the packet through longer delay paths
18 in the network.

19 **FIG. 10** illustrates an exemplary sequence for SDNP communication using
20 dedicated function SDNP nodes. In step 260 a client initiates a session via SDNP client
21 app 101 hosted on cell phone 95. The session may constitute any kind of communiqué,
22 including a VoIP phone call, a video chat, a video file, audio file, a file transfer, a
23 transactional process, group chat, conference call, etc. In step 261, the SDNP client app
24 101 contacts a default media node 106 to say hello to the network, i.e. to learn the
25 dynamic IP address of its nearest name server.

26 The call initiation, shown in **FIG. 11A**, typically involves routing of a query
27 packet over a last mile connection, in the example shown over radio link 112a to cell
28 tower 103a and then over wireline connection 120 to any SDNP media node 106. All
29 media nodes in a given region by default know the SDNP address of their nearest SDNP
30 name server nodes. They learn one or several SDNP name servers' addresses as part of

1 their automated registration process when they join the SDNP network. The media node
2 passes back this name server SDNP address to SDNP client app 101.

3 In step 262, shown in **FIG. 11B**, SDNP client app 101 then sends the query to
4 SDNP name server node 108 over wireline connection 113. SDNP name server node 108
5 then returns the current dynamic SDNP address or SDNP zip code of the device or phone
6 number the client is trying to contact, i.e. the callee. In an alternative but less secure
7 realization, the client app can obtain the SDNP address of a signal server node and have it
8 act as a proxy for the name server to obtain the callee's SDNP address.

9 In **FIG. 11C** corresponding to step 263, SDNP client app 101 then contacts SDNP
10 signal server node 107 sending the callee's SDNP address or SDNP zip code over
11 wireline connection 114. In step 264 shown in **FIG. 11D**, SDNP signal server node 107
12 contacts SDNP name server node 108 over wireline connection 121 to obtain the SDNP
13 addresses of all the media nodes it intends to execute routing over.

14 In step 265, shown in **FIG. 11E**, SDNP signal server node 107 dispatches
15 command and control (C&C) packets to the SDNP media nodes 106a-106f it selects to
16 carry the media packets, specifically using wireline connections 115a-115f to send
17 instructions to SDNP media nodes 106a-106f, respectively. It also passes state
18 information, numeric seeds, and cryptographic keys. In step 266, data transport of media
19 packets from caller SDNP app 101 to callee SDNP app 102 through the network of
20 SDNP media nodes 106a-106f commences.

21 SDNP's dynamic routing is exemplified by a changing path of each successive
22 data packet, as illustrated in **FIG. 11F** through **FIG. 11I**. Specifically, in **FIG. 11F** a first
23 data packet from client app 101 in cell phone 95 is carried to cell phone tower 103a over
24 radio link 112a, then by wireline 116 to the first media node 106a, acting as a gateway to
25 the meshed network of SDNP cloud 100. This gateway SDNP node then performs certain
26 state-based single-hop concealment operations on the content of the data packet and
27 forwards the packet to SDNP media node 106c, which repeats the process, sending the
28 packet to SDNP media node 106d, acting as a cloud gateway to last mile communication
29 with SDNP client 102 hosted by notebook 96, carried by router 104a, WiFi router 105
30 and radio link 112c.

As shown in **FIG. 11G**, a second data packet from client app 101 in cell phone 95 is carried to cell phone tower 103a over radio link 112a, then by wireline 117 to gateway media node 106b of SDNP cloud 100. Gateway node 106b then performs certain state-based single-hop concealment operations on the content of the data packet and forwards the packet to SDNP media node 106e, which repeats the process, sending the packet to SDNP media node 106d, acting as a cloud gateway to the same last mile communication with SDNP client 102 hosted by notebook 96, carried by router 104a, WiFi router 105 and radio link 112c.

As shown in **FIG. 11H**, a third data packet from client app 101 in cell phone 95 is carried to cell phone tower 103a over radio link 112a, then by wireline 118 to gateway media node 106c of SDNP cloud 100. Gateway node 106c then performs certain state-based single-hop concealment operations on the content of the data packet and forwards the packet to SDNP media node 106d, acting as a cloud gateway to a last mile communication with SDNP client 102 hosted by notebook 96, carried by router 104b, WiFi router 105 and radio link 112c. In this case, the last mile routing changed dynamically.

Finally in **FIG. 11I** a fourth data packet from client app 101 in cell phone 95 is carried to cell phone tower 103b over radio link 112b, then by wireline 119 to gateway media node 106a of SDNP cloud 100. Gateway node 106a then performs certain state-based single-hop concealment operations on the content of the data packet and forwards the packet to SDNP media node 106e, then to media node 106f, acting as a cloud gateway to the a last mile communication with SDNP client 102 hosted by notebook 96, carried by router 104a, then router 104b, WiFi router 105 and radio link 112c. In this case, both the first and last mile routing changed dynamically.

Referring again to **FIG. 10**, in step 267 a bidirectional dialog ensues between client apps 102 and 101 until then session ends in step 268. As described, SDNP cloud and last mile routing, packet security credentials, and packet content changed dynamically and continuously with no central control. The SDNP network as described constitutes a meshed network using *dedicated* resources, where certain servers are dedicated to hosting either SDNP name server node 108, SDNP signal server node 107, or one of SDNP media nodes 106a-106f functionality, but only one of them.

SDNP communication as described employs separate nodes for the name server, signal server, and media node functions, each hosted on separate dedicated devices. Network operation therefore requires the hardware hosts for all three classes of nodes remain perpetually available, i.e. *always on* and reachable. Global server clouds easily support perpetual availability by launching a new instance of installed software on an active server whenever a software host goes offline or the traffic on a single server exceeds certain levels.

In networks comprising uncertain resources such as *ad hoc*, peer-to-peer, and mobile networks, communication reliability is impacted by the fact that the requisite managerial functionality of the signal server node or the name server node may not be available all the time (even if the media nodes are sufficiently redundant to survive outages). In such cases, a fully decentralized realization of an SDNP cloud is needed to ensure that all functions are hosted and perpetually online.

Decentralization of SDNP network operation, however, requires the use of several inventive methods to maintain both functionality and hypersecurity.

B. Decentralized SDNP Hypersecure Communications

In order to retain both the functionality and hypersecurity of SDNP communications, a decentralized implementation of an SDNP network must adhere to the same governing principles as SDNP communication using dedicated function nodes hosted on fixed infrastructure, the principle of “zero knowledge” (or more accurately “incomplete knowledge”) specifically:

- No node with access to device or user identity should know data packet routing through the network
- No node with access to device or user identity should know the content of data packet communication.
- No node routing data packets should know the payload content of a data packet.
- No node routing data packets should know the owner identity of a data packet's payload.

- 1 • No node carrying data packets should know the owner identity of a packet's
- 2 payload.
- 3 • No node carrying data packets should have access to packet routing information
- 4 except for single hop information needed to identify incoming packets and to
- 5 direct outgoing packets to their next nodal destination.
- 6 • No node carrying data should know the selection process used to determine state-
- 7 based dynamic concealment of a packet.
- 8 • No node carrying data should ever contain a meaningful or useful amount of
- 9 payload data fragments.

11 A decentralized SDNP network has to perform hypersecure transport adhering to these
 12 criteria with no knowledge as to what resources are available to form the network. Such a
 13 requirement means any given HyperNode must be capable of performing any of the key
 14 roles, namely (i) the device identity management of a name server, (ii) the dispatcher
 15 function of a signal server, and (iii) hypersecure data transport of media nodes without
 16 comingling the data from one task with another.

18 **Decentralized d'SDNP Network Topology**

19 An example of a decentralized SDNP network (or d'SDNP cloud) is illustrated in
 20 **FIG. 12A**, comprising an ISP hosted cloud 200 of servers hosting a single-instance type
 21 of software to realize HyperNodes 206 labeled as [HN] communicating over high-speed
 22 wireline connections 210. For clarity's sake, in written text node names contain vertical-
 23 line brackets [XX] around the node ID (in figure graphics the brackets are excluded). All
 24 HyperNodes 206 comprise servers hosting executable code identical in functionality and
 25 capability. Accordingly, prior to commencing a HyperSphere communication session,
 26 HyperNodes 206x, 206y, 206z and 206a, all instances of a generic HyperNode 206
 27 behave identically as a cloud of homogeneous SDNP nodes.

28 As illustrated, cloud 200 connects to cell phone 95 via a mobile network
 29 comprising cell phone towers 103a and 103b, using cellular network radio links 112a and
 30 112b, respectively, to realize extra-cloud (last mile) communication. Cell phone 95 is
 31 able to access the HyperSphere cloud through installed d'SDNP software client

HyperSphere gateway node [HG] 201, software capable of performing the same connectivity as SDNP software 101 in dedicated function SDNP implementations. Similarly, last mile connection between cloud 200 and notebook 96 is facilitated via Ethernet router 104a, wireline connection 111a, WiFi router 105, and 802.11 microwave last link 112c.

In an alternative embodiment unique to decentralized SDNP communication, a hypersecure last mile connection between cloud 200 and notebook 95 can be facilitated by a SDNP-enabled HyperSphere router [HR] 204b, via wireline connection 211b to SDNP-enabled HyperSphere router [HR] 205, and 802.11 microwave last link 212c. Tablet 96 accesses HyperSphere cloud 210 through installed d'SDNP software, multifunction HyperSphere gateway node [HG] 202. The HyperSphere router software enables any component to operate with all the full features of a HyperNode even if the device is operating outside of all clouds performing last mile communication.

Given the fact that all the HyperNodes [HN] in cloud 200 are identical before communication commences; that the client's HyperSphere gateway [HG] software emulates many of the functions of HyperNodes; and that some routers are SDNP-enabled as HyperSphere routers [HR], then there is no meaning as to whether a group of HyperNodes are provided by a common ISP or network provider, hosted over a fixed network, common infrastructure, or network backbone. In the decentralized network of the HyperSphere, all nodes are of equal value. Accordingly, **FIG. 12B** illustrates the same HyperSphere network, represented graphically after removing the representation of cloud 200.

In essence, all devices hosting HyperNodes, including HyperSphere routers and HyperSphere gateways, essentially operate in an identical manner, forming a homogeneous network of nodes, hosted atop heterogeneous host devices. In this manner, the HyperSphere operates a uniform cloud completely agnostic as to its host platform devices – a key criterion necessary to realize a fully autonomous decentralized communication network. In other words in a decentralized network, there is no such thing as cloud and last mile communications – every HyperNode, HyperSphere router, and HyperSphere gateway are all equal partners, i.e. the HyperSphere's cloud is everywhere and omnipresent.

And unlike fixed infrastructure networks whose performance degrades as network traffic increases causing traffic jams and delays, the HyperSpheric cloud is dynamic and decentralized. The more HyperNodes that join the cloud, the more combinations exist and the better, not worse, the network's performance and quality-of-service (QoS) become.

HyperSphere Network Components

FIG. 12C illustrates the components of the HyperSphere network, including HyperNode HN 206, client HyperSphere gateway HG 201, and HyperSphere router HR 205 as well as non-SDNP enabled components including WiFi router 105, Ethernet router 104, and mobile wireless network 103. All three component nodes, HyperNode |HN| 206, client HyperSphere gateway |HG| 201, and HyperSphere router |HR| 205, are “metamorphic,” meaning that they can change their functionality based on the current demand for a particular network function.

In particular, metamorphic HyperNode |HN| 206 comprises HyperNode software installed on a host device 231, which may comprise a server, computer, smartphone, tablet, notebook, automobile, or IoT device. Unlike the previously disclosed dedicated function SDNP network where media nodes, signal server nodes, and name server nodes comprise separate executable application software, in d'SDNP code, every HyperNode is capable of performing any one of the three aforementioned functions with one proviso – a specific HyperNode can but only perform one of the three functions--task, name server, or authority node functions--at any one time. Mathematically, this feature is defined by the equation $|HN| = |T| \oplus |NS| \oplus |A|$ where $\oplus$ is the Boolean logic symbol for “exclusive OR” a function meaning one or the other but not both.

At the time an undifferentiated HyperNode receives an instruction to initiate a communication session, it must change (morph) into one of the three differentiated HyperSphere node types, either

- A task node |T| used to carry data packets (the same operations as the media node in dedicated function SDNP media nodes) and to execute dynamic concealment of datagram payloads. In dedicated function SDNP transport, an air-gapped DMZ server provides the instructions for dynamic concealment. In the disclosed

- 1 d'SDNP implementation, these instructions must be dispatched from a
2 decentralized DMZ server or d'DMZ (described later in this disclosure).
- 3 • An authority node |A| used to direct network traffic (the same function performed
4 by the signal server node in dedicated function SDNP networks) and to engage in
5 transaction validations in consensus based jury of peers (a blockchain function
6 described later). In dedicated function SDNP transport, a network propagation
7 delay table describing the time it takes for data to transit between node SDNP
8 nodal addresses need for real time data routing is stored in total (or in part) on
9 signal server nodes.
 - 10 • A name server node |NS| used to translate phone numbers and client IDs to
11 dynamic IP addresses, SDNP addresses, and SDNP zip codes. Unlike dedicated
12 function SDNP networks, the HyperSphere's name server database is
13 decentralized and disaggregated, comprising an ethereal multi-dimensional data
14 structure not physically located in any one server or storage device.

15
16 Because at the time a session is initiated, a HyperNode transforms, i.e. morphs
17 into one-of-three described node types, the HyperNodes are referred to as
18 “metamorphic,” the adjective form of metamorphosis meaning to "a change of the form
19 or nature of a thing into a completely different one." The metamorphic conversion of an
20 undifferentiated HyperNode into single-function HyperNode is analogous to an
21 undifferentiated stem cell transforming into a tissue-specific cell type. Unlike in
22 biological analogs, however, in the HyperSphere, metamorphosis is a reversible process.
23 After a differentiated HyperNode has completed its designated task, it reverts back into
24 its undifferentiated precursor awaiting its next job. In this sense the population of eligible
25 HyperNodes acting as network resources is inexhaustible and a given HyperNode's
26 network participation potential is indefatigable.

27 In summary, even though all HyperNodes |HN| start out as undifferentiated and
28 metamorphic, during a job the nodes morph into either a HyperSpheric name server |NS|,
29 authority node |A| performing dispatching and signal server functions, or task node |T|
30 performing media node functions, to complete a job then revert back into their
31 undifferentiated |HN| form after completing the job. During metamorphosis, the

undifferentiated node temporarily downloads requisite information from a diffuse, i.e. disaggregated or redundant, database layer (described later), performs its defined job and then reverts back to undifferentiated [HN] state, forgetting all the information it downloaded and everything it did. Operational amnesia, referred to as stateless operation, means a HyperNode holds knowhow only temporarily, and never possesses the information for node naming, packet routing, or payload content at the same time. Stateless operation, thereby ensures a node that can perform any of the [NS], [A], and [T] functions but can never do more than one of them at a time.

And although a HyperNode employs the resources of its host processor and memory, it does not share any information regarding the data it processes or tasks it performs with its host. Such a method known as sandboxing, a method that protects the host OS from the applications it runs. HyperNode software, however, is *symmetrically sandboxed*, meaning the HyperNode cannot interact with or obtain information from its device host. Conversely, a device hosting a HyperNode cannot interact with (or even comprehend) network traffic carried by the HyperNode. The HyperNode operates statelessly, meaning it leaves no record as to its operation in the memory of its host device. This means that any device can host a HyperNode without concern about the node and the device interfering with each other than the device deciding how much computational power (compute cycles) to allocate to the HyperNode.

Because of this symmetric sandboxing, the HyperNode cannot operate as a user interface for supporting applications or performing client specific functions. A special software interface is required for carrying data across the device application boundary. Specifically, for an application or client to access the HyperSpheric cloud, specialized application software 232 is installed into the device, for example into cell phone 95 or (any other edge device), to facilitate HyperSphere gateway HG 201. According to Wikipedia, an edge device is "a device providing an entry point into enterprise or service provider core networks. Examples include routers, routing switches, integrated access devices (IADs), multiplexers, and a variety of metropolitan area network (MAN) and wide area network (WAN) access devices." In HyperSphere parlance, an edge device is any device hosting HyperSphere gateway software. Examples include personal

1 computers, games, tablets, smartphones, IoT devices, automobiles, and other user
2 controlled network connected components.

3 Unlike a HyperNode, which is generic software designed to operate atop major
4 operating systems such as UNIX, Linux, MacOS, Windows, iOS, and Android,
5 HyperSphere gateway functionality necessarily interacts with the application it enables,
6 e.g. a bank login portal, a POS terminal, a factory control system, an automobile's
7 infotainment system, home security systems, home IoT clouds, etc. To facilitate this
8 integrated link, a HyperSphere-specific application programming interface (API) and
9 source development kit (SDK), along with special premade interfaces and software
10 utilities (library) are important engineering tools. Specifically, as an element of the
11 HyperSphere's Application Programming Interface (HAPI), the HyperSphere gateway
12 HG 201 provides a defined link, language, and protocol for a user application to contact
13 and communicate with the HyperSphere decentralized network and with other
14 HyperNodes. User applications are developed using the HyperSphere Software
15 Development Kit (HSDK).

16 While in most cases HyperSphere gateway HG 201 operates as a dedicated
17 interface and API to the HyperSphere, in some circumstances the gateway is
18 metamorphic. One metamorphic application of a HyperSphere gateway is in forming *ad*
19 *hoc* peer-to-peer networks (described later). In other words, a HyperSphere gateway [HG]
20 operates as a HyperNode which can in a limited manner, exchange certain types of data
21 between the HyperSpheric cloud and the device. HyperNode gateways, however, do not
22 operate as autonomous code but instead are integrated into a merchant's or service
23 provider's application software.

24 Another version of HyperNode software is the HyperSphere router [HR].
25 Although SDNP communication normally operates over the Network Layer-3 to the
26 Applications Layer-7, [HR] router software includes provisions for optionally managing
27 security on the PHY Layer-1 and MAC Layer-2, features valuable in routers, in RAID
28 storage, and in IoT devices. HyperSphere router 205 comprises conventional router
29 hardware such as Ethernet router 104 combined with HyperSphere dynamic router
30 software 233 to realize SDNP-enabled wireline router 234, defined herein as an
31 "Ethyernet" router. Alternatively, WiFi router 105, combined with HyperSphere dynamic

router software 233 can be used to realize SDNP-enabled WiFi wireless router 235, defined herein as a “HyFi” router.

HyperSphere router software offers special advantage in Last Mile communications outside of fixed infrastructure networks or high-speed clouds, because it manages security and user validation prior to establishing or allowing Network Layer-3 communications. Such a feature is valuable in thwarting denial of service (DoS) attacks because the |HR| node completely ignores nodes it can't recognize. HyperSphere routers offer numerous advantages:

- HyperSphere routers direct HyperSphere datagrams in accordance with SDNP routing instructions received from a HyperSphere authority node (performing the signal server function). A router's static routing table does not determine packet routing.
- By remotely controlling routing by an authority node, HyperSphere datagrams cannot be diverted or hijacked even in last mile communications.
- HyperSphere routers report hop delays to the network, allowing a HyperSphere authority node (the signal server function) to decide the lowest propagation delay path for last mile communication, a feature non-SDNP routers cannot support.
- HyperSphere routers are support SDNP dynamic concealment processing and interpret dynamic security credentials.
- HyperSphere routers support a higher refresh rate on dynamic security parameters in last link communication between the HyperSphere router and a client device's HyperSphere gateway, improving dynamic security.
- HyperSphere routers support shared secrets in last link communication between the HyperSphere router and a client device's HyperSphere gateway, improving dynamic concealment.

In **FIG. 12A**, for example, hypersecure communication via Ethyrnet HyperSphere router 204b and HyFi HyperSphere router 205 offers a higher performance last mile connection than data transport over conventional routers. In many cases, however, last mile communication between a caller (edge device) and the SDNP cloud invariably also

1 transports media packets across non-SDNP enabled devices, such as WiFi router 105 and
2 Ethernet router 104, or over wireless network 103.

3 Because the SDNP packet follows the 7-OSI format, non-SDNP routers interpret
4 the IP addresses as regular TCP/IP addresses for the Internet, the subnet, or the NAT
5 address, and route the packet the same as if it were a regular TCP/IP packet, never
6 realizing the packet is HyperSpheric. Referring again to **FIG. 9**, traffic through WiFi
7 router 105 therefore employs routing table 137 rather than dispatcher based routing.
8 Similarly, Ethernet router 104 employs the local ISP provider's routing table 138 until
9 each packet reaches an SDNP node-enabled device. In wireline communication, an
10 SDNP packet will likely traverse no more than one or two intermediary routers before
11 each reaches a HyperNode. In the case of transport of a SDNP packet over wireless
12 network 103 (or similarly over a cable or satellite network), routing occurs using the
13 carrier's proprietary network routing table 139 and custom packet format (e.g. 3G, 4G,
14 5G, satellite, or DOCSIS3 protocols). Network address translator (NAT) 132 is required
15 to convert the proprietary packet format back to Ethernet compatible TCP/IP packet
16 formatting for wireline connectivity.

17 In this manner a SDNP media packet can be transported across a network of
18 Internet routers and SDNP nodes, with no need for custom hardware or dedicated
19 privately-owned clouds, meaning that an SDNP cloud can be installed on any TCP/IP
20 network. Transport of fragmented concealed packets over non-SDNP enable devices does
21 not risk communication security, but may degrade call quality-of-service QoS by
22 unpredictably increasing a packet's propagation delay by routing the packet through
23 longer delay paths in the network.

24 25 **Decentralized SDNP Network Operation**

26 Data packet routing in a decentralized SDNP network employs the same
27 anonymous fragmented data transport over a meshed network using dynamic security
28 provisions as its predecessor comprising fixed infrastructure over dedicated SDNP nodes,
29 except that in a decentralized meshed network, the nodes available to carry data
30 constantly change in a fully autonomous manner with no network operator supervision.
31 **FIGs. 12D and 12E** describe the operation of an exemplary decentralized SDNP network,

also referred to as a 'd'SDNP network'. The corresponding packet routing across the decentralized meshed network is shown in **FIGs. 13A** through **13H**.

In step 270 of **FIG. 12D**, HyperSphere gateway |HG| 201a hosted on cell phone 95 opens a session with the d'SDNP network. In step 271a HyperSphere client gateway |HG| 201 contacts HyperNode |HN| 206x to fetch the call recipient's SDNP address. In step 271b HyperNode |HN| 206x morphs into HyperSphere name-server node |NS| 236a. As shown in **FIG. 13A**, HyperNode |HN| 206x then establishes a connection with HyperSphere client gateway |HG| 201a over radio link 112a to cell tower 103a, and over wireline connection 250 to HyperSphere name-server node |NS| 236a. In step 272a, HyperSphere name server node NS 236a passes the recipient's SDNP address to HyperSphere client gateway |HG| 201. In step 272b, HyperSphere name server node |NS| 236a reverts back into an undifferentiated HyperNode |HN| 206x.

In step 273a HyperSphere client gateway |HG| 201 passes the recipient SDNP address to undifferentiated HyperNode 206z, which in step 273b morphs into authority node |A| 237, a process shown in **FIG. 13B**, where the data is transported over radio link 112a to cell tower 103a, and over wireline connection 251 to authority node |A| 237.

In step 274a, authority node |A| 237 contacts undifferentiated HyperNode 206y to fetch the SDNP addresses of task node addresses, whereby in step 274b HyperNode 206y morphs into HyperSphere name server node |NS| 236b. In step 275 shown in **FIG. 13C**, HyperSphere name server node |NS| 236b delivers over wireline connection 252 the requested address table to HyperSphere authority node |A| 237.

In **FIG. 13D** corresponding to step 276a in **FIG. 12E**, HyperSphere authority node |A| 237 sends routing instruction to select undifferentiated HyperNodes 206 over wireline connections 253a to 253h, thereby in step 276b converting them into HyperSphere task nodes |T| 238a to 238h, respectively. Note that prior to differentiation, HyperSphere task nodes 238d and 238e comprised HyperSphere routers |HR| 204b and 205, not nodes hosted by ISP cloud 200.

In **FIG. 12E**, corresponding to step 277a, a first SDNP data packet is transported from HyperSphere client gateway |HG| 201 to cell tower 103a over radio link 112a, then over wireline 254 to task nodes |T| 238a, 238c, 238f, followed by Ethernet router 104a, WiFi router 105, and by WiFi microwave link 112c to HyperSphere gateway 202. In

FIG. 13F, a second SDNP data packet is transported from HyperSphere client gateway |HG| 201 to cell tower 103a over radio link 112a, then over wireline 255 to task nodes |T| 238b, 238g, 238f, followed by Ethernet router 104a, WiFi router 105, and by WiFi microwave link 112c to HyperSphere gateway 202.

In **FIG. 13G**, a third SDNP data packet is transported from HyperSphere client gateway |HG| 201 to cell tower 103a over radio link 112a, then over wireline 256 to task nodes |T| 238c, 238f, 238f, followed by HyperSphere Ethyrnet router task node 238d, then HyperSphere HyFi router 238e, and finally by hypersecure microwave link 212d to HyperSphere gateway 202. In **FIG. 13H**, a fourth SDNP data packet is transported from HyperSphere client gateway |HG| 201 to cell tower 103b over radio link 112b, then over wireline 257 to task nodes |T| 238a, 238g, 238h, followed by Ethernet router 104a, HyperSphere Ethyrnet router task node 238d, HyperSphere HyFi router 238e, and by hypersecure microwave link 212d to HyperSphere gateway 202.

Referring again to **FIG. 12E**, the sequence of transfers of fragmented data describes communication from gate HyperSphere HG 201 to 202 via collective task nodes 238. After delivery of fragmented data packets in step 277b, task nodes 238 revert to undifferentiated HyperNodes |HN|. The process 278 then repeats for a two-way dialog between gateway devices 201 and 202 until the session ends at step 279.

Metamorphic HyperNode Operation

In the HyperSphere, two means exist to implement metamorphic HyperNode operation. In one version three separate executable code files comprising |NS|, |A|, and |T| functionality are loaded onto the same server or host device. Each single function node uploads the MAC address of its host device. When a HyperNode is solicited to execute a task, it first checks with the other single-function HyperNodes having the same MAC address to see if they are running a job for the same client. If they are, the HyperNode will deny accepting the new job. For example, if task HyperNode |T| hosted on MAC address {00:A0:C9:14:C8:29} is carrying data packets for client gateway node 201, then a request for |A| node having the same MAC address to perform routing for the same client device, i.e. client gateway node 201, will be denied.

1 In a more efficient manner, the exclusivity function for a metamorphic
2 HyperNode can be integrated into a single piece of executable code including the selector
3 function. This inventive method is illustrated in **FIG. 14**, where HyperNode [HN] 206
4 realized by HyperNode software 231 installed on server 230 contains the executable code
5 to operate as either a [T], [A], or [NS] node, where selector 290 is employed to limit the
6 HyperNode's operation to only one of the three functions at any given time, i.e. selector
7 290 facilitates the exclusivity function.

8 In one embodiment, HyperNode 206 receives a job order in the form of a digital
9 file, referred to as a HyperContract 280, including job description 291, numeric seed 281,
10 and cryptographic key 282. The job description provides the necessary information for
11 selector 290 to determine whether the HyperNode 206 should be a task node, an authority
12 node, or a name server node. In the prior routing example, when metamorphic
13 HyperNode 206x receives instruction to perform name server functions at the request of
14 HyperSphere client gateway [HG] 201, undifferentiated [HN] 206x downloads data from
15 diffuse data cloud 283a and morphs into [NS] node 236a.

16 Similarly, when metamorphic HyperNode 206z receives instruction to perform
17 authority node functions at the request of HyperSphere client gateway [HG] 201,
18 undifferentiated [HN] 206z downloads data from diffuse data cloud 283b and morphs into
19 [A] node 237. The data downloaded from diffuse data cloud 283a to create [A] node 237 is
20 not the same as the downloaded data used to produce [NS] node 236a. Accordingly, the
21 name server and authority nodes are differentiated, accessing different information from a
22 diffuse cloud on a need-to-know basis.

23 Later, when metamorphic HyperNode 206y receives instruction to perform name
24 server functions at the request of authority node [A] 237, [HN] 206y downloads data from
25 diffuse data cloud 283a and morphs into [NS] node 236b. The data downloaded from
26 diffuse data cloud 283a to create [NS] node 236b is not the same as the data used to
27 produce [HN] 206x. Thus the two name server instances are differentiated, accessing only
28 the information it requires on a need-to-know basis.

29 Finally, when metamorphic HyperNode 206a receives instruction to perform task
30 node routing instructions at the request of authority [A] 207, undifferentiated [HN] 206a
31 downloads data from diffuse data cloud 283c and morphs into [T] node 238. The data

downloaded from diffuse data cloud 283c to create |T| node 238 is not the same as the downloaded data used to produce |A| node 237 or |NS| nodes 236a or 236b. Thus the task nodes, authority nodes, and names server nodes are all differentiated and mutually exclusive, accessing different information from a diffuse cloud purely on a need-to-know basis.

A key innovation in the HyperSphere is its ability to share information globally without dedicated storage devices using a disaggregated “diffuse” data cloud. Disaggregated data storage is data fragmented in small units (like jigsaw puzzle pieces) and stored in local storage devices diffused across the HyperSphere's decentralized cloud. Operation of the HyperSphere's disaggregated data storage, including writing, reading, searching and refreshing, is described later in this application. Suffice it to say, data stored in the disaggregated diffuse data cloud of the HyperSphere is data that can only be accessed by authorized HyperNodes on a need-to-know basis.

FIG. 15A exemplifies the role of diffuse cloud storage in metamorphic HyperNode operation. In the leftmost column, HyperSphere client gateway |HG| 201 requests name server information from metamorphic HyperNode |HN| 206x. A numeric seed 281 from HyperContract 280 authorizing access is passed to |HN| 206x. In one embodiment the numeric seed defines the creation state of the stored data, i.e. the state variable at the time the data was stored. The access request may also include state 284 used to verify the request is timely, e.g., to confirm the authorization period hasn't expired. The state may comprise time, location, security zone, or other dynamically changing parameters.

In the next step, shown in the center column, undifferentiated |HN| 206x establishes a secure channel with the name server node diffuse data cloud 283a, using encryption key 282 or using a digital CA certificate (described later) to authenticate the request. In the rightmost column of **FIG. 15A**, the requested data is located, aggregated into a compact file, and passed from diffuse data cloud 283a to |HN| 206x which immediately morphs into a differentiated name server HyperNode |NS| 236a. Name server HyperNode |NS| 236a then passes the requested NS data 285, in this case caller ID information, to the requesting HyperSphere client gateway |HG| 201.

As shown in **FIG. 15B**, diffuse data recall involves contact information of the edge device, e.g., the phone number 285a the caller wishes to call (callee), which is used to extract data from the name server diffuse data cloud layer 283a to extract callee ID info contained in name server |NS| file 285b including the device's SDNP address, SDNP Zip, and the dynamic IP address of the callee at that moment. The file may also include custom or application specific information labeled as “VIP data” used to determine routing priority, cost, routing redundancy, etc.

In the leftmost column of **FIG. 16A**, HyperSphere client gateway |HG| 201 requests informs metamorphic HyperNode |HN| 206z of its intent to place a call to called ID specified in |NS| data 285b. A numeric seed 281 from HyperContract 280 authorizing access is passed to |HN| 206z. In one embodiment the numeric seed defines the creation state of the stored data, i.e. the state variable at the time the data was stored. The access request may also include state 284 used to verify the request is timely, e.g. to confirm the authorization period hasn't expired. The state may comprise time, location, security zone, or other dynamically changing parameters.

In the next step, shown in the center column, undifferentiated |HN| 206z establishes a secure channel with the authority node diffuse data cloud 283b using encryption key 282 or using a digital CA certificate (described later) to authenticate the request. In the rightmost column of **FIG. 16A**, the requested data is located, aggregated into a compact |A| data file 286a containing network hop times, and passed from diffuse data cloud 283b to |HN| 206z which immediately morphs into a differentiated HyperNode comprising authority node |A| 237.

As shown in **FIG. 16B**, diffuse data recall involves caller ID information |NS| data 285b used to access data from the authority node diffuse data cloud layer 283b to extract network hop time info contained in name server |A| file 286a. The file contains a list of potentially relevant node-to-node propagation delays between two SDNP addresses and the last recorded transport time for the particular hop, specifically hops in the general path between the caller and the callee. The table contains an unranked list of the propagation delay between any two SDNP addresses where the propagation delay in milliseconds (ms) is represented in hexadecimal format. For example, the delay 2B hex is

equivalent to 43 ms. Note the SDNP address may be represented as IPv6 as shown or alternatively using IPv4 format.

In **FIG. 16C**, an algorithm used to rank and choose routing option 309 is employed to revise the edit 286a into abridged table 286b. The selection and ranking process employs an algorithm described previously in the abovereferenced U.S. Patent No. 9,998,434, entitled "Secure Dynamic Communication Network and Protocol." The algorithm will not be repeated here. The output of the routing selection process results in a list 286c of SDNP addresses of task nodes potentially useful for packet routing.

Shown in the leftmost column of **FIG. 17A**, table 286c is forwarded from |A| node 237 to metamorphic HyperNode 206y, which in conjunction with numeric seed 281 and state 284 contacting (for a second time in this session) name server node diffuse data cloud 283a, which as shown in the center column establishes a secure link with HyperNode 206y using cryptographic key 282. In response, as illustrated in the rightmost column, name server node diffuse data cloud 283a downloads data to HyperNode |HN| 206y, which instantly morphs into name server node |NS| 236b passing |NS| data file 287a containing task node ID table to authority node |A| 237.

As shown in **FIG. 17B**, diffuse data recall involves rank sorted task node data 286c, used to access data from the name server diffuse data cloud layer 283a to extract task node ID info contained in name server |NS| file 287a. The file contains a list which includes a translation table for each requested SDNP address and its corresponding SDNP Zip code and its dynamic IP address at that moment.

Shown in the leftmost column of **FIG. 18A**, table 287a is then forwarded from |A| node 237 to metamorphic HyperNode 206w, which in conjunction with numeric seed 281 and state 284 contacts task node diffuse data cloud 283c, which as shown in the center column establishes a secure link with HyperNode 206w using cryptographic key 282. In response, as illustrated in the rightmost column, name server node diffuse data cloud 283c downloads shared secrets in |T| data file 287b to HyperNode |HN| 206w, which instantly morphs into task node 238a ready to carry data.

As shown in **FIG. 18B**, diffuse data recall involves task node IP data 287a used to access data from the task node diffuse data cloud layer 283c to extract shared secrets contained in task node |T| file 287b. The file contains a list of decentralized DMZ

1 algorithms, i.e. d'DMZ dynamic concealment algorithms, including those algorithms
2 299a required for processing incoming data packets 321a and those algorithms 299b
3 employed to process outgoing data packets 321b as represented schematically as task
4 node 320 in **FIG. 18C**. Functional operation of a task node performing "media node"
5 operations is described previously in the abovereferenced U.S. Patent No. 9,998,434.

6 7 **Diffuse Data Cloud**

8 As illustrated in **FIG. 19**, a diffuse data cloud 283 stored as disaggregated data is
9 subdivided into three types of data or sub-clouds, namely name server cloud 283a,
10 authority node cloud 283b, and task node cloud 283c. Upon differentiating into a specific
11 node type, metamorphic HyperNode 206 is confined to interact only with its
12 corresponding diffuse data cloud layer on a need-to-know basis. For example,
13 differentiated into a name server node, [NS] HyperNode 236 can only access data from
14 diffuse cloud 283 related to name server cloud 283a but not from authority node cloud
15 283b or task node cloud 283c. Aside from client HyperSphere gateway [HG] gateway
16 requests, name server [NS] node can only communicate with authority node 237 but not
17 with task node 238.

18 When differentiated into an authority node, [A] HyperNode 237 can only access
19 data from diffuse cloud 283 related to authority node cloud 283b but not from name
20 server cloud 283a or task node cloud 283c. Aside from client HyperSphere gateway [HG]
21 gateway requests, authority [A] node 237 is able to request information from name server
22 HyperNode 236 and communicate instructions in command and control (C&C) packages
23 to task HyperNodes 238.

24 When differentiated into task node, [T] HyperNode 238 can only access data from
25 diffuse cloud 283 related to task node cloud 283c but not from name server cloud 283a or
26 authority node cloud 283b. Aside from client HyperSphere gateway [HG] gateway
27 requests, task [T] node 238 is able to receive command and control (C&C) instructions
28 from authority node HyperNodes 237 but is unable to communicate directly with name
29 server [NS] HyperNode 236.

30 Several possible methods to search and recall the diffuse cloud data are illustrated
31 in **FIG. 20** including tiered name servers 300 shown in the leftmost column, RAID

1 redundant memory 301 shown in the center column and disaggregated storage 302 shown
2 in the rightmost column. In tiered name servers, data fragments stored in servers 307 are
3 first sorted and aggregated into larger files in servers 306, and finally merged into one file
4 in server 306. The files comprise contain unencrypted portions used for searching and
5 sorting, and concealed or encrypted payloads containing user ID information. In response
6 to a request from metamorphic HyperNode |HN| 206x, name server 305 passes an
7 encrypted or concealed file to decryption process 287 that in combination with decryption
8 key 282 unlocks the data loading it into |HN| 206x and triggering the node to morph into
9 a differentiated HyperNode, in this case into name server |NS| 236a.

10 In RAID redundant memory 301, data fragments stored in data drives 308, are
11 defragmented 310 and loaded into server 305 which in turn passes the concealed data to
12 decryption process 287 that in combination with decryption key 282 unlocks the data,
13 loading it into |HN| 206x and triggering the node to morph into a differentiated
14 HyperNode, in this case into name server |NS| 236a.

15 Alternatively in disaggregated cloud 302, data fragments stored in cloud servers
16 307 are defragmented 310 and loaded into server 305 which in turn passes the concealed
17 data to decryption process 287 that in combination with decryption key 282 unlocks the
18 data, loading it into |HN| 206x and triggering the node to morph into a differentiated
19 HyperNode, in this case into name server |NS| 236a.

20 As shown in **FIG. 21**, diffuse data storage of fragmented payload packets is
21 graphically represented using the aforementioned puzzle piece metaphor. In fragmented
22 storage, original media content comprising unprocessed data (such as sound, video,
23 pictures, files, programs, etc.) contains elements having definitive interconnections and
24 interrelationships. Original file content 325, if disturbed even slightly, destroys the
25 functionality and may render the file unrecoverable. In the file storage process, original
26 file content 325 is algorithmically fragmented in step 326 into pieces in accordance with
27 state-based digital instructions or HyperContract 260. The disaggregated data elements
28 327 are then divided into separated components 327a (or sub-files) and stored in separate
29 file storage media 308, collectively representing diffuse data storage 301. In the storage
30 process, a cryptographic key 329 is generated and sent to the data owner or
31 HyperContract author in order to be able to recall the data.

FIG. 22 illustrates the redundant file storage of name server data. For example a parsed file of updated name server data 330a, 330b, 330c is written into a number of storage servers 331, 332, and 333 respectively comprising name server nodes [NS₁₃], [NS₉₂], and [NS₄₆], collectively performing diffuse name server cloud storage 302. As shown, file parsed file 1 comprising data 330a is redundantly stored on both [NS₁₃] and [NS₄₆] and others (not shown); file parsed file 2 comprising data 330b is redundantly stored on both [NS₉₂] and [NS₄₆] and others; and file parsed file 3 comprising data 330c is redundantly stored on both [NS₁₃] and [NS₉₂] and others.

The storage of file 1 and file 3 in [NS₁₃], file 2 and file 3 in [NS₉₂], and file 1 and file 2 in [NS₄₆] automatically generates file storage access link keys 319a, 319b, and 319c respectively, all of which are stored redundantly on name server nodes [NS₂₁] and [NS₇₇] hosted on servers 334a and 334b, respectively, and others (not shown). If any [NS] storage node goes offline, the data is cloned onto another name server node. In one embodiment, each [NS] node issues and maintains a dynamic HyperContract listing backup servers, currently online. If a server goes offline, the HyperContract automatically clones the data to another server, which in turn issues a new backup HyperContract, perpetually *ad infinitum*.

The process of updating the name server database stored by the name server diffuse cloud occurs every time a data storage resource comes online or disappears. As shown in **FIG. 23**, the registration process occurs every time there is a change in resources. For example, when a new HyperNode [HN] 206p joins the HyperSphere network, the info informs a name server node [NS] 207m to register the node. The initiation of the node registration can be done directly or in an alternative embodiment using an authority node as part of the 'hello' process. After receiving the dynamic IP address of the new node and assigning a new SDNP address and SDNP Zip code, the name server diffuse data cloud downloads a relevant portion of the database into [NS] 207n hosted on storage server 333 where parsed file 330a whereby new content 330p is appended onto the file. The file is then processed in step 328 for storage by [NS] 207n by reparsing the file into multiple pieces such as file 330r and 330s, then uploaded to name server node diffuse data cloud 383a. At the same time, a revised link key 329b is forwarded to name server [NS] 207p and used for name server operational requests.

1 Meanwhile, HyperNode 206p commences a regular network broadcast, i.e. a ping,
2 to ensure connectivity with [NS] 207m or any other node in the network. If the ping is
3 missed several times, the registration process is repeated, removing node [HN] 206p from
4 the name server list of active nodes.

6 **Decentralized Meshed Network Security Provisions**

7 The security of data communication in the decentralized SDNP network involves
8 numerous innovative methods. These methods, while similar to those disclosed in the
9 above-referenced U.S. Patent No. 9,998,434, are adapted for decentralized operation with
10 no dedicated function network nodes, central authority or network operator, namely:

- 12 • *Division of labor* means no one HyperNode has access to all information needed
13 to extract a meaningful payload, whereby [NS] nodes know information regarding
14 HyperNode identity but have zero knowledge about any payload or the routing of
15 any d'SDNP data packets; [A] nodes control packet routing but have zero
16 knowledge as to the identity of any HyperNode users or the content of any
17 payload; and [T] nodes carry content but have no knowledge as to the identity of
18 any HyperNode users or the ultimate routing of any packet. HyperNodes only
19 know the next hop destination, but have zero knowledge as to the identity of
20 any edge devices or callers.
- 21 • *DMZ (air gapped) shared secrets* means [T] nodes carrying data packets have zero
22 knowledge as to how the concealment algorithms are chosen to encrypt or
23 obfuscate payload content. Concealment algorithms and ciphers are zero day
24 components installed in diffuse d'DMZ clouds preinstalled and existing prior to
25 any communication session.
- 26 • *Metamorphic HyperNodes* eliminate the use of dedicated function nodes, thereby
27 preventing the opportunity to develop function specific attacks, because the role
28 of a specific HyperNode constantly changes among [A], [T] and [NS] functions.
- 29 • *A dynamic HyperSphere name server [NS]*, a database not affiliated with the
30 Internet's DNS name server, contains the HyperSphere's current list of every
31 SDNP node registered on the network. The node list is stored in a [NS] diffuse

cloud and is not available to task nodes transporting the data packets (thereby maintaining a separation of roles to prevent phishing).

- *Fragmented data transport* involves parsing a file into small snippets (sub-packets), and transmitting the snippets in data packets over a meshed network whereby a hacker intercepting a single packet cannot extract useful content from the packet's payload.
- *Meshed routing* dynamically changes data paths based on timely knowledge of network propagation delays— information stored and updated in a diffuse $|A|$ data cloud not accessible except by the authority node responsible for a session's routing.
- *Dispatcher-based packet routing* where an $|A|$ node route data packets by informing each task node the information to execute only a single hop to the next destination in the network.
- *Anonymous data packets* where each SDNP datagram shows only the IP source address of a single hop (i.e. the node sending the packet) and the IP address of the packet's next destination without ever revealing the original source or the final destination of the data packet, the name comprising the dynamic IP address of the SDNP node retrieved from the $|NS|$ diffuse cloud. Data packet anonymity renders the collection of metadata meaningless in profiling a cyber target or discerning patterned behavior to develop attack stratagems.
- *Dynamic security credentials and algorithms* change the concealment methods (including scrambling, encryption, junking, splitting and their anti-functions) using state-based shared secrets securely stored in a $|T|$ diffuse cloud accessible to participating task nodes on a need-to-know basis.
- *Stateless node operation* where each differentiated HyperNode reverts to its undifferentiated metamorphic state after completing its designated job, upon which it forgets all information regarding its last action (or the data related to performing such actions).

Combining metamorphic HyperNodes with meshed dynamic routing executing authority node-issued command and control packets, packet routing within the

HyperSphere occurs unpredictably without any central authority or network operator. Decentralized SDNP communication using autonomous *meshed routing* as disclosed herein enhances network security because no operator is involved in directing data traffic or managing the network. An example of decentralized SDNP routing of three successive packets sent from client |HG| 350a to |HG| 350b is illustrated in **FIG. 24**. In the left column, data traverses through secure connection 352a to task node |T| 351d, then over secure connection 352b to client |HG| 350b. The authority node decides packet routing based on propagation delay data downloaded from the authority node diffuse cloud,

A second feature of decentralized SDNP communications is the use of *dynamic security and concealment techniques*. For example, the security credentials and concealment algorithms of secure connections 352d and 352e differ, including employing unique single-hop numeric seeds, cryptographic keys, and dynamic state conditions. As a result, should the virtually impossible task of breaking encryption and defeating dynamic security in 100 ms— the time it takes data to traverse secure connection 352d—be achieved, the entire process would have to be reprised again 100 ms later, this time as data is transported over secure connection 352e using entirely different security methods and keys.

Dynamic security in decentralized SDNP communication is further elaborated in **FIG. 25**, where the security method 352d used in transporting a data packet from node |T₁| to node |T₂| comprises a data packet comprising a MAC Layer-2 addresses 390a consistent with the host device hardware; source and destination routing addresses 391a and 391b comprising the dynamic IP addresses corresponding to the SDNP addresses of HyperNode |T₁| and |T₂| supplied by name server diffuse cloud |NS|; transport Layer-4 criteria 393a (shown by example as TCP), and session Layer-5 data 394a used to validate a session confirming both parties identity through a CA digital certificate with AAA validation comprising a sequence of authentication, authorization, and administration. A successful session validation is needed to commence processing the remaining portions of the data packet. SDNP unique security credentials are carried with the datagram in the presentation field 395a and in the application data field 398a.

Presentation Layer-6 data field 395a, comprising a state variable like time t₁, zone Y security methods, plus a cryptographic key 1 and a numeric seed Y1 are then conjoined

1 in process 396a with other seed or key data (delivered prior to data routing by command
2 and control (C&C) packet from the authority node responsible for routing the session).
3 This information is used in conjunction with security algorithms supplied by task node
4 diffuse cloud $|T|$ to execute concealment algorithms 397a to de-process (recover)
5 incoming file data and convert it into newly concealed data 389a.

6 The entire process is repeated by task HyperNode $|T_2|$ 351b as data passes from
7 the single-hop security method 352d to method 352e. Accordingly, MAC address 390a is
8 updated to 390b with the device hosting node $|T_2|$ becoming the source MAC address and
9 the device hosting node $|T_3|$ the destination address. The datagram source IP address is
10 changed from the dynamic IP address of SDNP node $|T_1|$ 391a to that of dynamic IP
11 address of SDNP node $|T_2|$ 398b, namely DIP {SDNP $|T_2|$ }. Similarly, the datagram's
12 destination IP address is changed from the dynamic IP address of SDNP node $|T_2|$ 391b to
13 that of dynamic IP address of SDNP node $|T_3|$ or 391c, namely DIP {SDNP $|T_3|$ }. The L4
14 transport data field 393b is updated with the proper dynamic port number, and the CA
15 digital signature and credentials are maintained in L5 field 394b without the need for a
16 new AAA validation process.

17 Layer-6 SDNP data 395b is then updated to new security credentials including
18 time t_2 , zone Y (unchanged), cryptographic key 2 (a new key), and updated numeric seed
19 Y2. Concurrently C&C packets download a link to new concealment 397b algorithms
20 stored as d'DMZ task cloud diffuse data $|T|$. The result is a change in concealed payload
21 398b, unrecognizable from its predecessor 398a one hundred milliseconds earlier.

22 In *data fragmentation and transport* as shown in **FIG. 24**, the data contained
23 within a single packet represents only a fraction of the original media content. Without
24 collecting all packets comprising the original file, data file assembly reconstruction
25 cannot be performed, i.e. without the entire encrypted file, decryption and recovery of the
26 plaintext source is not possible. And because of the aforementioned meshed dynamic
27 routing, an attack on a single node cannot intercept all the packets needed to gather and
28 recreate the original encrypted file. For example, in the center column the same
29 illustration, data packets are transported through task nodes 351a, 351b, and 351c using
30 secure connections 352c, 352d, 352e, and 352f, a totally different routing from the
31 previous routing. In the rightmost column data is routed over secure connection 352g to

task node 351c, then over connection 352h to task node 351b, connection 352i to task node 351d, and finally over secure connection 352j to client gateway 350b.

FIG. 26 illustrates the principle of division of labor in hypersecure communications in preventing the concentration of knowledge in any one device or storage medium. In packet routing, name server node [NS] 400 with dynamic IP address 401 sends a datagram to authority node [A] at IP destination address 402 carrying a payload 403 comprising a SDNP address to IP address translation table which in turn is loaded during step 404 into the destination IP address 407 and into the C&C (command and control instructions) in payload 408 of a datagram sent from [A] node 405 having a dynamic IP address 406. The C&C instruction file is forwarded to task node [T₁] 351a describing the construction of a media packet comprising source IP address 391a of node [T₁], destination IP address 392a of node [T₂], SDNP info 395a comprising security credentials, and encrypted payload 398a to be transported to its next destination. In this manner [NS] node 400, [A] node 405, and [T₁] node 351a share only the information they need to instruct the nodes what job must be executed. Separation of labor combined with limiting information exchange to content on a need-to-know basis confounds systematic attacks on a device or on the decentralized SDNP network.

One way to represent the SDNP communication protocol is using the same 7-layer OSI abstraction model used to describe the Internet's TCP/IP protocol. Accordingly, **FIG. 27** depicts two devices communicating over separate SDNP stacks 428 and 429 using SDNP datagram 420 to execute decentralized SDNP routing comprising network Layer-3 data 423 and transport Layer-4 data 424. Atop SDNP hypersecure data transport, session Layer-5 data 425, presentation Layer 6 data 426, and application Layer-7 data 427 facilitate HyperNode operation supporting client applications, online application, and blockchain applications. As shown, HyperSphere communication is performed using Layer 3 through Layer 7 irrespective of PHY Layer-1 data 421 and MAC Layer-2 data 422 data (except when realizing a SDNP enabled HyperSphere router).

Routing, executed via network Layer-3 and transport Layer-4, is fully decentralized, so although HyperSphere *routing* is not open source *per sé*, transport over a dynamic meshed network in autonomous and distributed, meaning no one controls it or is even capable of monitoring packet routing. Moreover, as a decentralized system, no

1 network operator, government, or hacker can circumvent packet routing or usurp network
2 operations. Aside from autonomous network utilities, HyperNode user functions
3 available on Layers 5 through Layer 7 are open source, allowing users to develop and
4 enforce their own proprietary authentication, security, and applications atop the
5 HyperSphere as a platform.

6 As represented in network Layer-3 of the SDNP protocol stack, not only are
7 decentralized SDNP-packet IP-addresses dynamic (changed frequently), HyperSphere
8 routing does not involve the Internet's domain name servers (DNS). Instead the SDNP
9 name server function linking dynamic IP addresses to a user's identity, phone numbers,
10 physical devices, MAC addresses, etc. is realized in a fully decentralized manner via the
11 stored on the d'SDNP name-server diffuse [NS] cloud, accessed through metamorphic
12 HyperNodes and executed only through HyperSphere [A] authority nodes. HyperSphere
13 IP datagrams specify only single-hop source and destination IP addresses but not disclose
14 a packet's point of origination or its ultimate destination.

15 Obfuscating the SDNP packet's true origin and destination using anonymous data
16 packets prevents tracing a call's origin, thereby protecting clients from profiling. Without
17 meaningful packet routing addresses, there is no way for hackers to use sniffing or
18 surveillance to determine which packets are related to one another. To further confound
19 metadata surveillance and DOS attacks, SDNP protocol for Transport Layer-4 employs
20 *ad hoc* dynamic port addresses having no particular assigned port number or defined
21 service (such as email, FTP, etc.) by which an attacker can analyze packet content
22 contextually. And rather than using SSL or TLS transport security (notoriously
23 vulnerable to attack), d'SDNP transport security is preformed by tunneling protocols
24 (such as IPSec), executed on a hop-by-hop basis in the cloud.

25 To maximize quality-of-service (QoS), the HyperSphere's transport protocol
26 employs both TCP and UDP transmission methods depending on the nature of the
27 payload. While the Transmission Control Protocol (TCP) is employed for high reliability
28 payload delivery such as software code and content delivery, User Datagram Protocol
29 (UDP) is employed for realtime (RT) communication such as voice, live video, and other
30 realtime services. Moreover, RT datagrams will be routed by authority node [A] over the

1 network's shortest propagation delay paths. TCP routing is reserved for 'high integrity'
2 delivery focused on maximizing reliability with a higher degree of redundancy.

3 Being less time sensitive, TCP packets are likely routed over entirely different
4 meshed routes than UDP packets. Using the foregoing methods, unauthorized
5 reconstruction of disaggregated digital content, conversations, media, or transactional
6 sessions is prevented. As described, d'SDNP network operation confounds single-point
7 attacks by limiting the content carried by any single node in the network, both by
8 fragmenting the data and by only routing single packets through the same task nodes.

9 A unique function of SDNP packet transport is the generation of "HyperNode hop
10 codes," namely HHC 434. A HHC is a transitory blockchain created adjunctively during
11 packet transport needed for blockchain processing (BCP) in blockchain apps,
12 HyperContracts, blockchain-as-a-service (BaaS), and used in token and cryptocurrency
13 transactions. Because they are generated as part of decentralized dynamic meshed data
14 routing of data packets, HHC 434 contains randomized cryptographic hash values that
15 repel counterfeiting and cloning.

16 To ensure privacy, the HyperSphere privatizes each session (and its contents
17 therefrom) using Layer-5 data 425. Session data includes identity confirmation via AAA
18 login dialogs, security credentials delivered by authority node issued |A| command and
19 control (C&C) packets, and through digital CA certificate based trusted devices and
20 transactions (described later).

21 Another attribute of d'SDNP network operation employs authority node issued |A|
22 command and control (C&C) packets containing security credentials and algorithms
23 combined with presentation Layer-6 data 426 to conceal the content of data packet
24 payload 427 and to prevent unauthorized access. State-based security means that the
25 security methods and credentials used to protect SDNP datagrams change with its state. A
26 security 'state' is a condition existing at the moment of a data packet's creation, e.g.
27 network time, location, security zone, etc. Packet concealment in the HyperSphere
28 comprises modifying payloads using a variety of state-based security mechanisms
29 executed as data packets traverse the spatiotemporal network, including ...

- 30
- 31 • Dynamic splitting and mixing,

- 1 • Dynamic scrambling and unscrambling,
- 2 • Dynamic encryption and decryption [398] [399]
- 3 • Dynamic junk data (or packet) insertion and deletion
- 4 • Time and zone dependent states

5

6 Decentralized SDNP payload security employs the same dynamic concealment

7 methodology using *state-based* security algorithms and credentials as the fixed-

8 infrastructure SDNP protocol disclosed in the above-referenced U.S. Patent No.

9 9,998,434, except that the HyperNodes are metamorphic and the DMZ server is realized

10 over a diffuse cloud. As explained previously, state-based dynamic concealment is

11 illustrated in **FIG. 25** and **FIG. 26** where security credentials, algorithms, and payload

12 content depend on network time, changing on a hop-by-hop basis delivered in SDNP

13 command and control packets 431. The dynamic states are further subdivided into zones,

14 geographic regions (subnets) within the HyperSpheric cloud.

15 Using the foregoing methods means that no two packets traversing the

16 HyperSphere have the same construction. So even if in the unlikely event two packets

17 could be identified as part of the same conversation or session, the packets won't have the

18 same state-based security credentials (keys, seeds, tags, zip) and won't employ the same

19 fragmentation, scrambling, encryption, or junk data algorithms. In other words,

20 identifying two related datagrams does not help improve a hacker's chance of

21 reconstructing secure message content. Dynamic security, where the network state

22 changes constantly with time, disclosed herein therefore comprises a DyDAG, a dynamic

23 version of a DAG (directed acyclic graph). DyDAG operation and its technological

24 realization is a fundamental component of HyperSphere routing, commerce, and

25 blockchains (discussed later herein).

26 Payload 427 of SDNP datagram 420 may contain a variety of content types,

27 including files 430 containing real time data (live video, audio), media content such as

28 photos and videos, application code (software distribution), and user and application

29 specific Layer-7 security credentials including cryptographic keys, CA certificates, etc. In

30 |A| routing tasks, payload 427 may include SDNP command and control instructions such

31 as C&C packet 431. Payload 427 may also securely deliver a number of blockchain

related files including executable code (HyperContract 432), transactions on a DLT/BC database 433, and tokens or cryptocurrency such as HyperCoin 434, Bitcoin, and other digital assets.

Deploying HyperSpheric Nodes

As shown in **FIG. 28**, HyperNodes 380 can be installed onto any host device to convert it into a HyperSpheric node. Devices may include high-performance AC-powered systems including servers 370, desktop computers 384, and crypto mining machines 361. Other suitable devices, albeit having lower performance, include WiFi routers 368, Ethernet routers 369, gaming consoles 365, HDTVs 367, and IoT devices, e.g. smart refrigerators 366. Mobile devices such as tablets 361 and cell phones 360 have extensive computing capability but limited battery life. Vehicles such as cars 371 and commercial trucks 372 have the potential of forming dense communications grids, especially valuable when mobile networks fail.

The significance of realizing an interoperable hypersecure high-performance cloud based entirely on software with no need to acquire or install a single piece of hardware means that the total number of HyperNodes is capable of rapidly expanding to exceed every commercial network on earth, even AWS, GWS and Azure (which also host HyperNodes). But unlike large commercial clouds, the HyperSphere also is able to enlist the support of personal computers, smartphones, Internet routers, gaming consoles, Bitcoin miners, HDTVs, IoT devices (like refrigerators), cars and trucks.

Today the Internet today employs 200 million servers and supports 1 billion network connected PCs. Cell phone users comprises 66% of the world population, 5 billion people purchasing 9 billion subscriptions— one billion more subscriptions than there are people on earth, meaning many people have more than one cell phone. Moreover, IoT devices today comprise 27B units and are expected to exceed 60B units by 2024 (<https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/>).

As described previously, access to the HyperSphere may comprise a HyperNode |HN| 206 software-based autonomous network element, a HyperSphere gateway |HG| 201 embedded in a client's application, or a HyperSphere router |HR| 205 managing L1/L2

physical connectivity. As shown in **FIG. 29**, for convenience these HyperSpheric software instantiations may collectively be referred to as “HyperSphere portal software 380.” And although a HyperSphere portal comprises generic software available for each major host OS, its functional performance varies based on the host device's hardware. In particular, the HyperSphere ports can be optimized for communication-networking 381, distributed computing 382, diffuse cloud data storage 383, and IoT cloud connectivity 384:

- HyperSphere portal software 380 comprising communication network components 381 including client edge devices (such as mobile phone 360), network routers, and commercial server clouds (not shown) perform minimal computing but *require high bandwidth connectivity* across high speed networks preferably comprising optical fiber wireline and microwave radio links, gigabit Ethernet routers, high speed WiFi such as 802.11ac, and 4G or 5G mobile networks. One key requirement of fixed infrastructure hosting of HyperSphere portals is the need for steady reliable power, i.e. high uptime and perpetual availability.
- HyperSphere portal software 380 also supports peer-to-peer (P2P) or vehicle-to-vehicle (V2V) communication. Peer communication is important when access to fixed wireline or mobile infrastructure is unavailable (e.g. when driving in the mountains) or when the network is offline (such as in a natural disaster). HyperSpheric peer communication necessarily operates differently than communication over omnipresent networks, soliciting or commandeering available resources such as cell phones and automobiles as available. In rural regions lacking any wireline infrastructure whatsoever, HyperSphere communication must rely on low carrier frequency mobile telephony, satellite networks, and in extreme cases drone hosted wireless connectivity or on COW (cell-on-wheels) networks. Performance and QoS aside, HyperSphere portal software 380 operates in a manner agnostic to the physical network that hosts them, with the only proviso being the HyperSphere portal software for |HN|, |HG| or |HR| must be compatible with the OS of its device host.

- 1 • HyperSphere portal software 380 comprising distributed computing components

2 382 employs devices with substantial computing capability (generally measured

3 in MFLOPs or millions of floating point operations per second) and sufficient

4 capacities of scratch pad memory (volatile memory, i.e. RAM) to execute the

5 calculations without undue amounts of memory swaps (read / write cycles). Hosts

6 may include high-speed commercial servers and server farms, or an aggregation

7 of large numbers of personal computers 364. Other high performance hosts

8 include cryptocurrency mining computers and gaming computers. The most

9 beneficial computing resources maintain constant power for high uptime and

10 network availability. That said, the sheer number of smartphones in the world

11 (9B), properly managed, represent a significant computing potential, provided

12 each subtask doesn't require more time than the per-charge battery life of the

13 device or overwhelm the volatile memory capacity of the device.
- 14 • HyperSphere portal software 380 comprising diffuse cloud storage 383 employs

15 either large capacity server farms (with correspondingly substantial memory

16 capacity) or a large number of smaller capacity online-connected memory

17 devices. Smaller capacity storage demands a greater degree of fragmentation and

18 memory maintenance.
- 19 • HyperSphere portal software 380 comprising IoT cloud connected devices 384

20 comprise lightweight devices having little demand for computation, storage, or

21 communication bandwidth. Remote devices, e.g. security cameras however,

22 require low power consumption. The HyperSphere portal software, however, does

23 not require significant power to host and as such is naturally IoT compatible and

24 eco-friendly.

26 **Decentralized (d'SDNP) Hypersecure Communication**

27 Examples of the decentralized d'SDNP hypersecure communication use cases include:

- 29 • HyperSphere d'SDNP messenger app
- 30 • HyperSphere V2V *ad hoc* peer networks
- 31 • HyperSphere multiband communication

- 1 • HyperSphere over 5G mobile network
- 2 • HyperSphere distributed computing
- 3 • HyperSphere diffuse data cloud storage
- 4 • HyperSphere cloud connected devices

5
6 Each of the foregoing applications includes inventive matter required to execute the
7 required tasks for the specific applications:

8 *HyperSphere d'SDNP Messenger App:*

9 One application of HyperSphere communication is to provide hypersecure, text,
10 voice, and video in a personal messenger application. As shown in **FIG. 30**, d'SDNP
11 enabled personal messenger 447 communicates with personal messenger 443 over the
12 decentralized SDNP cloud 440. Personal messenger 447 comprises d'SDNP 448
13 messenger app hosted on a cell phone or tablet (not shown) interfacing with microphone
14 450a and speaker 450b. The personal messenger includes embedded HyperSphere
15 gateway software HG 449 providing access to the d'SDNP cloud 440. The gateway
16 facilitates registration of the messenger and device on the HyperSphere name server
17 diffuse cloud and manages the connection dialog to open a communication session with
18 other d'SDNP communication apps.

19 In a call from messenger 447 to 443, sound from microphone 450a is digitized,
20 fragmented, scrambled, encrypted (using shared key 442), and then split into multiple
21 packets. During data transport within d'SDNP cloud 440, data packets are transported
22 using dynamic meshed transport of fragmented data packets over HyperNodes 441, the
23 datagram ultimately reaching HyperSphere gateway 445 embedded within messenger app
24 443. Once a series of related datagrams are received, recombined, decrypted and
25 unscrambled, the digital sound packet is converted into analog signals, driving speaker
26 446b. In normal VoIP telephony, full-duplex conversation involves the concurrent
27 delivery of messages from microphone 446a in personal messenger 443 to speaker 450b
28 in messenger 447. Each sequential packet follows a unique trajectory through the d'SDNP
29 cloud 440 under direction of |A| node performing acting as a dispatcher.

30 Messenger security involves two unrelated processes, one occurring outside cloud
31 440 involving the edge devices, the other functioning autonomously inside cloud 400. In

one embodiment, shown in **FIG. 31**, edge security employs encryption using one or more cryptographic keys 442 transferred directly between the two messengers. As shown, audio waveform 451a is converted into an unencrypted file 451b (also known as plaintext) then encrypted in process 452 using key 442 resulting in ciphertext 451c. After transport through the d'SDNP cloud, edge encrypted file 451c is transmitted to application 443, decrypted in process 453 into unencrypted (plaintext) file 451b, and finally converted back into audio waveform 451.

Because edge encryption involves a cryptographic key 442 transferred directly without network 440's involvement (for example by transferring keys in person, or hidden in a innocuous data stream of voice, video, or music), then edge encryption facilitates privacy benefits to callers separate from network security. In any event, key 442 transfer should occur prior to initiating any communication session— days, weeks, or months before so that use of the encrypted channel with edge shared secrets is undetectable, like forming a VPN long before communicating.

Security within the decentralized SDNP cloud involves a separate process from edge security. Unlike edge security, dynamic cloud security does not involve any party's participation. It occurs autonomously without the involvement of any network operator. **FIG. 31** illustrates the dynamically varying content of d'SDNP datagrams during network transport. Security provisions within d'SDNP cloud 440 occur on a hop-by-hop basis using dynamic security credentials and shared secrets, secret information exchanged prior to the commencement of any communication session. For example, data processed by HyperSphere gateway |HG| 449 involves scrambling operation 453 and encryption operation 454 made in accordance with state S_1 . State S_1 may be based on the time the scrambling operation occurs, the time the task node accepts a job from a HyperContract, the security region of a device, or some combination of these and other time-varying parameters.

Scrambling of packet 451c by operation 453 results in data packet 451d. The packet is then encrypted and split into fragments, including the algorithmic insertion of junk data into the data payload or other packet data. In addition to being encrypted and contaminated with meaningless junk data, the packet is also mixed with the content of other communication sessions. The resulting data packet 451e lacks the full content of its

ancestral lineage (packet 451d) because some of the fragmented pieces are diverted onto different paths through the meshed network. Unrelated data or junk data are represented in the figure by empty squares □ in the text.

HyperNode |HN| 441a decrypts packet 451e in decryption process 455 in accordance with state S_1 , recovering the fragmented unencrypted remains of the data packet, which bears no resemblance whatsoever to the original data packet 451b. HyperNode |HN| 441a then encrypts the data packet's payload anew in accordance with state S_2 . Knowing how to decrypt data packet 451e, therefore, offers no help in decrypting data packet 451g. The decryption process 457 (using state S_2) followed by encryption process 458 (using state S_3) is repeated in |HN| 441b and any other intermediary HyperNodes (not shown) until exit gateway node |HG| 445 de-junks and combines all related incoming packets from the meshed network, then merges the files algorithmically, and decrypts the resulting packet in accordance with the last state used in its concealment process (shown here to be state S_3), a process that recovers scrambled packet 451d. Since the restored packet 451d is still scrambled, a brute force attack to break the cipher and emulate decryption process 459 is wholly unproductive because there is no way for the hacker to know if they successfully decrypted the incoming file.

In the last step, packet 451d is unscrambled within HyperSphere gateway |HG| 445 embedded in messenger application 443, recovering data packet 451c, still encrypted by the edge encryption process. Only a holder of edge user key 442 can then decrypt the packet to extract the plaintext file 451b.

HyperSphere V2V ad hoc Peer Networks:

Another virtue of the decentralization of a meshed autonomous network is its ability to instantly form autonomous peer-to-peer (P2P) and vehicle-to-vehicle (V2V) networks whenever fixed infrastructure or mobile networks fail. As stated previously, the interface between an edge device and the d'SDNP cloud is a multi-function software portal referred to as a HyperSphere gateway. Unlike a symmetrically sandboxed HyperNode being isolated from its host, a |HG| node has access to application data from its host's device. For example, in **FIG. 32A** a group of automobiles 475, 476, 477 and

1 478 communicate with mobile network cell tower 470 over radio links 471, 472, 473, and
2 474, respectively.

3 Since cell tower 470 is enabled with HyperSphere router |HR| 470a connected to
4 corresponding HyperSphere gateways |HG| 475a, 476a, and 477a hosted in cars 475, 476,
5 and 477, respectively, radio links 471, 472, and 473 are hypersecure because they employ
6 HyperSphere node-to-node packet routing (the same as any cloud hosted HyperNode
7 operates). In this sense, the HyperSphere's cloud reaches all the way to the edge devices.
8 In other words, for network Layer-3 there is no such thing as Last Mile communication in
9 a d'SDNP network's implementation. Note that because car 478 does not host a
10 HyperSphere gateway, radio link 474 is not secure.

11 As an academic point, it can be argued that HyperNode gateways behave as a
12 peer-to-peer (P2P) network (or in the case of automobiles, as a vehicle-to-vehicle (V2V)
13 network) even when the cellular network is available as a network backbone. This peer
14 communication feature uniquely occurs on network Layer-3 because HyperSphere portals
15 (nodes, routers, gateways) only trust one another. In this sense, every HyperSphere node
16 operates as a dynamic autonomous VPN with its neighbors, except that the
17 HyperSphere's node-to-node tunnels are created through the |A| node's dispatcher
18 function. In other words, the d'SDNP cloud forms virtual tunnels on Layer-3 tunnel, not
19 on application Layer-7 (the way conventional VPNs operate). Moreover, VPNs must be
20 established prior to communication. Nodes in the d'SDNP cloud form hypersecure links
21 with other nodes, autonomously and automatically.

22 **FIG. 32B** illustrates the instinctive adaptation of the d'SDNP cloud into a V2V
23 network in the absence of network infrastructure. For example, if cars 476, 477 and 478
24 travel into the foothills of a mountain range, they may be cut off from cellular network
25 connectivity. If, however, car 475 is still able to connect to the cell tower 470, then
26 HyperSphere gateway |HG| nodes 475a, 476a, and 477a spontaneously convene a V2V
27 peer network maintaining connectivity to the cell tower 470 and HyperSphere router 470a
28 through the intervening vehicles' |HG| nodes. As a result, vehicle to infrastructure (V2I)
29 link 471 and V2V links 480 and 481 are all hypersecure. Even if data from car |HG| 477
30 is transported to cell tower |HR| node 470a, intermediate vehicular HyperSphere
31 gateways 475a and 476a cannot inspect the data being transmitted. Without a

HyperSphere gateway node, vehicle 478 loses all connection once it is out of tower range.

In **FIG. 32C**, cars 475, 476 and 477 all drive out of range of cell tower 470. In such case the HyperSphere gateways spontaneously transform into a fully self-sustaining V2V network to maintain connectivity across hypersecure channels 480 and 481. V2V communication is important, especially in inclement weather, to prevent against collusions and to warn other cars of impending hazards.

The *ad hoc* V2V network is fully autonomous, adding and removing resources as they enter and leave the radio range between the cars. For example, in **FIG. 32D**, another vehicle 485 containing |HG| node 485a drives into the vicinity of the other cars. The HyperSphere gateway nodes 475a and 477a instantly and spontaneously add hypersecure links 483 and 484 to connect to |HG| 485a. In **FIG. 32E**, car 476 leaves the road and is dropped from the V2V peer network. To maintain the best possible QoS connectivity, |HG| node 475a automatically establishes hypersecure link 486 to |HG| node 477a.

HyperSphere Multiband Communication:

A key principle of d'SDNP cloud communication is fragmented data transport across multiple paths in a meshed network. Although these paths can represent different combinations of HyperNodes in a fixed infrastructure cloud of servers, in connections to edge devices additional redundancy and security can be achieved by sending the various packets across multiple PHY connections.

One such implementation, shown in **FIG. 33A**, comprises a wireless access point, forming a bridge between Ethernet MAC access 500 and a multiband WiFi radio 501. The access point combines three communication stacks 502, 513a, and 513b, facilitating routing among Ethernet 501 and microwave radios 515a and 515b. The host platform comprises interface circuitry & radio host kernel, (herein “kernel”) 509 connected through driver interface 506 to host processor 508, through driver interface 505 to local data storage (nonvolatile memory), through driver interface 503 to the PHY Layer 1 of 802.3 (Ethernet) communication stack 502, through driver interface 511a to the PHY Layer 1 of 802.11 (WiFi) communication stack 513a for radio A, and through driver

1 interface 511b to the PHY Layer 1 of 802.11 (WiFi) communication stack 513b for radio
2 B.

3 Data packets carried by Ethernet connection 501 are interpreted by
4 communication stacks 502 and passed via data link Layer-2 data 504 to kernel 509. In
5 accordance with routing instructions received by d'SDNP authority node, HyperSphere
6 router [HR] node 510 processes the incoming packet, parses it into separate payloads. The
7 payloads are then assembled into packets having different IP and MAC addresses– one
8 transferred over data-link layer connections 512a and 514a to radio 515a and broadcast
9 over antenna 516a with a carrier frequency f_A .

10 The second payload is transferred over data link layer connections 512b and 514b
11 to radio 515b and broadcast over antenna 516b over a carrier frequency f_B . While the
12 above sequence describes an incoming Ethernet packet being split and sent across
13 multiple radio carrier frequencies, in full duplex communication and telephony, the
14 process is bidirectional. For example, multiple WiFi radio signals once received, are
15 processed, mixed, then split, and sent as multiple packets over Ethernet 501 sharing a
16 common MAC address but with separate IP addresses.

17 The HyperSphere enabled WiFi router (herein referred to as a HyFi router)
18 uniquely involves a process of mixing and splitting data packets and sending or receiving
19 them over multiple carrier frequencies. This methodology is exemplified in **FIG. 33B**
20 where data packets carried by Ethernet wireline 522 to HyFi router 368 are split and sent
21 to cell phone 360 over a multi-PHY microwave radio links comprising, for example, a
22 2.4 GHz channel 521a, a 5 GHz channel 521b, and a 900 MHz channel 521c. Packet
23 construction is managed using HyperSphere portal software comprising [HR] router 510
24 and [HG] gateway 520. In the following discussion, in regards to MAC addresses and IP
25 address for brevity's sake, the abbreviation L2 shall mean data link Layer-2 and L3 shall
26 mean network Layer-3. Although the example shows two radio links and one Ethernet
27 connection, the number of wireless and wireline router channels may be increased with
28 minimal impact to performance.

29 Ethernet packet 530 is routed from a server hosting HyperNode [HN] (not shown)
30 at L2 source address 530b (with a value MAC [HN]) and a L3 source address 530c (with a
31 value IP {SDNP [HN]}). Ethernet packet 530 is routed to HyperSphere HyFi router 368

hosting router node |HR| 510 at L2 destination address 530a (with a value MAC |HR|) and a L3 destination address 530d (with a value IP {SDNP |HR|}). Blank field 530e contains L4 to L6 data left blank for clarity's sake. The payload 530f of Ethernet packet 530 contains both SDNP1 and SDNP2 sub-packets. For clarification, the term SDNP |HN| is the SDNP address of HyperNode |HN| while the value IP {SDNP |HN|} is the dynamic IP address of the device hosting HyperNode |HN| and associated with address SDNP |HN|.

After payload processing and concealment, WiFi packet 531 is routed from HyperSphere HyFi router 368 hosting router node |HR| 510 at L2 source address 531b (with a value MAC |HRC| for channel C) and a L3 source address 531c (with a value IP {SDNP |HR1|} where HR1 is the 1st IP address). WiFi packet 531 is routed to smartphone 360 hosting gateway |HGC| 520 at L2 destination address 531a (with a value MAC |HGC|} for channel C) and a L3 destination address 530d (with a value IP {SDNP |HG1|}). Blank field 531e contains L4 to L6 data left blank for clarity's sake. The payload 531f of WiFi packet 531 contains only SDNP1 sub-packets.

After payload processing and concealment, WiFi packet 532 is routed from HyperSphere HyFi router 368 hosting router node |HR| 510 at L2 source address 532b (with a value MAC |HRA| for channel A) and a L3 source address 532c (with a value IP {SDNP |HR2|} where HR2 is a 2nd IP address). WiFi packet 532 is routed to smartphone 360 hosting gateway |HGA| 520 at L2 destination address 532a (with a value MAC |HGA|} for channel A) and a L3 destination address 532d (with a value IP {SDNP |HG2|}). Blank field 532e contains L4 to L6 data left blank for clarity's sake. The payload 532f of WiFi packet 531 contains only SDNP2 sub-packets.

The methods of d'SDNP packet routing apply equally to wireline communication such as 802.3 (Gb Ethernet). **FIG. 34A** illustrates a HyperSphere enabled wireline router 570, forming a bridge among multiple Ethernet MAC addresses 575a, 575b, and 575c. The router-bridge combines three communication stacks 577a, 577b, and 577c facilitating routing among Ethernet connections 576a, 576b, and 576c. The host platform comprises interface circuitry & radio host kernel, (herein “kernel”) 571 connected to host processor 574, to local data storage (nonvolatile memory) 573, to the PHY Layer 1 of 802.3 (Ethernet) communication stacks, namely 579a, 579b, and 579c.

1 Data packets carried by any of the Ethernet connections, e.g. 576b, are interpreted
2 by the corresponding communication stack 577b and passed via L2 data link 578b to
3 kernel 571. In accordance with routing instructions received by d'SDNP authority node,
4 HyperSphere router |HR| node 572 processes the incoming packet, parses it into separate
5 payloads. The payloads are then assembled into packets having different IP and
6 optionally MAC addresses, e.g. transferred over unused Ethernet channels 576a and
7 576c.

8 The physical media may vary from copper twisted pair to the more common
9 optical fiber at various wavelength of infrared light. The protocol itself may also vary in
10 data rate from 100 Mb/s to 1,000 Mb/s (so called Gb Ethernet) and in cable length from
11 50 km down to a few meters. For example, multiple Ethernet protocols, once received,
12 are processed, mixed, split, and sent as multiple packets over Ethernet connections 576a
13 through 576c with separate IP addresses and distinct MAC addresses.

14 The HyperSphere enabled Ethernet router (herein referred to as a Ethyrnet router)
15 uniquely involves a process of mixing and splitting data packets and sending or receiving
16 them over multiple Ethernet channels. This fragmented transport methodology is
17 exemplified in **FIG. 34B** where data packets carried by Ethernet wireline 556 are
18 received by Ethyrnet quad channel router 369, split, recombined into new payloads, and
19 forwarded by router |HR1| via multiple hypersecure channels. The wireline channel may
20 include channel 555a comprising 100BASE--KX (over copper), channel 555b
21 comprising 1000BASE-SX limited to 220m optical fiber, and channel 555c 1000BASE-
22 BX10 capable of supporting up to 10-km fiber communication.

23 The destination of any Ethyrnet packet depends on packet routing determined by
24 the dispatcher function of the HyperSphere's |A| authority nodes. Examples include |HG|
25 gateway 551a hosted by personal computer 552, |HN| HyperNode 551b hosted by server
26 553, or HyperSphere router |HR2| hosted on high bandwidth router 554. Packet
27 construction is managed using HyperSphere portal software, comprising |HR1| router
28 550, |HG| gateway 551a, |HN2| HyperNode 551b, or |HR2| router 551c. As in the prior
29 WiFi example, for brevity's sake the abbreviation L2 shall mean data link Layer-2 and L3
30 shall mean network Layer-3. Although the example shows three Ethernet wireline

connections, the number of router channels may be increased with virtually no adverse impact on device performance.

Referring again to **FIG. 34B** incoming Ethernet packet 560 is routed from a server hosting HyperNode |HN1| (not shown) at L2 source address 560b (with a value MAC |HN1|) and a L3 source address 560c (with a value IP {SDNP |HN1|}). Ethernet packet 560 is routed to HyperSphere Ethyrnet router 369 hosting router node |HR1| 550 at L2 destination address 560a (with a value MAC |HR1|) and a L3 destination address 560d (with a value IP {SDNP |HR1|}). Blank field 560e contains L4 to L6 data left blank for clarity's sake. The payload 560f of incoming Ethernet packet 560 contains both SDNP1 and SDNP2 sub-packets.

After payload processing and concealment, packet 561 is routed from Ethyrnet router 369 hosting router node |HR1| 550 at L2 source address 561b (with a value MAC |HR1|) and a L3 source address 531c (with a value IP {SDNP |HR1|} where HR1 is the 1st IP address). Ethyrnet packet 561 is routed to high bandwidth router 554 hosting router |HR2| 551c at L2 destination address 561a (with a value MAC |HR2|) and a L3 destination address 561d (with a value IP {SDNP |HR2|}). Empty field 511e contains L4 to L6 data left blank for clarity's sake. The payload 561f of WiFi packet 561 contains only SDNP1 sub-packets.

After payload processing and concealment, packet 562 is routed from Ethyrnet router 369 hosting router node |HR1| 550 at L2 source address 562b (with a value MAC |HR1|) and a L3 source address 561c (with a value IP {SDNP |HR1|} where HR1 is the 2nd IP address). Ethyrnet packet 562 is routed to desktop 552 hosting gateway |HG| 551a at L2 destination address 562a (with a value MAC |HG|) and a L3 destination address 562d (with a value IP {SDNP |HG|}). Empty field 562e contains L4 to L6 data left blank for clarity's sake. The payload 562f of WiFi packet 562 contains only SDNP2 sub-packets.

Another example of HyperSphere communication is in cable modems used to connect homes and businesses to high-speed wireline connections. Like the descriptions of HyFi and Ethyrnet routers described previously in this application, content distribution using DOCSIS-3 over cable and fiber is bidirectional capable of full duplex operation, using packet switched technology. Cable operators use a special protocol called DOCSIS-

3 to manage content distribution and high-bandwidth communication traffic over hybrid networks of fiber and coaxial cable. An acronym for “Data Over Cable Service Interface Specification,” is an international telecommunications standard designed to support a dynamic blend of Internet, audio, and video (media) content audio including unicast HDTV, subscription HDTV (including broadcast television content), pay-per-view HDTV, and web-streaming services. CableLabs released the 'DOCSIS-3.1 Full Duplex' (or D3.1-FD for short) specification in October 2017 supporting symmetrical speeds of 10 Gb/s.

Although cable TV represents a declining business, the advent of Internet video streaming services is creating new demand for the high bandwidth offered by cable systems. This trend is forcing new alliances between global Internet cloud-as-a-service (CaaS) providers and cable network operators in order to enhance Last Mile performance to homes and merchants, e.g. Google-Comcast is deploying an extensive cable network for global key client businesses such as Starbucks, who broadcast their own music channels to all their locations. Since significant Internet traffic comes from clients in coffee shop chain using unsecured public WiFi, the potential for cybercrime is enormous. As such, privacy and security protections over cable networks are important considerations in modern networking.

By employing light instead of electrical or microwave signals to carry information on its PHY layer, optical fiber, in particular offers superior bandwidth compared to other forms of communication. The OSI communication stack for DOCSIS3 in a cable distribution system is illustrated in **FIG. 35A** illustrates Layer 1 PHY connectivity and the Layer 2 data link made in accordance with the specification. The DOCSIS specification remains agnostic as to the use of network Layer-3 data.

In order to provide content distribution and video streaming combined with full-duplex high-bandwidth communication, cable distribution services employ a multichannel up-stream device referred to as Cable Modem Termination System (CMTS) 621 combined with multiple client devices comprising cable modems (CM) 601 or set-top boxes (STB) 602 deployed in client's homes and businesses. Specifically, cable modem termination device CMTS 621 and its corresponding communication stack 620 contains a Layer-1 PHY network interface 627 connected to cloud servers 625 and

Internet cloud 625 over wireline 623, or alternatively to various media and content sources including a video headend, IPTV, or VoIP systems (not shown). The combination of network interface 625 and data link layer 628 comprise the device interface communication stack of CMTS 621.

On data link Layer 2, data is passed from the network interface communication stack to the cable network interface communication stack through forwarding function 629, specifically into link level control LLC 669. Link level control LLC 669 comprises a hardware-independent protocol defined in accordance with IEEE specification 802.2. The packet data is then modified by link security 630 to provide limited packet security, primarily to prevent unauthorized viewing of content such as pay-per-view unicast broadcasts. The data packets are then formatted in accordance with DOCSIS3 to include cable MAC 631 addresses for distribution to cable modems and set top boxes.

The Layer-1 PHY cable interface 632 then sends the data frames over cable distribution network 619 comprising either coaxial cable 619b or optical fiber 619a to the corresponding Layer-1 PHY cable interface 618 within communication stack 600 of cable modem CM 601 or set top box STB 602.

Upon receiving a data packet, cable MAC interface 617 then interprets the cable MAC addresses, passing its payload to link security 616 for decryption and ultimately to hardware independent link layer control LLC 622 for interpretation. The input data to the CM or STB cable network communication stack 600 is then passed through transparent bridging 613 to the CM or STB device portion of the interface communication stack, specifically to device independent link layer control LLC 614 made in accordance with the specification for IEEE 802.2. The packet is then passed to either HSD & IPTV MAC block 613 or to WiFi 802.11 MAC block 612 to update the packet's MAC addresses. In the case of WiFi communication, the data packet is then passed from 802.11 MAC block 612 to WiFi PHY Layer 1 radio interface 610 for transmission on WiFi radio 605. In the case of wireline connections, the data packet is then passed from HSD & IPTV MAC block 613 to Ethernet or HDMI interface block 611 for connecting to smart HDTV 604a or desktop 604b.

Because of external connections to network interface 627, to Ethernet 611, and to WiFi 610, cable networks suffer the same security vulnerabilities as any TCP/IP based

1 network. As a result, the installation of HyperSphere router software facilitates
2 hypersecure communication with no need to alter the DOCSIS3 protocol. Specifically, to
3 secure the network port of CMTS 621, HyperSphere router software |HR1| 623 is
4 installed in communication stack 620. Similarly, to ensure port security for Ethernet 611
5 and WiFi 610 HyperSphere router software |HR2| 603a and |HR3| are installed in either
6 CM 601 or STB 602. The resulting HyperSphere enabled hypersecure cable network is
7 shown in **FIG. 35B**, comprising HyperSphere router |HR1| 623 hosted by CMTS 621,
8 router |HR2| 603a hosted by STB 602, and router |HR3| 603b hosted by CM 601.

9 Communicating with other HyperNodes in the d'SDNP cloud (not shown),
10 HyperSphere router |HR1| 623 facilitates secure wireline communication over Ethernet
11 634 to the HyperSphere. WiFi communication link 642 between cable modem 601 and
12 notebook 363 is secured by HyperSphere router |HR3| 603b and HyperSphere gateway
13 |HG3| 639, software installed into their corresponding host devices as shown. Similarly
14 installed HyperSphere portal software secures other last link connections including
15 Ethernet link 642 between router |HR2| 603a hosted on set top box 602 and HyperSphere
16 gateway |HG2| 640 hosted on notebook 363; as well as Ethernet link 641b between |HR2|
17 and gateway |HG2| 638 hosted on IPTV 364a. Note that HDMI connection 641a to
18 HDTV 364b is not secure because the client device lacks HyperSphere portal software.

19 Another unique embodiment of this invention relates to hypersecure
20 communication between CMTS 621 and client device CM 601 over fiber link 619y and
21 to STB 602 over fiber connection 619x, involving a cable-specific data modulation
22 scheme referred to as trellis encoding 645. By adapting d'SDNP routing to employ “trellis
23 encoding” 645, different IP and MAC addresses can be carried within the cable
24 distribution sharing a common PHY layer (optical fiber). Similar to OFDM used in WiFi
25 or OFDMA used in 4G/LTE and 5G communications, DOCSIS3 communication
26 employs multiple orthogonal, i.e. non-overlapping frequencies, either in the microwave
27 or optical spectrum of electromagnetic radiation in which in encodes and transmits its
28 information. Rather than assigning content specifically dedicated to each channel, in
29 trellis encoding, video and high-speed data across are dynamically allocated across all
30 available frequency channels based on demand.

As represented in **FIG. 35C**, trellis-encoding 645 involves breaking fiber bandwidth into modulation carrier channels (shown as A through G) and into sequential fixed interval time slots. For example, on-demand video carried on channels B and C 647a is reallocated to a single channel D 647b at time slot 20. The dynamic feature of trellis encoding can be adapted to carry d'SDNP datagrams with separate IP addresses and packet specific security provisions. For example, SDNP |HR3| data carried on channel-A 646a changes to channel-B 646b in time slot 22. Concurrently, using entirely different security credentials and algorithms, SDP |HR2| data 648a is carried split across channels F and G using fragmented data transport. At time slot 25, the carriers are reassigned to channels E and G containing fragmented data 648b and 648c. So in a manner similar to transmitting d'SDNP packets across multiple WiFi carrier frequencies or Ethernet cables, DOCSIS3 accommodates d'SDNP transport for hypersecure communications.

The HyperSphere process of mixing and splitting data packets and sending or receiving them over multiple channels applies to mobile telephony and wireless networks. This fragmented transport methodology is exemplified in **FIG. 36A** where data packets carried by Ethernet wireline 674 are received by mobile network cell tower 470, then processed by HyperSphere router |HR1| 670 for broadcast over radio carriers 671a at 1.9GHz, 671b at 600 MHz, and 671c at 2.5GHz. Carrier 671a connects to HyperSphere router |HR2| hosted by cell phone tower 672a. Carriers 671b and 671c are both routed to HyperSphere router |H32| hosted by cell phone tower 673b.

Although cellular routing employs modulation schemes (2G, 3G/LTE, HSDPA, HSUPA, 5G) and carrier frequencies defined by the PHY Layer-1 and link Layer-2 data and the mobile carrier's network, the ultimate destination of any mobile network packet depends on packet routing determined by the dispatcher function of the HyperSphere's |A| authority nodes. Referring again to **FIG. 36A** incoming Ethernet packet 680 is routed from a server hosting HyperNode |HN| (not shown) at L2 source address 680b (with a value MAC |HR1|) and a L3 source address 680c (with a value IP {SDNP |HNR|}). Ethernet packet 680 is routed to HyperSphere cell tower 470 hosting router node |HR1| 670 at L2 destination address 680a (with a value MAC |HR1|) and a L3 destination

1 address 680d (with a value IP {SDNP |HR1|}). Blank field 680e contains L4 to L6 data
2 left blank for clarity's sake. The payload 680f of incoming Ethernet packet 680 contains
3 both SDNP1 and SDNP2 sub-packets.

4 After payload processing and concealment, packet 681 is routed from cell phone
5 tower 470 hosting router node |HR1| 670 at L2 source address 681b (with a value MAC
6 |HR1|) and a L3 source address 681c (with a value IP {SDNP |HR1|} where HR1 is the
7 1st IP address). Mobile 5G packet 681 is routed over radio link 671c to cell tower 672b
8 hosting router |HR3| 673b at L2 destination address 681a (with a value MAC |HR3|) and
9 a L3 destination address 681d (with a value IP {SDNP |HR3|}). Empty field 681e
10 contains L4 to L6 data left blank for clarity's sake. The payload 681f of WiFi packet 681
11 contains only SDNP1 sub-packets.

12 In a similar manner, after payload processing and concealment, 4G radio packet
13 682 is routed from cell phone tower 470 hosting router node |HR1| 670 at L2 source
14 address 682b (with a value MAC |HR1|) and a L3 source address 682c (with a value IP
15 {SDNP |HR1|} where HR1 is the 2nd IP address). Mobile 4G packet 862 is routed to cell
16 tower 672a hosting router node |HR2| 673aa at L2 destination address 682a (with a value
17 MAC |HR2|) and a L3 destination address 682d (with a value IP {SDNP |HR2|}). Empty
18 field 682e contains L4 to L6 data left blank for clarity's sake. The payload 682f of mobile
19 4G-packet 682 contains only SDNP2 sub-packets.

20 Although fragmented data transport over multi-PHY mobile networks is
21 beneficial for tower-to-tower (T2T) communication and transcontinental routing, the
22 same method can also be adapted to upcoming 5G equipped phones and tablets. This
23 fragmented transport methodology is exemplified in **FIG. 36B**, wherein data packets
24 carried by tablet 362 and HyperSphere gateway 700 are broadcasted over radio carrier
25 channels 701a at 1.9GHz, 701b at 600 MHz, and 701c at 2.5GHz. Carrier 671a connects
26 to HyperSphere router |HR2| hosted by cell phone tower 702a. Carriers 701b and 702c
27 are both routed to HyperSphere router |HR3| hosted by cell phone tower 702b.

28 As shown, transport of mobile 5G data packets 710, 711, and 712 from cell phone
29 362 and HyperSphere gateway |HG| 700 occur across three different carriers 701a, 701b,
30 and 701c.

Carrier 701b transporting 5G packet 710 comprises LS source address 710b having a value MAC |HG1| where HG1 represents the 600 MHz band of cell phone 700. The L2 destination 710a of the packet is MAC |HR3| an address assigned by HyperSphere router 703b. Network routing of data packet 710 from L3 source address 710c to L3 destination address 710d occurs using dynamic IP addresses of the source as IP {SDNP [HG|]} and destination IP {SDNP [HR3|]}.

Carrier 701a transporting 5G packet 712 comprises LS source address 712b having a value MAC |HG3| where HG3 represents the 1.9 GHz band of cell phone 700. The L2 destination 712a of the packet is MAC |HR2| an address assigned by HyperSphere router 703a. Network routing of data packet 712 from L3 source address 712c to L3 destination address 712d occurs using dynamic IP addresses of the source as IP {SDNP [HG|]} and destination IP {SDNP [HR2|]}.

Carrier 701c transporting 5G packet 711 comprises LS source address 711b having a value MAC |HG2| where HG2 represents the 2.5 GHz band of cell phone 700. The L2 destination 711a of the packet is MAC |HR4| an address assigned by HyperSphere router 703b. Network routing of data packet 711a from L3 source address 711c to L3 destination address 711d occurs using SDNP addresses, not dynamic addresses) of the source as SDNP [HG|] and destination SDNP [HR3|. In this manner routing does not necessarily depend on dynamic IP addresses.

HyperSphere Cloud Connected Devices:

Decentralized SDNP routing can also be implemented to secure IoT devices. As shown in **FIG. 37A**, IoT cloud-connected device 780 comprises WiFi radio 780c and host device 780a including IoT host kernel 781, microcontroller 784, local nonvolatile memory 783, power supply 792, load driver 793, 802.11ah compatible radio 786 and two communication stacks 788a and 788b. In operation signals received by antenna 785 are demodulated and converted to digital data 787 and passed to IoT kernel 781 via connection 790a. Data link information 789a is then processed by HyperNode gateway 782 and the control instructions 789a passed to the data link layer of communication stack 788b, which interprets the information converting PHY digital data 790b into control signals 791 to drive the load via load driver 793 powered by power supply 792.

As depicted in **FIG. 37B**, HyperSphere gateway 782 prevents unauthorized control of IoT device 780, control of the IoT load 790, or usurpation of the network through infiltration of “dumb,” low level IoT devices. Using the HyperSphere as a control platform, HyperNodes [HN] 795 can send commands 795a and software downloads into HyperSphere gateway [HG] 782 enabled IoT devices, but uploads from the [HG] node 782 to the local network are limited to data 797. Commands and information requests 799 represent unauthorized access and are blocked by sandbox protections 799.

An example of a HyperSphere protected IoT network is illustrated in **FIG. 37C**, where wireless router 368 is HyFi enabled by HyperSphere router software [HR1] 751, protecting the entire network from unauthorized intrusion. IoT devices including smart refrigerator 366, IPTV 364a, thermostat or HVAC controller 750a, security camera 750b, IoT door locks 750c and controllable lighting 750d are protected by IoT embedded HyperSphere gateways HG3 753a through HG8 753f, respectively and controlled by application software installed in tablet 362 or personal assistant 751 including protective HyperSphere gateways HG2 752c and HG1 752a. Secure control of the IoT devices outside of the trusted HyFi zone can be accomplished using wireline connected desktop 364 hosting HyperSphere gateway [HG9] 754, or controlled via cell phone 360 through HyperSphere gateway 756 facilitating a hypersecure network through cell tower 702a to server 370 hosted HyperNode [HN] 755 comprising one of many decentralized nodes in the HyperSphere.

HyperSphere Diffuse Data Cloud Storage:

FIG. 38 illustrates the redundant file storage of user data. For example a parsed file of client files 801a, 801b, 801c is written into a number of storage servers 802, 803, and 803 respectively comprising task nodes [T1], [T2], and [T3] collectively performing diffuse cloud storage for client data 800. As shown, parsed file 1 comprising data 801a is redundantly stored on both [T1] and [T3] and others (not shown); parsed file 2 comprising data 801b is redundantly stored on both [T2] and [T3] and others; and parsed file 3 comprising data 801c is redundantly stored on both [T1] and [T3] and others.

1 The storage of file 1 and file 3 in |T1₃|, file 2 and file 3 in |T2|, and file 1 and file 2
2 in |T3| automatically generate file storage access link keys 805, 806, and 807
3 respectively, all of which are stored redundantly on client HyperSphere gateway devices
4 |HG1| and |HG2₇| hosted on servers 808 and 809, respectively, and others (not shown). If
5 any |T| storage node goes offline, the data is cloned onto another name server node. In
6 one embodiment, each |T| node issues and maintains a dynamic HyperContract listing
7 backup servers, currently online. If a server goes offline, the HyperContract automatically
8 clones the data to another server, which in turn issues a new backup HyperContract,
9 perpetually *ad infinitum*.

10 As shown in **FIG. 38B**, disaggregated data stored for a user in the HyperSphere
11 can be retained for private use, for example where the data owner uses HyperSphere
12 gateway |HG| 820 to facilitate write 811 and read access 812 to user diffuse data cloud
13 810, or alternative to share write 813 and read 814 access to the cloud with collaborator
14 gateway device |HG2| 821 or just offer read 823 access to reviewers via gateway device
15 |HG3| 822.

16 17 *HyperSphere Distributed Computing:*

18 The HyperSphere's hypersecure communication cloud is also well suited for cloud
19 computing facilitating secure and rapid data transfers across the d'SDNP cloud. As shown
20 in **FIG. 39A**, a cloud computing application hosted on server 831 comprising a
21 HyperSphere portal via gateway |HG| 840 is instructed by HyperContract 850, a
22 computing job comprising a series of tasks defined by job description 851 (including
23 tasks, subroutines, matrix calculations, and job allocation) along with security related
24 information including numeric seed 852, cryptographic key 853, and state 854. The
25 gateway node |HG| 840 then allocates the job to a number of selected task nodes 842,
26 843, 844 and others.

27 As shown in **FIG. 39B**, HyperSphere gateway node |HG| 840 then establishes a
28 secure session link and communication channel with servers 832 through 837 whose
29 installed HyperNodes morph into task nodes |T2| 842 through |T7| 847. The job
30 instruction and security credentials also enable the task nodes to access distributed

1 computing diffuse data cloud 850 in order to upload and download files during job
2 execution.

3 4 **Summary of HyperSphere Decentralized Communication**

5 6 ***Public-Private Network:***

7 The HyperSphere's unique connectivity facilitates the above electronic business
8 processes using a single common-communications platform supporting interoperability
9 and decentralized applications not possible over the Internet. Unlike public-network
10 hosted communication and transactions executed over-the-top of TCP/IP, the
11 HyperSphere operates independently from yet co-existing with the Internet using a global
12 dynamic real-time meshed network. The d'SDNP network is dynamic, able to
13 automatically expand or contract to match regional resources with changing demand
14 while insuring the routing of data packets occurs over the shortest propagation delay
15 paths. In this way HyperSphere network communication maintains superior real-time
16 performance and quality-of-service (QoS) for voice communication and live video.
17 HyperSphere communication employs "hypersecure" protocols, using dynamic network
18 communication methods far beyond simple encryption techniques, sending the data
19 packets on different and changing routes and using anonymous data packets comprising
20 ever-changing encryption and security methods to prevent access or monitoring.

21 The HyperSphere applies these techniques equally to voice, video, text and video
22 communication, to distributed computing tasks, and to HyperSphere financial
23 transactions, thereby preventing any "meaningful" surveillance of network traffic while
24 obfuscating the identity of its users, camouflaging the ultimate source and destination of
25 data packets, and categorically thwarting man-in-the-middle (MiM) attacks. The
26 HyperSphere's network co-exists in parallel with the Internet, sharing some physical-
27 layer connections with cellular networks and the Internet but otherwise operating as a
28 completely independent and parallel system, i.e. metaphorically as "Internet 2.0".

29 Physical realization of the HyperSphere network comprises a heterogeneous cloud
30 of HyperSphere node portals – connections to the HyperSphere comprising HyperNodes,
31 HyperSphere routers and HyperSphere gateways. These nodes comprise application

1 software downloaded onto any computing or communication device, applicable for any
2 size device ranging from global server networks to home PCs, or smartphone to IoT light
3 bulbs. The number of nodes in the HyperSphere naturally grows in proportion to user
4 adoption, requiring no investment in infrastructure to deploy, and no capital expense to
5 expand coverage, improve capability, or enhance performance. So although the
6 HyperSphere's network is public in the sense that any individual or company may join it
7 by downloading and activating a HyperNode, it routes and secures all data transport as a
8 private network. In this manner, a large group of users can exchange private
9 communication as a fully distributed private network even though the private cloud is
10 coexisting within the public HyperSphere. In that sense the HyperSphere could
11 paradoxically referred to as a "public-private" network.

12 As an alternative description, a private HyperSpheric cloud operating within the
13 public HyperSphere operates like an ad hoc virtual private network (VPN), but with no
14 need to establish the VPN prior to making a call or sending a file. The publically hosted
15 private cloud could be described as a self-launching VPN or alternatively as an
16 autonomous VPN. The same self-launching mechanism can be used to create an ad hoc
17 peer-to-peer network. In such cases, any HyperNode being cut off from the cloud
18 automatically looks for nearby devices hosting HyperNodes to connect to even if no
19 cellular or WiFi network exists in the vicinity. Each device in turn searches for other
20 HyperNode host devices until the cloud connection can be re-established.

21 Collectively then, HyperNodes form the HyperSphere's network, not the physical
22 devices that host them. There is no limitation how many HyperNodes may co-exist on a
23 single server or on a common piece of hardware. For example, although the HyperSphere
24 may directly launch its own HyperNode on a specific AWS server, nothing prevents a
25 regular client of AWS services from installing and launching their own HyperNodes over
26 the AWS cloud or even on the same server (so long that the clients pays AWS for their
27 own data traffic). Since HyperNodes hosted by a commercial cloud and clients using the
28 same cloud have their own separate digital signatures and owners, the HyperSphere
29 considers the HyperNodes as independent resource suppliers, and even as competitors.

30 For example, an AWS Cloud-as-a Service (CaaS) client may be using AWS cloud
31 services to compete for HyperContract business with AWS cloud services. Since the

HyperSphere considers each HyperNode as uniquely identified by the digital certificate used to authenticate it, its CA-certificate, the two HyperNode will appear as competing independent vendors. In cryptoeconomics, like in regular commerce, increased competition naturally drives down costs and improves capital efficiency.

Stateless Meshed Routing:

HyperNodes route data packets through the HyperSphere across a meshed network using anonymous data packets with no centralized control whatsoever. As a dispatcher based communication system and protocol, the jobs of HyperNodes involve one of three interrelated yet independently executed functions:

- Identifying devices and HyperNodes chosen to carry data or to execute tasks, referred to as the name server nodes or |NS| nodes.
- Determining data packet routing and providing instructions to HyperNodes executing the tasks as well as confirming when a task has been completed, referred to authority nodes or |A| nodes.
- Carrying data or executing tasks, referred to as task nodes or |T| nodes (called media nodes in the original SDNP patent).

As described in the original Secure Dynamic Network and Protocol (SDNP) patent filing for which the application is a continuation in part, tri-channel communication over dedicated function SDNP nodes utilizes a name server or device hosting a SDNP name server node, a signaling server or device hosting a SDNP signaling server node, a media server or device hosting a SDNP media server node, and a DMZ server or device, aka an 'air-gapped computer', hosting offline functions of a DMZ server. Together these electronic assets, or 'resources' deliver hypersecure communication, multi-level security, and superior performance unparalleled by any network today. In the original SDNP patent filing four key functions – the name server function, the signal server function, the media server function, and the DMZ server supporting shared secrets are hosted on specific computers. Because no one computer has

1 all the information involved in call routing, content, or device addresses, packet transport
2 is secure.

3 Although the network administrator cannot intercept the dynamic real-time
4 operation of any SDNP node, the fact that computers are loaded with specific types of
5 SDNP nodes, and that a network administrator is responsible for the deployment and
6 allocation of the executable code, means the SDNP network deployment is not fully
7 decentralized (even though the network's meshed routing operations are decentralized).
8 Using d'SDNP, the HyperSphere's implementation of the network and the SDNP
9 protocol is, in contrast, fully decentralized, meaning no network administrator knows
10 what functions are being hosted by a specific server at any given time.

11 To achieve this fully decentralized capability, the HyperSphere replaces the
12 dedicated-function SDNP nodes with the newly disclosed metamorphic HyperNode. A
13 metamorphic HyperNode represents software that can perform any one of the
14 aforementioned SDNP functions – name server, signal server, and media (task) server
15 functions, BUT can only perform one of these functions at a time in a given conversation.
16 For example a metamorphic HyperNode can perform the task node services in one
17 conversation, act as an authority node (signal server) in another call, and function as the
18 name server [NS] function in another transactions, but it cannot perform two or more of
19 the task, name server, or authority node functions in the same call or session. In logic,
20 this feature can be considered as an “exclusive OR” function symbolized by $\oplus$.
21 Mathematically the metamorphic HyperNode function [HN] means a functional state
22 comprising “one or another but not more than one”, or as [MHN] = ([NS] $\oplus$ [A] $\oplus$ [T]).
23 The term ‘metamorph’, adopted from science fiction parlance refers to an ability to
24 change, i.e. morph, into something else.

25 In this sense when a call or a transaction is initiated, the metamorphic HyperNode
26 is chosen to serve as one of the three functions, either a [NS], [A] or [T] HyperNode. One it
27 is selected to perform one of the three functions, it is automatically excluded from being
28 able to perform the other two functions in the same job. In that manner no information is
29 concentrated within a single HyperNode. In one embodiment, the job selection of a
30 metamorphic HyperNode is made during HyperContract negotiation in a decentralized
31 AI-based environ referred to as the HyperSphere Marketplace, at which time a

HyperNode's eligible to serve as name server, authority node, or task node is decided. In the selections process, more HyperNodes are selected than are actually required. These extra “backup” HyperNodes are held in reserve in case a selected HyperNode goes offline or is otherwise unable to fulfill its assigned role.

Once selected for a given role, the HyperNode then acquires the information it needs to perform its assigned task by accessing data stored in disaggregated data storage of the HyperSphere. The disaggregated data storage layer operates like a DMZ server in the sense that the data it contains cannot be recalled or read directly from the Internet. Only a HyperNode can extract data from the disaggregated data storage layer. During design metamorphosis of the HyperNode into a specific type of HyperNode, the HyperContract provides the HyperNode the codes it needs to extract the relevant information from the disaggregated data storage layer. For example, a [NS] HyperNode accesses the necessary node list to do routing. The [T] HyperNode extracts the required state-based concealment algorithms, and the [A] HyperNodes accesses propagation delays. Metaphorically, the process of HyperNode metamorphosis is similar to biological differentiation of human stem cells, cells that can change into nearly all types of specialized cell types. Cytological differentiation of stem cells is based on the environment used as a template to initiate differentiation. Similarly, a metamorphic HyperNode differentiates based on the template it receives from a HyperContract and the information it accesses from the disaggregated data storage layer. Other HyperNodes are used not to execute tasks, but to confirm transactions by functioning as blockchain observers in a jury of peers.

Another key feature of a metamorphic HyperNodes is “stateless” operation. Immediately after completing a task, the HyperNode forgets all the information or instructions it received, experiencing instantaneous amnesia and automatically reverting back to undifferentiated metamorphic HyperNode.

No Master Encryption Keys:

Because the HyperSphere comprises a meshed network with decentralized control, execution of concealment mechanisms occurs locally with master encryption

1 keys. As such, each [T] HyperNode receives decryption and encryption keys on a need-to-
2 know basis relating only to the next expected incoming data packet and the next outgoing
3 data packet. In other words, encryption and concealment occurs on a hop-by-hop basis
4 with no master decryption key available to decipher network traffic or its contents.
5 Operating as a dynamic directed acyclic graph (DyDAG) described later in this
6 application, packet routing involves every changing routes and security credentials,
7 meaning no HyperNode state is every repeated. Even in the unlikely event a single data
8 packet were to travel through the same node twice, the node's state and security
9 credentials will have changed. This features means any attempt to divert network packets
10 to a common server, will result in unreadable content.

11 No person, group, or corporation owns or controls the HyperSphere, its network,
12 or its operations. Instead, the HyperSphere functions as a non-profit decentralized
13 organization aggregating resources of its corporate, private, and research constituents.
14 Comprising an autonomous network of participating members with virtually no fixed
15 operating costs, merchants and service providers contract and pay resource providers on
16 an *as needed* basis, with the HyperSphere Foundation having no material interest in any
17 HyperSpheric transactions.

18 In this manner, the HyperSphere's network is made of its resource providers– a
19 heterogeneous community of devices earning income for their owners mutually interested
20 in protecting privacy for themselves and the HyperSphere's user base. Not to be confused
21 with decentralized applications, as a fully distributed network, packet routing and
22 network security are executed dynamically without central authority. Instead functions
23 are shared among nodes dynamically dividing the tasks of traffic management, packet
24 concealment algorithms and methods, and in the issuance of security credentials and
25 cryptographic keys. In fact, because network encryption and packet concealment is state
26 based, no master keys exist whatsoever. Instead dynamic security is 'state based'
27 occurring dynamically hop-by-hop as data packets traverse the HyperSpheric cloud.

30 C. HyperSphere Identity & Privacy Provisions

1 **Privacy Protection**

2 Although data packets traverse the network anonymously, HyperNodes represent
3 registered users, whose individual or corporate identity is validated and authenticated
4 whenever a HyperNode joins the network. By facilitating financial transaction
5 traceability, professional identity registration of HyperNode users dissuades criminals in
6 the illegal use of the HyperSphere to engage in crime, money laundering, trafficking,
7 racketeering, or terrorism. Adopting a policy that “identity inspires responsibility”, the
8 HyperSphere avoids the Internet’s problems of anonymity, overcoming the metaphorical
9 “payphone” problem by insuring a user’s identity can be traced to their financial
10 transactions even if the content of their communications remains private. Although user
11 registration identifies each HyperNode, stateless network operation still insures personal
12 and business privacy, where even the network operator is unable to monitor network
13 traffic and packet content.

14 Identity in the HyperSphere is confirmed using a digital certificate trust chain
15 comprising a digitally verified, i.e. “signed” digital CA-certificate. Such system
16 generated ‘network native’ CA-certificates are unique to the HyperSphere. The CA-
17 certificates are used in innumerable ways in the HyperSphere such as to confirm and
18 authenticate devices, HyperNodes, cryptocurrency wallets, software installation, and
19 perpetual blockchain transactions (chronicling asset debits and credits).

21 **Network Native Certificate Authority**

22 The HyperSphere acts as its own network-native certificate authority in
23 generating identity-trust-chains for its users and their devices. During account setup, the
24 HyperSphere first establishes a parental “identity” certificate as either a verified ‘true
25 identity’ owner or alternatively using a pseudonym. For the purpose of banking, asset
26 management, legal and business transactions, a user’s true identity must be established
27 through a know-your-client anti-money-laundering (KYC/AML) identity confirmation
28 procedure.

29 The account opening process in **FIG. 40**, establishes an irrevocable link between
30 a person’s identity and their personal identity-trust-chain of CA-certificates. In one
31 embodiment of this invention, personal identity documents 900 are translated into private

1 information using a cryptographic hash to produce hash ID 901 which is used to generate
2 a parental HyperSphere certificate authority, specifically 'account' CA 902, stored offline
3 for security reasons. Instead of employing an unknown 3rd party certificate authority to
4 issue the account CA 902, the HyperSphere provide system specific digital cryptographic
5 verification through system CA certificate 903. The system certificate 903 is stored
6 offline in the HyperSphere's diffuse cloud storage, meaning the system certificate 903 is
7 read accessible, but not at risk for corrupted or revised.

8 The system signed CA certificate 903 is then used to sign group certificate 904
9 which may include 2nd factor authentications. HyperSphere generated group CA
10 certificate 904 cannot be duplicated outside of the HyperSphere. By combining its digital
11 signature with hash ID 901, the parental account CA certificate 902 is issued. The
12 parental account CA certificate 902 cannot be counterfeited from outside the
13 HyperSphere. Any party trying to perpetrate or distribute a counterfeit from within the
14 HyperSphere will be detected and their actions traceable to their identity, their accounts,
15 and their assets. This fraud prevention feature is unique to HyperSphere generated trust
16 chains.

17 Thereafter, account certificate 902 is used to generate root certificates 905a and
18 905b. As shown in **FIG. 41**, offline root certificate 905b is then used to sign online CA-
19 certificate 906, which in turn is used to generate 'leaf' or 'issuing' certificates 907 (used to
20 sign HyperSphere blockchain records of assets), and multiple CA's collectively as 908n
21 used to verify and sign wallets, contracts, devices and installed software (such as
22 HyperNodes).

23 Regardless of whether an account is created using verifiable true identity or
24 pseudonymously, the account and its trust chain digitally signs (and therefore is
25 connected to) all hardware on which the HyperNodes reside. In true identity accounts, the
26 topmost personal CA-certificate, the 'parent' CA-certificate 902 is linked to identity
27 documents, e.g. passport, drivers license, social security number, etc. as evidenced by
28 image scans, biometrics, signatures, etc. The process of forming a trust chain by using
29 trusted verified certificate to sign antecedent certificates is illustrated in **FIG. 42** and
30 **FIG. 43**. When executed by a bank, qualified merchant, or by a trusted third-party agency
31 during account setup, the independent confirmation procedure confirms and corroborates

1 the legal identity of the person or corporation using multiple sources of ID validation
2 creating the account CA certificate 910. Once trusted identity is established, the account
3 owner is able to obtain a HyperSphere-issued ‘root’ certificate 911.

4 The approved root certificate enables its owner to make intermediate certificates
5 912 and leaf (end entity) certificates 913, useful for signing specific transactions or
6 authenticating specific devices. In this manner a person or their devices can engage in
7 commerce without revealing their personal identity or risk identity theft. As shown in
8 **FIG. 42**, the account holder’s identity certificate 910– their parental CA-certificate, is
9 used to generate a personal root certificate 911. The identity certificate 910 then signs the
10 account holder’s root certificate 911, which in turn is used to sign and authorize one or
11 more intermediate certificates 912 (IM CA-certificates) and ultimately leaf (end entity)
12 certificates 913. Both the identity certificate 910 and its root CA-certificate 911, once
13 used to sign subordinate certificates, can be placed in cold storage (i.e. offline or in a
14 diffuse cloud) as a backup in case its antecedent CA-certificates become corrupted. A
15 CA-certificate confirms ownership of a public key by the named subject of the certificate.
16 In the signing process, each certificate passes its public key to a subordinate, i.e. a would-
17 be issuer, which in turn encrypts confidential info using the public key and returns it to
18 the signing authority. Using its private key, the signing authority is able to decrypt the
19 file, proving it alone is the owner of the public key.

20 The authority then signs the issuer’s identity information with an encrypted
21 version of its private key and passes it back to the issuer. The certificate issuer can in turn
22 digitally sign subordinate certificates creating a chain-of-trust tracing back to the root and
23 parent CA-certificates. In the HyperSphere, while the controlling certificates involve the
24 identity of the account holder’s identity, the IM and leaf certificates may use
25 pseudonymous identities to further protect user privacy. In addition to protecting personal
26 privacy, CA-certificates also prevent fraud. All derivative CA-certificates sharing a
27 common lineage from a parental certificate are useful only to the parent certificate
28 owner’s accounts and devices. Even if an account’s login information is stolen, a thief
29 will not be able to match the pedigree of the account holder’s personal CA-certificate to
30 their devices and accounts. In the case where the account owner and its signer

1 cooperatively commit fraud, criminal investigation will invariably discover and expose
2 the conspiratorial relationships through the irrevocable identity-trust-chain.

3 In an alternative embodiment of the invention (shown in **FIG. 43**) uses a
4 combination of the group CA certificate 914 and the owner's public key 915 to generate a
5 verified intermediate certificate 916 used to generate end entity certificates 917.

6 As such, HyperSphere identity protects account security, transactional integrity,
7 and personal privacy while thwarting criminality. In the HyperSphere, users are able to
8 access identity-trust-chains to execute AAA verified transactions without added cost or
9 delays. The term AAA refers to a process of 'Authentication, Authorization, and
10 Administration' where (i) the certificate is first checked for a valid signature, (ii) the
11 corresponding transaction process is approved for the confirmed user, and finally (iii) all
12 relevant records are updated including, as applicable, appending new blocks onto a
13 blockchain.

14 Together, the unique combination of network-generated CA-certificates, identity-
15 validation, and digital signing using hypersecure public-key-infrastructure (PKI)
16 cryptography, establishes the HyperSphere as the pioneer of enterprise-grade CA-
17 certificates deployed natively over a public cloud. In contrast, enterprise-level CA-
18 certification over the Internet is both vulnerable and expensive, with costs of hundreds of
19 dollars per certificate not uncommon. And because the Internet is unable to confirm the
20 true origin of a CA-certificate, undetected Internet fraud is rampant with malware
21 infections at epidemic levels. With its pioneering deployment as a hypersecure 'privacy'
22 network, the HyperSphere protects personal identity and privacy by combining identity-
23 trust-chains and verified CA-certificate lineage with digitally signed authentication of
24 devices, HyperNodes, accounts, blockchains, transactions, and wallets. The privacy
25 network's protection provisions operate in a myriad of ways, including:

- 26
- 27 • Pseudonymous identity of user HyperNodes based on a disaggregated HyperSphere
28 name-server function, dynamically assigned to cryptographic identities via *ad hoc*
29 dynamic IP addresses and dynamic port numbers, to prevent account mapping.

- 1 • Personal network-generated CA-certificates for trusted dynamic signing of transactions
- 2 for devices, HyperNodes, and HyperWallets, thwarting imposter attacks, certificate
- 3 fraud, and cryptocurrency theft,
- 4 • Session based certificate exchange using personal CA-certificate to secure session
- 5 dialog and prevent eavesdropping.
- 6 • End-to-end encryption with identity-based private key exchange capability combined
- 7 with decentralized session-based certificates to ensure personal privacy independent of
- 8 SDNP cloud operations.
- 9 • Stateless HyperNodes not containing a record of calls, files, communiqués, or
- 10 cryptocurrency transactions on the device or HyperNodes (cloud portals) to prevent
- 11 forensic attacks and content reconstruction.
- 12 • A distributed network with fully decentralized control, where all transactions and data
- 13 routing use private keys with no master key or system authority, preventing usurpation
- 14 of the network electronically or through offline attacks of sysops or other personnel.
- 15 • Individually owned multi-tree blockchains with limited access for cryptocurrency
- 16 transactions and record keeping comprising dynamic directed acyclic graphs
- 17 (DyDAGs) eliminating the risk of privacy leakage through observer backtracing a
- 18 master blockchain.
- 19 • Transaction validation of replicant blockchain observer segments (RBOS), through a
- 20 decentralized cloaked (unidentifiable) jury-of-peers with limited blockchain provenance
- 21 access to prevent backtracing, fraud, blockchain attacks while insuring blockchain
- 22 transactional integrity.
- 23 • Root recovery capability using a newly disclosed device, the sequential quantum key or
- 24 (SQK) facilitating account restoration without exposing the identity-trust-chain to a
- 25 malefactor's usurpation by online cyberattack.

26

27 To prevent illicit attempts to generate fraudulent certificates outside the

28 HyperSphere, the network also facilitates system level certificate authority linking each

29 user account to a group using digital signature credentials impossible to imitate, as they

30 are network-native, generated and signed through system operations. As depicted in **FIG.**

31 **43**, for added privacy protections, intermediate CA-certificates can utilize multi-factor

1 authentication using dual signatures, one from the owner's root certificate 915, and a
2 second certificate from the system generated group certificate 914. In addition to
3 preventing fraud, the second authentication facilitates added protection against
4 conspiratorial malfeasance in business transactions. In any event, the HyperSphere is not
5 a good platform for criminals to practice their trade. Account information remains
6 indefinitely discoverable by law enforcement vis-à-vis authorized jurisdictions under
7 court order or by subpoena.

8 Likewise, while pseudonymous accounts are useful for engaging in legal
9 confidential business, because of identity based ownership, in the HyperSphere they do
10 not offer a conduit by which to subvert law or evade its agents. In the HyperSphere all
11 transfers from pseudonymous accounts to true identity accounts needed for banking are
12 recorded on the blockchains. Another element of the HyperSphere is its novel use of
13 topological trust networks. Although the foregoing methods rely on strong cryptographic
14 defense using network-generated CA-certificates and identity-trust-chains not subject to
15 fraud and theft, no system is immune to every attack. As such, the HyperSphere's
16 architecture employs *topological trust networks*, or 'trust layering', to limit the potential
17 damage of a successful intrusion into a person's accounts or devices— a security feature
18 operating much the same way that fire doors prevent the rapid spread of fire to limit
19 damage.

20 **FIG. 44** illustrates a hierarchical structure of the HyperSphere's topological trust
21 network. The most secure portion, the security core, contains an account holder's identity
22 based 'root certificate' 920. After it is used to generate 'intermediate' CA-certificates
23 (not shown), the root certificate 920 is stored offline in 'cold storage' 921, such as a bank
24 vault to prevent account usurpation. As a downside to its superior identity-based privacy
25 protection, damaged or lost root CA-certificates may become permanently unrecoverable.
26 To protect root CA-certificate privacy while insuring their recoverability, the
27 HyperSphere employs a new cryptographic key, the sequential quantum key 922 or
28 (SQK), introduced herein for the first time.

29 SQK employs methods adapted from quantum physics, namely the quantum
30 observer effect, the process whereby observing a system changes its state. This effect
31 includes quantum entanglement, the pairing of states where anything affecting one

particle's state also impacts its entangled pair too. Although a SQK may eventually be realized using quantum electron devices, in the HyperSphere its quantum behavior can be emulated using multidimensional software realized across the HyperSphere's stratified virtual-network layers. For example, in one embodiment a SQK key's implementation comprises a number of key segments (cells), each of which contains an ASCII alphanumeric character. The SQK key, comprising an encrypted version of a certificate access passcode includes both user selected and system-generated components. SQK decryption requires knowing an owner's passphrase and executing a read-write sequence in a precise order, i.e. reading, selecting and entering data into each segment in the proper order. Only when all the segments are viewed and modified in the proper read-write sequence will access to the root certificate recovery process be unlocked. Committing a single sequence misstep or entry error will result in a cascade of dead ends and meaningless challenge-response dialogs without revealing the entry has already failed. In this manner the misdirection consumes a hacker's CPU cycles wasting time, energy, and money. Without the proper read-write sequence, even knowing the passphrase is useless because the QSK is multidimensional, appearing at the HyperSphere user level as a cryptographic password having a different length than the passphrase. The missing pieces of the passphrase exist on at least three different virtual-network layers, appearing only if the proper sequence is entered. As such, the length of the SQK segment field length is variable, its appearance changing as entries are made or viewed. This variable key length feature makes it impossible for a cyber-attacker to guess how long of a passphrase they are looking for. For example, if the entry field has a constant length of 16 segments and each segment may constitute one of 37 alphanumeric characters (26 letters, 10 numbers, 1 null entry), the odds against a successful single-dimension brute-force incursion skyrocket well beyond 10^{25} -to-one per virtual-network dimension used. If the passcode entry segment-length varies, however, the odds against discovering a successful passphrase using brute force attacks increase exponentially. Operation of the SQK will be described in greater detail in a separate publication and in patent applications.

Returning to **FIG. 44**, beyond offline cold storage 921 (also known as air gap or DMZ security), the HyperSphere partitions its topological trust networks into three zones, namely a trusted network 923, a protected network 924, and an untrusted network 925. In

1 trusted network 923, the HyperSphere's network-native leaf CA-certificates 926a and
2 926b are used to sign all network-connected devices (such as notebook 927a, server
3 927b, and smartphone 927c) and all installed HyperNodes 928a through 928c. Even
4 though the same devices may interact with untrusted networks such as company
5 networks, university clouds, cyber café subnets, or the Internet 930, the HyperSphere's
6 symmetric sandboxing of HyperNodes prevents incursion or surveillance of the
7 HyperContract execution. A separate leaf certificate 926b is also used to sign an account
8 owner's trusted HyperWallet 931, holding cryptocurrency and other digital assets.
9 HyperWallet 931 does not, however, interact directly with the unprotected network of
10 online and POS transactions, mobile and other applications, users or independent digital
11 currency exchanges. Instead, all transactions are processed through a protected network
12 924 comprising a temporary wallet 932 separate from the user's personal trusted
13 HyperWallet 931. The temporary wallet 932 in turn executes transactions 935 with the
14 one-time transaction token, the OT³ proxy 933, preventing any access of vendors or users
15 to a HyperSphere account holder's HyperWallet 931 or their blockchains. As such, the
16 HyperSphere's built-in topological trust network protects both parties in a transaction 935
17 (including point-of -sale (POS), online, mobile, apps exchanges, and users) from fraud
18 and theft against third parties and also against one another.

19 20 **D. HyperSphere Blockchain Processing**

21 22 **Directed Acyclic Graphs (DAGs)**

23 Aside from the Internet's fundamental security flaws, cryptocurrency transactions
24 over the Web are made vulnerable by reliance on a single communal blockchain
25 accessible by everyone and anyone. In contrast, the HyperSphere eliminates the use of a
26 common public blockchain altogether, instead adapting multiple connected blockchains
27 having personal identity-based ownership. To ensure transactional integrity through peer
28 consensus, blockchain interconnectivity is facilitated using a multi-tree data structure best
29 described as a directed acyclic graph or DAG, also known as a 'digraph'.

30 The HyperSphere employs a novel variant of this DAG data structure used not
31 only in its cryptocurrency generation, payments, and transfers, but also in its network

operations, fragmented data transport, disaggregated data storage, and identity-trust-chains. To better understand how the DAG applies to HyperSpheric operations, we should first consider graph theory– the mathematical theory of the properties and applications of graphs. While in mathematics the term graph has several interpretations, in the broadest sense a graph is a collection of vertices and edges that join pairs of vertices. Applicable to a diverse range of disciplines including physics, biology, chemistry, electronics, computer science, communications, commerce, and more, graphs provide topological insight into connectivity, relationship, hierarchy, and processes. One class of topologies, ‘directed’ graphs, is particularly well suited in describing processes, flows, and algorithms containing sequence information. Shown in various forms in **FIG. 45**, directed graphs comprise graphs with vertices connected by edges employing vectors (arrows) signifying directionality. As shown, a graph 950 containing at least one graph cycle (a path of edges and vertices wherein a vertex is reachable from itself, i.e. a loop) is referred to a cyclic graph. In theoretical physics, an example of a cyclic process is a Carnot engine, a reversible isothermal gas expansion process (used to model the upper limit on the efficiency of thermodynamic engines converting heat into work). In each Carnot cycle temperature and entropy repeat the same loop, returning the system to its original state unchanged from the last cycle. In e-commerce, cyclic processes are problematic because they provide a means to change the past with no record of the change, affording the opportunity to commit undetected fraud and theft. For this reason, in accounting erroneous ledger entries cannot be changed, but instead must be amended as a new entry comprising a debit-credit pair recorded the date of the change.

Like traditional accounting ledgers, blockchains and DAGs comprise sequential records containing no ‘cycles’, meaning transactions proceed unidirectionally, never returning to the same vertex. Other examples of DAGs include ancestral family trees, epidemiological graphs of infectious disease spread from a single origin or index case, and computer malware diffusion, where each generation’s antecedents are followed by their own progeny, and so on.

While it has been suggested on the Web that a blockchain and a DAG are distinct concepts– that a DAG is a new construct intrinsically superior to blockchains, a more accurate description is that a blockchain is a one-dimensional DAG 951 comprising a

1 single tree. In other words, a blockchain is the degenerate form of a DAG, a chain
2 evolving in one dimension. DAGs can also exist in two dimensions: a blockchain with a
3 single sidechain 952 is the trivial case of a 2D DAG comprising a single common tree. In
4 graph vernacular, a tree comprises vertices connected to a common ancestor. It follows
5 that a 2D multi-tree DAG is simply a DAG containing multiple independent trees 953
6 containing both common and distinct vertices. Conceptually, the advantage of multi-tree
7 DAGs over a single-chain (1D DAG) blockchain is ‘parallelism’– the ability to divide
8 content and spread transactions across multiple ‘interconnected’ blockchains. Compared
9 to conventional blockchains, parallel processing offers the potential for improved
10 transactional efficiency, shorter chain lengths, lower storage demands, and faster
11 transaction processing. Converting a single communal blockchain into multiple
12 interconnected blockchains, albeit a step in the right direction, does not alone fix today’s
13 cryptocurrency issues of blockchain technology.

14 Cryptocurrency’s slavish reliance on nonce-hash puzzle solving remains
15 fundamentally energy and time inefficient, irrespective of transactional processing
16 efficiency improvements offered by DAGs. Furthermore, all cryptocurrency transactions
17 over the Internet remain vulnerable to security and trust attacks including blockchain
18 consensus exploits, privacy invasions, fraud, and cryptocurrency theft. Alone, converting
19 cryptocurrency to DAGs cannot (and will not) prevent crypto-wallet theft and blockchain
20 attacks. Only a holistic approach to security, privacy, and cryptoeconomic transactional
21 integrity can possibly hope to overcome the Internet’s ongoing epidemic of cyber-theft
22 and fraud.

23 As described previously, the HyperSphere is based on state-based communication
24 made in accordance with Secure Dynamic Network & Protocol technology, methods, and
25 apparatus. In graph theory, this means that each time a HyperNode performs a task or
26 executes a transaction the vertex’s state at that moment is distinct in space and time. As
27 such, the HyperSphere uniquely comprises a *spatiotemporal* network. Accordingly, all
28 transactions in the HyperSphere are dynamic and state-dependent, constantly changing in
29 accordance with time and location. In order to adapt the features of multi-tree DAGs to
30 operate on a SDNP based dynamic spatiotemporal network, the HyperSphere employs a
31 new graph topology, the *dynamic directed acyclic graph* or DyDAG, introduced here for

1 the first time. In a DyDAG topology, vertices are defined by two characteristics– identity
2 (vertex name or number), and state, symbolically as vertex v_x and state s_y . A state is the
3 condition defining the rules by which the vertex operates and interacts with other
4 vertices. In the HyperSphere, a vertex’s state includes time, its resident security zone, and
5 other location information. As such, revisiting the same vertex does not constitute a
6 cyclic loop so long that the state is different.

7 For example, in the Carnot cycle each repeated loop is cyclic because whenever
8 the system returns to a vertex, the state of the vertex is exactly the same as the prior
9 cycle. In contrast, time travel in the sci-fi blockbuster “Back to the Future– Part II” is an
10 example of acyclic spatiotemporal behavior. In the story, the characters Doc Brown and
11 Marty McFly travel to the future in their DeLorean time machine fully expecting nothing
12 to change during their travels. Upon their return to their time and place of origin, to their
13 chagrin they discover everything to be horrifically unfamiliar. Metaphorically, even
14 though the vertex v_x was the same as before, the state s_y had changed unexpectedly. In
15 this sense, the HyperSphere intentionally changes states constantly and in unexpected
16 ways, confusing and confounding any cyber-hacker trying to discern a pattern. As
17 graphically represented in **FIG. 46**, a DyDAG sequence 953 comprising a transaction
18 from (v_1, s_1) to (v_2, s_2) to (v_1, s_2) does not constitute a cyclic graph so long that the states
19 $s_1 \neq s_2$. Although in a two-dimensional planar projection 954 of a three-dimensional
20 DyDAG the graph appears cyclic, in 3D the graph illustrated as a helix or spiral 955
21 clearly shows the state-space is not cyclic or closed loop. In essence, the network
22 autonomously exhibits a sequence of irrevocable changes with such rapidity it defies
23 analysis. By including a state variable, a 2D multi-tree DAG becomes a 3D DyDAG
24 blockchain of superior performance, integrity, and security. The HyperSphere applies this
25 DyDAG principle is a number of ways in the HyperSphere including:

- 26
- 27 • Hypersecure SDNP communication using state-based security credential
- 28 and algorithms.
- 29 • Data transport over a distributed network of active and redundant
- 30 HyperNodes minimizing propagation delay while improving network resiliency
- 31 (described below).

- 1 • Personal CA-certificate based identity-trust-chains used to chronicle the
2 inclusion or revocation of HyperSphere connected devices and HyperNodes,
3 thereby controlling access and privileges.
- 4 • Personal CA-certificate based identity-trust-chains used to sign and
5 manage perpetual DyDAG blockchain transactions, RBOS observers, and OT³
6 proxy payment processors.
- 7 • Personal CA-certificate based identity-trust-chains used to sign and
8 manage perpetual DyDAG data in HyperWallets.
- 9 • Transitory DyDAG blockchains ('tBC') used in HyperContract job
10 execution of contract pledges, task execution, juror consensus, and HyperCoin
11 generation through minting or by melting and recycling (re-minting).

12
13 In regards to the use of a DyDAGs in network operation, the SDNP cloud
14 intrinsically forms a dynamic DAG comprising four tiers of HyperNode resource
15 providers. Nodes are added to the network based on the number of prospective minters
16 wishing to earn cryptocurrency and by automatic node instantiation in the event of local
17 network congestion or DoS attacks. In each instance, the more HyperNodes joining the
18 network the more redundancy the cloud exhibits and the more efficient the network
19 becomes in finding and using the shortest propagation delay paths for data routing. In
20 graph theory vernacular, SDNP network operation represents spatiotemporal destination-
21 oriented directed dynamic directed acyclic graphs for meshed and multipath routing.
22 Another benefit scaling with nodal density is resiliency, the ability of the network to
23 maintain an acceptable level of QoS (quality of service) while surviving
24 misconfigurations, faults, power failures, natural disasters, and attacks. The resilience of
25 a network scales non-linearly with the number of participating nodes. While theoretically
26 the total number of combinational connections scales with the number of nodes n by the
27 relation $n \cdot (n-1)/2$ approaching n^2 for a large number of nodes, many of the connections
28 are excluded as cyclic. Although in a DyDAG reusing a node isn't truly cyclic (because
29 the states differ), in real time networks only short hop-counts are valuable in delivering
30 low propagation delays and are also beneficial in achieving low routing power, especially
31 important in mobile Last Mile connections. A representative model of a DAG describes

the number of combinations “a” by a recurrence equation comprising k outflows (exiting edges) over ‘n’ nodes.

$$a_n = \sum_{k=1}^n (-1)^{k-1} \binom{n}{k} 2^{(n-k)} a_{n-k}$$

In the binomial expression shown, the number of available paths rises proportionally with the population of participating nodes in the network. Although the expression is more realistic than the idealized permutation population $n \cdot (n-1)/2$, it does not embody certain features of dynamic DAGs. For example, cyclic loops excluded in a DAG may not be cyclic in a DyDAG because of state changes. Conversely, in nodes executing transactions at extremely high transaction rates, some nodes may (at least for brief intervals) behave as elements of a static DAG, whereby a number of possible loops (outflows) should be excluded from the tree population as cyclic. And although while *mathematically* speaking distant remote nodes comprise valid DyDAG trees, in real time networks they must be excluded for their unacceptably long propagation delays (meaning they are so far away from the callers their participation in the network doesn’t help transport at all). In other words, in a spatiotemporal DyDAG graph, the trees must be excluded.

Multi-tree DyDAG Blockchains

To ensure the security, integrity, and speed of cryptocurrency generation and blockchain transactions, a blockchain must be limited in size and length, and therefore involve limited membership to avoid uncontrolled growth and prevent intrusion from unknown users. Existing blockchain technology used by Bitcoin, Ethereum, etc. employs a single public ‘communal’ blockchain with global permissionless participation.

Communal permissionless blockchains are also subject to privacy leakage, theft, content contamination, and illegality. The resulting public blockchain is too cumbersome, slow, and vulnerable to attack to meet the HyperSphere’s design goals and operational objectives. To circumvent long blockchain weaknesses and vulnerabilities, the HyperSphere employs a completely new blockchain structure and control system for blockchain processing, cryptocurrency transactions, and traffic management– the

dynamic directed acyclic graph, or ‘DyDAG’, developed and introduced here for the first time. Adapted for dynamic real-time processes from static graph theory, DyDAG mathematics, graph theory, and control algorithms are employed extensively throughout HyperSpheric operations including governance of dynamic meshed data routing, HyperContract execution, rapid blockchain transactions, HyperSphere cryptocurrency generation, and e-commerce.

Contrasted against conventional single-chain ledgers, DyDAG blockchains shown in **FIG. 41** are personalized and multi-tree, thereby limiting blockchain length, reducing storage demands, and accelerating transaction resolution rates. Beyond these obvious performance benefits, DyDAG blockchains are robust, ensuring tamper-proof consensus for transaction validation.

Unlike the global communal permissionless single-chain blockchain in conventional cryptocurrency, the various trees in the HyperSphere’s DyDAG blockchains are ‘individual’ (not communal), with each blockchain having personal or enterprise ownership through an identity-trust-chain. DyDAG blockchains include both transitory blockchains or tBC, i.e. limited-life, ledgers used for contract execution, and perpetual (i.e. permanent) blockchains BC used to immutably record financial transactions and enshrine legal records. Like unitary blockchain implementations, all transactions on DyDAG blockchains are time stamped, immutably chronicling a record of sequential transactions not subject to backdating and revision. Unlike communal unitary public blockchains, however, since each DyDAG blockchain tree is personalized and owned by different individual or corporate entities, a mechanism is required to interlink transacting blockchains and entities.

As depicted in **FIG. 49**, this link is realized by adapting the dual-column credit-debit ledger concept of general accounting for blockchains, where every credit corresponds to a debit on another blockchain. In the HyperSphere, all blockchain-to-blockchain asset transfers are executed through HyperContracts specifying the participants including buyers, sellers, jurors, and alternate jurors. At contract completion all credit-debit transactions are recorded and time stamped as debits on the payor’s DyDAG blockchain and as credits on the payee’s private blockchain. In the case of public blockchains, the modified DyDAGs are then published on the HyperSphere using

pseudonyms to protect the owner's true identity from hackers and thieves. Although these pseudonyms do not reveal an owner's true individual or corporate identity, in criminal investigations or in cases of civil litigation, a pseudonymous blockchain owner is traceable to their true identity. The HyperSphere is also capable of supporting private blockchains. Unless a buyer waives built-in protective provisions, tokens recorded on private blockchains are not directly transferable to the HyperSphere's cryptocurrency. Instead such tokens should be exchanged through a bank or independent digital currency exchange into fiat currency subsequently used to purchase either HyperMetal or HyperCoins.

E. HyperSphere Cryptoeconomic Platform

The HyperSphere operates as a fully distributed network and ecosphere for decentralized e-commerce. Transactions between HyperSphere merchants and resource providers occur on a peer-to-peer basis with no central authority, after which compensation, i.e. rewards, transfer directly between the parties. Published block chain ledgers of completed transactions are verifiable by a jury of peers within the HyperSphere. Customers of HyperSphere merchants vary with the services a merchant offers. Broadly, HyperSphere merchant services include HyperSecure cloud communication, cloud computing, disaggregated cloud data storage, network connected devices, and a myriad of cloud-based e-services.

As a Cloud-as-a-Service (CaaS) provider, the HyperSphere facilitates a hypersecure platform for supporting a wide range of commercial and scientific efforts. Unlike commercial CaaS providers, however, the HyperSphere operates as a platform at arms length from the transactions its hosts, and (aside from small routing fees) is not a party to any services, transactions, or businesses using the HyperSphere. In short, the HyperSphere does not compete with its users or its resource providers. As such the HyperSphere is more accurately described as a Platform-as-a-Service provider.

HyperSphere Platform-as-a-Service

1 As a non-profit independent Platform-as-a-Service, the HyperSphere is able to
2 support a diverse range of economic, commercial, scientific, and philanthropic endeavors
3 without conflict. Limited only by merchant offerings and participant imagination, the
4 HyperSphere is therefore able to serve virtually every sector of business and study,
5 including banking and finance, manufacturing, marketing, merchandizing and selling,
6 distribution, health care and medicine, energy and ecology, transportation and shipping,
7 safety and security, education, research, development, science, and information storage,
8 to name but a few examples. The HyperSphere offers a unique approach e-commerce
9 supporting a diverse range of the hottest high-tech business topics today, including:

- 11 • Cloud computing & real-time communication
- 12 • Big data
- 13 • Artificial intelligence (AI)
- 14 • Security and privacy
- 15 • Decentralized digital currencies
- 16 • Energy efficiency

17
18 Services enabled by the HyperSphere platform for the foregoing efforts are
19 described in greater detail here below:

20
21 ***Cloud Computing and Real-Time Communication:***

22 The promise of cloud computing and cloud-based communication is plagued by
23 its reliance of the Internet for connectivity. From its very inception, the Internet was
24 created for reliable redundant delivery of files, but never as a real time network. As such
25 cloud-based communication today including Line, KakaoTalk, WhatsApp and others are
26 often unreliable, dropping calls and suffering delays and “network instability,” a
27 euphemism for an unacceptably long propagation delay of data traversing a packet
28 switched network. This same instability makes distributed computing extremely
29 problematic because network delays of hundreds of milliseconds can be a billion times
30 slower than computer calculations, meaning a server spends all its time waiting to
31 reestablish network connectivity and transfer updated data files. The HyperSphere

addresses these issues by employing a patented real time dynamic network and protocol where expediency is realized using small data packets delivered over the lowest propagation delay pathways through the network.

Big Data:

The analysis or storage of large data sets means that a single drive or memory farm may contain an extraordinary amount of personal and private data in one place. A successful attack or unauthorized incursion on these massive records exposes the public to fraudulent transactions, identity theft, blackmail, extortion, or even personal attack. The number of successful hacks on government, credit card, credit bureau, insurance, and merchant databases in recent years has exposed hundreds of millions of people to personal and financial attacks from stolen data. This problem will only worsen with today's migration to big data file structures and more online and personal cloud storage, e.g. with iCloud, Amazon Drive, Google Drive, iDrive, Box.com, and Dropbox. The HyperSphere combats storage-drive attacks using disaggregated data storage – redundantly distributing file contents across hundreds of devices. This approach makes it impossible for hackers to locate and gather all the components of a file or to re-assemble the pieces to restore the original content.

Artificial Intelligence:

Artificial intelligence is a rapidly evolving but still underutilized discipline in computer science today. Without amassing significant compute power, progress in this field is expected to proceed slowly. Artificial intelligence plays two roles in the HyperSphere. Firstly, as an enabling technology, AI is employed in the HyperSphere marketplace to autonomously negotiate job contracts between HyperSphere merchants and HyperSphere resource providers. In this manner, AI eliminates the central authority concern in the HyperSphere performing scheduling (like a dispatcher) but not acting as a party to any transaction or its validation. Secondly, with its distributed computing resources and open source architecture, we expect the HyperSphere will make an attractive cloud platform for research in AI technology.

Security and Privacy:

The world's platform for international business and e-commerce, the Internet, is intrinsically un-secure. A limitless list of cyber-attacks have been recorded since the advent of Internet based communication and e-commerce including denial-of-service attacks, worms, viruses, spyware, packet redirection, faux cell towers, keystroke loggers, packet sniffers, zero-day attacks, port sniffers, and other malware. Moreover, because data packets identify information about a packet's origin and destination, metadata analysis can be used for user profiling based attacks. The HyperSphere in contrast uses anonymous fragmented data transport with dynamic routing. With packet content, security credentials, and routing dynamic changing every fraction of a second, tracking network transport and capturing related packets in the HyperSphere, decrypting them and extracting their payload, then reassembling their original content is virtually impossible to successfully execute (even once). To hack an entire communication requires constantly monitoring every node in the HyperSphere and performing the aforementioned hacking sequence at a pace faster than new packets change. As such, the HyperSphere's dynamic security methods render the network impervious to man-in-the middle attacks. So unlike purely encryption-based Internet communication, HyperSphere communication is HyperSecure lacking any central control or system encryption key.

Decentralized Digital Currencies:

Existing cryptocurrencies relentlessly face rapid value fluctuations from speculation affecting a coin holder's buying power. Bitcoin price fluctuations of 27% have occurred in a single day of trading. Further cryptocurrency concerns include the rising expense and longer times required for coin mining (the so-called "Tragedy of Commons"), hidden Ponzi schemes & elaborate pyramid scams, unreliable exchanges, hacking of Internet based communication used for coin transactions, suspicious exits of ICO founders, insider trading, securities fraud, identity theft, money laundering, drug trafficking, and other substantive concerns regarding security, ethics, digital currency integrity and value. As such, cryptocurrencies today are dubious financial instruments having little or no role in the daily life of consumers, businesses, banking, or international commerce. The HyperSphere employs tokens generated autonomously and conjunctively

as part of data transport through the network. Since tokens are generated based of Proof-of-Use, whenever a Resource Provider completes a task or job, the process of minting is not subject to Ponzi schemes, 51% attacks, or the Tragedy of Commons.

Energy Efficiency:

The mining of present-day cryptocurrencies involves the wasteful expenditure of vast quantities of energy (now over 0.15% of the world's entire annual electrical consumption) to perform computational tasks having little or no beneficial purpose aside from generating new crypto-coins. The HyperSphere employs tokens generated autonomously and conjunctively as part of data transport through the network, i.e. doing useful work. This method of token generation consumes twelve-orders-of-magnitude less energy than Proof-of-Work based mining used in Bitcoin generation.

HyperSphere Design Architecture

The design objective of the HyperSphere is to facilitate an open source platform for e-commerce supporting a global community of users while both protecting user privacy and ensuring transactional integrity. To that end, the HyperSphere's design methodology is based on five fundamental precepts comprising the attributes of...

- Identity
- Security
- Privacy
- Integrity
- Responsibility

As a computer network and communication cloud adhering to these core principles, the HyperSphere's design offers vastly superior operational command and control compared to other networks and clouds. The Internet, by its very nature, relinquishes control to unknown devices connected to it. Internet-connected devices determine packet routing, the security methods employed (or ignored) in data transport, and even who can access or surveil a packet's contents or metadata. As such, any bad

actor can through a variety of means subject other users to theft, privacy invasion, and other malefactions without consequence, all protected by anonymity of the cloud.

Metaphorically speaking, in this regard the Internet operates as a ‘payphone’, meaning anyone can communicate anonymously without revealing personal identity information to the network or to other users. Worse yet, with no ability to confirm identity or confidently establish trust, imposters can with relative ease use the Internet to usurp another’s user’s identity without detection. In many cases, Internet attacks can be launched from IoT devices– the least secure components in a network. In this way, a refrigerator, smart TV, thermostat, or dimmable ‘smart’ light bulb can compromise the integrity and security of an entire network and its users, becoming the attack vector of choice for discerning cybercriminals. The HyperSphere, by contrast, explicitly controls network access by identifying and authorizing every user and attached component. Through software-based network portals called *HyperNodes*, the HyperSphere manages process and call initiation, controls the handling of different data types (voice, text, video, software...), directs data packet routing, selects security concealment algorithms and security credentials, and validates processes.

It also carefully scrutinizes embedded cryptocurrency transactions, manages network operation to ensure high quality-of-service (QoS), and carefully verifies connected device and user identities. In the HyperSphere, security and privacy are addressed through separate mechanisms. Rather than augment Internet communication, HyperSpheric security is achieved by utilizing its own dedicated communication protocol– the Secure Dynamic Network & Protocol or SDNP. As such, the HyperSphere is not subject to traditional Internet security vulnerabilities and deficiencies.

Previously deployed over private networks for municipalities and emergency services in Germany, the UAE, and by various shipping port authorities, SDNP communication operates using proven field-tested technology with over fifteen years experience in professional communication and private networks. Its use by the US Army during the Iraq War confirmed the protocol’s capability of delivering military-grade security over private radiophonic networks compliant with FIPS140-2 standards. The HyperSphere’s design objectives as described here represent a public network open source deployment of that same technology combined with enterprise-grade certificate

1 authority and embedded network-native cryptocurrency. A brief overview of these
2 objectives and how the HyperSphere addresses the issues, follows here below:

3 4 ***HyperSphere Identity:***

5 In contrast to the Internet, in the HyperSphere no user is anonymous– every user,
6 personal or corporate, holds a corresponding unique HyperSphere identity, privately
7 protected from other users’ inspection. This personal or corporate HyperSphere identity
8 permanently interlinks a user’s devices, HyperNode cloud portals, accounts, and wallets
9 to an *identity-trust-chain* comprising HyperSphere network-generated CA-certificates.
10 The Internet depends on third-party certificate authorities subject to theft and fraud. In
11 contrast, the HyperSphere generates its own *network-native* CA-certificates. As shown
12 previously in **FIG. 40**, all identity-trust-chains exclusively employ CA-certificates signed
13 by the HyperSphere’s master certificate 903, rejecting all self-signed or third party
14 certificates as untrusted. By interlinking a user’s CA-certificates to a corresponding
15 identity-trust-chain, stolen or fraudulent certificates will not match other instances of the
16 user’s CA-certificates, and the fraud will be detected, rejecting all transactions involving
17 the fraudulent certificate.

18 19 ***HyperSphere Security:***

20 While the HyperSphere’s enterprise-grade certificate authority for identity
21 verification is important, alone it is inadequate to prevent network incursions. In order to
22 protect data, maintain transactional integrity, and prevent cryptocurrency theft or fraud,
23 the HyperSphere employs military-grade ‘hypersecure’ data transport and multi-tiered
24 security features made in accordance with its patented Secure Dynamic Network &
25 Protocol (SDNP). Although encryption is employed in packet transport operations, the
26 SDNP process does not depend exclusively on encryption to achieve its superior security
27 protection. Instead, hypersecure communication combines the principles of *fragmented*
28 *transport of anonymous data packets* together with *dynamic routing and concealment*. In
29 accordance with its protocol, SDNP data transport in the HyperSphere is secured by (i)
30 limiting the quantity of data traveling through any single node in the network, (ii)
31 obfuscating the packet’s true origin and destination, (iii) concealing the content of data

1 packets, and (iv) limiting the time in which to break the security provisions and launch an
2 attack before everything changes (e.g. new security credentials, algorithms, packet
3 routing, content, and more).

4 The last described security method of limiting ‘time’, more accurately described
5 as *dynamic routing and concealment*, is especially frustrating and costly to
6 cybercriminals because it constrains the useful duration of any successful hack to a mere
7 fraction of a second, after which the attackers must start all over again. Changes in
8 routing and concealment methods change perpetually, meaning even in the unlikely event
9 a cyberattack breaks into a packet, they will be unable to ascertain where the next
10 successive packet is or how it is being routed. In the HyperSphere’s meshed network, it is
11 unlikely two successive packets will ever traverse the same nodes. And since SDNP data
12 packets carry fragmented data, even if an attacker is able to break a packet’s cipher
13 (requiring the perfect execution of a century worth of brute force decryption in one tenth
14 of second), without the other corresponding pieces a decrypted packet’s fragmented
15 contents are incomplete, meaningless, and utterly useless, discouraging further attacks on
16 the HyperSphere’s cloud and network traffic.

18 ***HyperSphere Privacy:***

19 Privacy is the right to control what information you share and with whom you
20 share it. A secure network does not automatically guarantee privacy—ensuring privacy is
21 more stringent and demanding than simply facilitating security. As such, the
22 HyperSphere does not rely solely on its SDNP secure network capability to guarantee
23 private communiqués and files remain so. Instead, a *privacy network* must, in addition to
24 preventing hacking and surveillance, control access to personal content and private
25 information on a need-to-know basis utilizing ‘verifiable identity’ to limit access.

26 Authorization by verifiable identity is especially critical in preventing imposters
27 from capitalizing on anonymity to obfuscate their true identity, misrepresent their
28 purpose, or secretly engage in malicious attacks against a person or enterprise. In order to
29 function as a privacy network, the HyperSphere utilizes the principle of confirming user
30 and device identities during the connection process, i.e. using network-native CA-
31 certificates to establish trust of persons or devices *before granting user access to*

1 privileged information. Beyond hypersecurity, the HyperSphere's privacy provisions
2 protect personal identity and private information through a sophisticated combination of
3 identity-trust-chains and verified CA-certificate lineage not possible over the Internet.
4 Shown previously, these safeguards include digitally signed authentication of devices,
5 HyperNodes, accounts, blockchains (BCs), HyperContract transactions, and wallets,
6 employing issuing (leaf) certificates distinct from its intermediate IM parents. Beyond a
7 strong cryptographic defense, the foregoing methods uniquely employ HyperSpheric
8 network-native CA-certificates and identity-trust-chains not subject to counterfeiting.
9 One downside to identity-based privacy protection is that, without some means of
10 backup, damaged or lost root CA-certificates may become permanently unrecoverable– a
11 problem which the HyperSphere addresses with an innovative solution, the Quantum
12 Sequential Key or QSK, described later in this manuscript.

14 ***HyperSphere Transactional Integrity:***

15 As a privacy network for hypersecure global e-commerce, transactional integrity
16 depends on secure network operation, user authentication, identity-trust-chains, assured
17 HyperContract execution, and verifiable cryptocurrency transactions. Ensuring
18 transaction integrity in the HyperSphere involves several important mechanisms
19 including (i) preventing the creation of fraudulent (fake) cryptocurrency, (ii) preventing
20 blockchain attacks intended to perpetrate double spending and theft, (iii) avoiding
21 destabilization of cryptocurrency value impacting the HyperSphere's utility and
22 cryptoeconomics, and (iv) ensuring expedient transactional processing and resolution.

24 ***HyperSphere Responsibility:***

25 The final consideration of the HyperSphere is its principled dedication to personal
26 privacy, fiscal, ethical, and ecological *responsibility*. As a fully decentralized network
27 using fragmented data transport with no network cryptographic master keys,
28 HyperSphere operation naturally protects its user's confidentiality and personal privacy.
29 Because of its dynamic meshed transport, privacy attacks using packet sniffing,
30 surveillance, and metadata monitoring are completely unproductive.

1 In the HyperSphere, a user, not the network, owns their private data. Unless a user
2 grants rights to a service provider to access or distribute it, merchants have no capability
3 to obtain, know, share (or steal), a HyperSphere client's personal information. Moreover,
4 by using pseudonymous leaf CA-certificates, clients can engage in e-commerce without
5 revealing any personal data whatsoever or risking identity theft. Combining identity-
6 based CA-certificates with advanced multifactor and biometric authentication, a user's
7 accounts, blockchain, wallet, and personal data are not subject to inspection, data
8 collection, attack, or usurpation.

9 Although the HyperSphere protects personal privacy in the lawful use of the
10 network, the HyperSphere's inventors and proponents condemn all acts of criminality,
11 financial and business fraud, privacy attacks, theft, and terrorism. As an ethical
12 communication network, the HyperSphere supports law enforcement in accordance with
13 legal jurisdictions of the session's terminus HyperNodes, i.e. wherever a transaction
14 between parties originates or terminates. Because of fragmented data transport across a
15 meshed network and stateless node operation, no useful content or metadata is available
16 except on the terminus nodes.

17 Environmentally, the HyperSphere represents the world's first and most eco-
18 friendly method of cryptocurrency generation. Unlike PoW cryptocurrencies wasting vast
19 amounts of energy, consuming precious resources, and exhibiting large carbon footprints
20 only to solve useless puzzles and games, the HyperSphere's cryptocurrency is highly
21 energy efficient, using data transport through its network as a symbiotic mechanism to
22 generate new cryptocurrency. As such, the HyperSphere's Proof-of-Performance
23 adjunctive synthesis and lightweight blockchains consume one-trillionth (10^{-12}) the
24 energy of Proof-of-Work cryptocurrencies such as Bitcoin, Ethereum, and their sidechain
25 derivatives. In comparison to existing and hypothetical token and cryptocurrency
26 generation schemes, the HyperSphere's adjunctive method of minting HyperCoins
27 represents the world's first ecologically friendly and environmentally sustainable
28 cryptocurrency. As a final point, the potential of the HyperSphere is not limited to
29 commercial and personal profit-minded projects, but extends to all socioeconomic
30 groups. For example, the HyperSphere can be adapted to support research, to facilitate
31 funding of a new generation of entrepreneurs, and to facilitate a variety of charitable and

philanthropic projects, including its potential role in will and trust execution and estate planning.

Architectural Summary:

In summary "The HyperSphere" disclosed herein comprises a new *dual digital token based* decentralized global electronic marketplace for e-commerce that overcomes issues of digital currency integrity and instability, network security, real time performance, and energy inefficiency. Running over a global hypersecure private network – a cloud separate and distinct from the Internet yet co-existing on the same hardware, the HyperSphere delivers eco-friendly cloud-based computing, data storage, real-time communication, secure network-connected devices, and e-services to any user within the HyperSphere.

HyperSphere Beneficial Features

As described in its provisional patent, the HyperSphere is an open-source hybrid-cloud platform amalgamating the global functionality of the Internet with the best features of premier professional communication, private networks, VPNs (virtual private networks), dynamic real-time networks, global telephony, military-grade cybersecurity, enterprise-grade certificate authority, trusted transactions, intrinsic privacy protections, and private blockchains. The HyperSphere is wholly unique in its novel method of real-time data routing, traffic management, cryptocurrency generation, and blockchain transactional execution. During operation, tasks are performed autonomously and adjunctively, unassisted by network operators. Routing occurs dynamically based on network conditions without relying on pre-defined (static) routing tables. Instead, the HyperSphere represents a fully decentralized system employing dynamic meshed routing designed to minimize network propagation delay to securely and rapidly execute transactions.

Combining beneficial features of high-reliability fixed and backbone networks, dark-fiber and backhaul, wireless, and *ad-hoc* peer-to-peer communication, with an AI-based de-centralized marketplace, the HyperSphere dynamically analyzes and ascertains the best match between network performance and a client's performance and cost

objectives. Because the network's nodal density increases with its number of users, 'the more people who use the HyperSphere– the better it performs', quite the contrary of fixed network clouds.

The HyperSphere is especially unique in its generation and use of its network-native (embedded) cryptocurrency. In the Internet, conventional cryptocurrency is “mined” using costly and energy-wasting Proof-of-Work puzzle solving such as nonce-hash or prime number challenges with uncertain payment and ever-diminishing fiscal returns to its miners. In stark contrast, cryptocurrency generation in the HyperSphere is “minted”, created adjunctively as data packets traverse the cloud as shown in **FIG. 47**. Unlike the uncertain return of PoW miners, in minting HyperSphere resource providers receive *guaranteed compensation for supporting completed transactions*, paid in accordance with pre-negotiated HyperContracts 964 specified by service providers or merchants 958 providing services for their clients 959). Because the coin generation occurs adjunctively with network operation, virtually no additional energy is spent on minting cryptocurrency beyond the energy spent completing useful work needed for communicating or computing tasks. Other than being energy efficient and ecologically responsible, dynamically generating network-native blockchains by data transport in the cloud prevents counterfeiting. Producing cryptocurrency using dynamic blockchain synthesis comprises a process of inter-nodal data transport that cannot be imitated outside of the HyperSphere. And because the cryptocurrency is network native, it can be transferred and retained in HyperWallets and reused in the HyperSphere without exposing blockchains to the Internet's hacking, theft, fraud, and online transaction risks.

HyperSphere access is entirely software-based with no need for specialized hardware. User interfaces for smartphones, notebooks, desktop PCs, gaming platforms, smart TVs, IoT etc. include support for major operating systems including Windows, MacOS, Linux, Unix, iOS, and Android. Businesses, corporations, research institutes, and universities can facilitate HyperSphere access to their private servers and networks via personal devices, i.e. enabling convenient and cost-effective Bring-Your-Own-Devices (BYOD) connectivity, while supporting corporate IT department security provisions and control. A cluster of devices can also operate as a private network within

the HyperSphere, i.e. as a publically hosted private-network. HyperSphere users may engage in transactions in several ways including in the roles of:

- *Resource providers*– By downloading HyperNode portal software into one or more devices, individuals, companies, and institutions provide resources to the HyperSphere and earn HyperCoin cryptocurrency as compensation.
- *Merchants & service providers*– By creating a HyperSphere API-generated application or user interface, merchants and service providers 958 can offer communication, computing, storage, cloud-connected devices, or e-services and products to their clients (even if their customers are not HyperSphere clients).
- *Users*– As clients of merchants and service providers 958, users 959 can utilize the resources of the HyperSphere, paying in fiat currency or using earned or commercially acquired HyperCoin cryptocurrency.

Hierarchically, rather than employing software running on the Internet as over-the-top (OTT) applications, the HyperSphere co-exists with the Internet, sharing resources, physical networks, last mile carriers, and data links. In this sense the HyperSphere essentially operates “on-the-side” (OTS) of the Internet representing partially overlapping peer networks. Furthermore, the HyperSphere is agnostic to last mile connectivity between the cloud and a user’s device, seamlessly compatible with any medium including WiFi, Ethernet, DOCSIS-3, wireless (3G/LTE, 4G, 5G), etc. Aside from its superior security and its embedded native cryptocurrency, as a ‘privacy-network’ the HyperSphere uniquely employs network-specific pseudonymous identities to protect personal account information. Using digitally signed CA-certificates to privately execute transactions, open HyperContracts, deliver network resources, or trade cryptocurrency, HyperSphere users are thus able to engage in e-commerce without exposing their true identity to potential attacks. As a further precautionary feature, consensus verification of blockchain transactions employs a unique innovation– a replicant blockchain observer segment (RBOS) e.g., RBOS 1063 shown in **FIG. 57**, a limited length blockchain mirror used to validate transactions while preventing blockchain backtracing and privacy leakage. Another inventive element, a one-time-transaction token (OT³) 1041 shown in

FIG. 56, employs a single-use temporary transactional payment mechanism to prevent a payee's third-party transaction processor from gleaning private information from a payor's blockchain. In e-commerce, the HyperSphere offers numerous benefits over the Internet including:

- The ability to anonymously, securely, and privately transport real-time audio and video content: functionality needed by service providers offering communication and secure messenger services.
- The ability to anonymously, securely, and privately transport high-integrity data files including email; databases; private media content; and software: functionality needed by providers of secure email, database services, customer contact management, and online collaboration platforms.
- The ability to anonymously, securely, and privately dispatch; manage; and collate the execution of distributed cloud computing supporting researchers and online cloud computing providers.
- The ability to anonymously, securely, and privately transport, store and recall data in disaggregated form, functionality needed for big data analysis and by purveyors of online and cloud storage services.
- The ability to anonymously, securely, and privately transport command-and-control (C&C) instructions for cloud-connected devices while preventing security and privacy attacks on cloud connected devices: functionality and privacy features important to IoT device users and service providers.
- The ability to securely and pseudonymously execute financial transactions, payments or money wires using cryptocurrency intermediaries comprising network-native dynamic blockchains.
- The ability to anonymously, securely, and privately execute a wide variety of e-services for merchants.
- The ability to facilitate the use of pseudonymous data to facilitate personalized AI-based recommendations without revealing a user's true identity or enabling the unauthorized access or sale of personal or private information. In the HyperSphere, a user owns their personal data, not the merchant or the network.

- The ability to form a merchant-operated hypersecure private overlay network securely deployed *within* the public HyperSphere cloud, i.e. using fully sandboxed processing to protect corporate and personal privacy and data integrity.
- The ability to dynamically tunnel past a Last Mile subnet to circumvent denial of service attacks or to access the Internet without exposing a user's identity to unsecured networks or clouds.
- The ability to securely accept, transfer, and hold various forms of cryptocurrency (including HyperCoins, Bitcoins, and Ether) in private HyperWallets using personal CA-certificate identity-based ownership validation and network-based anti-theft provisions.
- The ability to provide Blockchain-as-a-Service (BaaS) to HyperSphere merchants and startups.
- The ability to support cryptocurrency and token offerings as a platform for a variety of blockchain based companies, services, and startups.

The foregoing features as articulated describe but a few of the HyperSphere's innumerable beneficial hallmarks.

HyperSphere Business Services

As disclosed, the HyperSphere is an open-source electronic communication and e-commerce environment where participants actively contribute by creating demand and by supplying resources needed to transact and fulfill that demand. In a manner similar to the role of the non-profit Linux Foundation provides an open-source platform for secure computing, the HyperSphere is designed to facilitate a fully-distributed decentralized open-source privacy platform for hypersecure networking, communication, cloud computing, and e-commerce. Metaphorically, the HyperSphere is designed to be the Linux of communication and e-commerce.

- Cloud-based hypersecure communication,
- Distributed computing,
- Disaggregated cloud data storage and data backup,

- Secure network-connected (IoT) devices, *and*
- A myriad of online e-services, online and electronic transactions.

HyperSpheric Cloud Communication:

In HyperSpheric cloud communications, merchants may use the network's unique security to offer HyperSecure telephony, conference calling, text messaging, live video, and hypersecure email to individuals, businesses, and corporations, including individual targeted vulnerabilities (such as corporate and executive officers, board members). Merchants may also use the HyperSphere to deliver FIPS-140 compliant professional communication services for governments and its officials, police and emergency services, port authorities, national defense, and homeland security. Corporations may employ the HyperSphere for executing private business networks within a large corporation or in the transportation and shipping industry.

HyperSpheric Cloud Computing:

In HyperSpheric cloud computing, merchants may use to the HyperSphere to deliver locally sourced online business computing for small corporations and public accountants, or to provide cloud-computing services to hospitals and clinics for analysis of medical images. In big data projects, the HyperSphere's distributed computing capability can supply unlimited computing resources to a corporation, government agency, or research institute at a price and performance to match a client's budget. Big data projects include epidemiological studies, DNA analysis and human genome studies, climate and weather modeling, macroeconomics, tectonic and volcanic activity prediction, identifying NEOs (near earth objects), high-energy physics and subatomic particle research, and philanthropic projects such as SETI (the search for extraterrestrial intelligence).

HyperSpheric Cloud Storage:

In HyperSpheric disaggregated data cloud storage merchants can offer low cost, distributed, and massive data storage to individual, corporate, government, and civic clients including hypersecure fragmented storage of personal pictures and videos,

1 corporate archiving of financial data, insurance medical records and medical images, tax
2 and accounting records, corporate IT backup services, as well as archival storage of the
3 contents of libraries, film archives, and rare books. Disaggregated data storage has the
4 advantage that successfully raiding a storage device or facility will yield no useful
5 information whatsoever because the corresponding digital content of the data are spread
6 across the network and saved elsewhere. Moreover, no information as to how data
7 fragmentation occurred is included with the stored media file.

9 ***HyperSpheric Network Connected Devices:***

10 In HyperSpheric *network connected devices* merchants can employ the
11 HyperSphere to provide secure connections to devices used in home, commercial,
12 transportation, government and infrastructure applications. It is well known that Internet
13 connected devices (referred to as the Internet of Things or IoT) are subject to hacking and
14 a variety of identity attacks. By employing a HyperSphere enabled WiFi hub,
15 HyperSphere of Things (HSoT) connected devices prevent outsider intrusion or
16 commandeering of control. Personal HSoT connected devices can include security
17 cameras and home security systems, control IoT devices and personal assistants,
18 appliances, thermostats, entertainment devices and home entertainment networks, voice
19 activated devices, radio-controlled speakers and more.

20 Applications of HSoT connected devices in the workplace include shared drives,
21 printers, conference display screens, HVAC systems, lighting and blinds control, alarm
22 systems, security systems, building maintenance, and more. In factory automation and in
23 power generation, network connected devices may include surveillance cameras, sensors,
24 monitors, locks, failsafe systems, backup systems, gas and air monitors, biosensors,
25 emergency lighting, emergency systems, and more. Infrastructure applications include
26 cameras, traffic flow sensors, traffic lights, freeway metering and commuter lane lights,
27 mass transit security sensors, radio controller parking meters, smart payphones, etc.

28 Connected and autonomous vehicles represent another broad class of network connected
29 device and applications where security and privacy are strictly important. Vehicular and
30 smart hiway applications include vehicle-to-vehicle (V2V), vehicle-to-infrastructure
31 (V2X), and more broadly as vehicle-to-everything (V2E) communications.

HyperSpheric e-Services:

In HyperSpheric *e-services*, merchants can monetize the intrinsic security of the HyperSphere to deliver secure services to risk adverse industries such as banking, transportation, communication, energy, security, treasury, medical, emergency response, and defense uses. Applications include credit card readers, remote ATMs, mobile banking and payments, dispatcher-based professional communication services, and industry specific customized services including HIPA-compliant medical devices and files, FIPS-140 compliant services for military and government applications, or online DRM-compliant video distribution as a replacement for cable and satellite TV subscriber services. In the foregoing applications, the HyperSphere enables a merchant to offer secure transactions to its client base that would be difficult or impossible to guarantee using Internet based communication and commerce.

Other HyperSphere Features:

Other uniquely inventive features of the HyperSphere include its green methods for cryptocurrency generation, its coexistence with the Internet as a public private network, its novel method for decentralized data packet routing, its ability to facilitate privacy protection of a user's identity and asset ownership, and a novel dual cryptocurrency based cryptoeconomic system. When compared to the Internet, the HyperSphere is uniquely capable in e-commerce, a hypersecure platform with the following characteristics:

- Eco-friendly autonomous conjunctive cryptocurrency & digital tokens
- Public-private fully distributed real time network
- Decentralized data packet routing over a stateless meshed network
- Distributed DyDAG network with no master encryption keys
- Privacy protected identity & ownership
- Embedded cryptocurrency for bank-less transactions (HyperSphere cryptoeconomics)

HyperSphere Market

HyperSphere Marketplace:

As an e-commerce platform, the HyperSphere enables merchants to engage in real-time cybersecure network communication and cloud computing with no capital investment in hardware, infrastructure, R&D, or cyber-security developments. Rather than depending on privately owned or contractually obligated leases of servers, VPNs, or dedicated dark-fiber channel capacity, HyperSphere merchants and service providers use HyperContracts to solicit and contract independent resource providers (HyperNode owners) to facilitate network communication and execute their transactions. Using artificial intelligence and machine learning, the decentralized HyperSphere Marketplace then solicits and procures the necessary HyperSphere resource providers to complete each contract.

HyperContracts can be ‘hard coded’ by HyperSphere contract savvy software engineers or, as shown in **FIG. 48**, generated automatically or quasi-automatically through API interfaces and templates, including the use of ‘HyperSphere services 949,’ utilities created and digitally signed by the HyperSphere as system validated transactional process code. These HyperSphere service utilities render certain commonly executed processes such as HyperCoin sales, asset transfers, point-of-sale transactions, installing HyperNodes onto devices, creating HyperNode clusters, signing HyperWallets, etc., easy to use without the need to write executable code from scratch.

To match HyperSphere merchant and service provider’s requirements to resource providers willing to meet the terms and deliverables specified in a HyperContract, the HyperSphere utilizes the HyperSphere Marketplace, a decentralized electronic marketplace. In operation, the HyperSphere Marketplace solicits resources and negotiates terms mutually acceptable to all parties. HyperNodes participating in the contract resolution process are by definition not a party to the HyperContract. After negotiation is completed, the broker nodes are appended to the HyperContract for their role in brokering an actionable agreement. HyperMetal compensation is only paid upon successful execution of the HyperContract.

1 ***Resource Providers (HyperNodes):***

2 HyperSphere resource providers deliver communication, computing and storage
3 capability to merchants in accordance with performance requirements stipulated in
4 HyperContracts— electronic contracts offered by merchants describing tasks, deliverables,
5 and compensation. A HyperSphere resource provider is any network-connected
6 communication device hosting operation of a HyperNode— a software-based portal for
7 accessing the HyperSphere. HyperNodes may be downloaded from a trusted app store or
8 from the HyperSphere’s website. For identity validation, HyperNode owners use digitally
9 signatures to ensure ownership by a specific parental CA-certificate and identity-trust-
10 chain.

11 In operation active HyperNodes participating in completed transactions
12 immediately earn HyperCoin cryptocurrency in accordance with their contributions.
13 Contribution value and compensation depends not only on market demand, but also on
14 intrinsic capability, speed, reliability, etc. of the HyperNode’s host device. HyperNodes
15 are not limited to operation on a single hardware host but may comprise clusters of
16 devices forming a shared account linked to a specific perpetual blockchain and parental
17 CA-certificate. Specifically, in the HyperSphere, resource providers are subdivided into
18 four tiers of HyperNode owners based on their performance, speed, capacity, and uptime
19 capability of their hosts, namely:

- 20
- 21 • 1st Tier: High-speed, high-capacity global server networks with high availability,
22 such as Azure, AWS, GWS, IBM Cloud Services, etc.
 - 23 • 2nd Tier: High-speed, local server clouds including ISPs, cable networks, bitcoin
24 miner farms, etc.
 - 25 • 3rd Tier: Medium-speed, AC-powered computers and CPUs including PCs,
26 gaming consoles, smart TVs, routers, etc., and
 - 27 • 4th Tier: Mobile and IoT devices including notebooks, tablets, smartphones,
28 games, appliances, etc.
- 29

30 Merchant access to and pricing of a specific tier of resource provider is
31 determined by the cost and performance requirements stipulated in a merchant’s

HyperContract and by the market dynamics of supply and demand. During execution of a HyperContract, pledged payments (made in HyperMetal or HyperCoin) are recorded on the account owner's corresponding blockchain. Upon completion of a HyperContract and confirmation by a jury-of-peers, the HyperNode mints HyperCoins in accordance with the pledge specified in the HyperContract.

HyperSphere Merchants (Service Providers):

Referring to **FIG. 47**, A HyperSphere merchant [RKW1]958 [RKW2] is an independent company that uses the HyperSphere to deliver products and services to its client 959. The merchant 958 pays only the HyperNodes 957 that are involved in carrying its data 956 or executing its prescribed tasks. HyperSphere [RKW3] resource providers, operating HyperNodes 957, perform tasks and execute contracts for the HyperSphere merchant 958.

The merchant 958, in turn, provides services for its client 959, including such services as business-to-business (B2B) and business-to-consumer (B2C) support. Since the HyperSphere is a platform and not a company, client 959 is unable to use or to access the HyperSphere except through the HyperSphere merchant 958. Commercial merchants are able to compete more effectively using the HyperSphere than the Internet because they can access greater resources with higher performance at a lower cost.

The service provided by HyperSphere merchant 958 may [RKW4] include:

- Financial services
- Asset management
- Entertainment
- Manufacturing
- Healthcare

By creating [RKW5] a HyperSphere API-generated application or user interface, merchants and service providers can offer communication, computing, storage, cloud-connected devices, or e-services and products to their clients (even if their clients are not HyperSphere clients).

HyperContracts:

Transactions in the HyperSphere occur using digitally specified procedures called HyperContracts issued by HyperSphere merchants and service providers to solicit and contractually stipulate deliverables from HyperNode resource providers. As shown in **FIG. 47**, every HyperContract comprises a *job specification* 964a and a HyperMetal or HyperCoin *reward pledge* 964b, describing the compensation reserved, i.e. pledged, for payment to resource providers participating in the contract's successful execution (including jurors and backup nodes). To provide both transparency and to confound blockchain attacks, the jury-of-peers used for consensus-based validation includes both public and cloaked members, observers unknown by the parties transacting business until after the consensus option has been rendered. To solicit job resources and encourage participation, a HyperSphere merchant or service provider attaches a reward pledge to the HyperContract along with the job specification.

The pledge, once attached, is temporarily sequestered, i.e. removed, from the merchant's blockchain and essentially held in digital escrow pending contract completion or failure, thereby ensuring payment with the proviso that the contract is executed. The merchant next delivers the proposal to the HyperSphere Marketplace, a decentralized market using AI-based algorithms executed by HyperNodes. The bidding process is iterative using various silent auction methods, continuing until all the required resources including participants, jurors, and backups are committed. The accepted contract is then executed as specified. Remuneration is likewise paid in accordance with contractual obligations.

Sharing Economy:

The HyperSphere itself serves as the electronic framework and ecosphere in which communications and transactions occur. Comprising a heterogeneous decentralized network, the HyperSphere cloud co-exists with the Internet using the same servers, microwave towers, satellites, fiber networks, dark fiber channels, and routers as public and for-profit private clouds. Similarly, the HyperSphere's 'last mile', is routed over local ISP's Ethernet and fiber lines, over cable networks, and over 2G, 3G/LTE, 4G,

1 and 5G networks with no requirement to deploy new or dedicated infrastructure. The last
2 link to a user's device employs whatever connectivity is available, be it Ethernet, WiFi,
3 mobile telephony, or an *ad hoc* peer-to-peer (P2P) or vehicle-to-vehicle (V2V) network
4 in which the user participates. In the HyperSphere, network operations comprise fully
5 decentralized operations with no central authority, no master encryption keys, and no
6 governing security credentials. In essence, users are unaware of how the HyperSphere
7 achieves its communication and transactional security and privacy.

8 Users, whether merchants (service providers) or resource providers, connect to the
9 HyperSphere using 'bring-your-own-device' (BYOD) hardware simply by installing a
10 software- or firmware-based network-portal called a HyperNode. HyperNodes not only
11 enable a device to access the HyperSphere and to reach other HyperNode owners, but
12 also serve as a communication node in the network. Each new HyperNode connecting to
13 the HyperSphere expands the size, density, capacity, and performance of the
14 HyperSphere. In this sense the HyperSphere is the 'the people's network,' a network
15 made up of its users, merchants, and resource providers.

16 Unlike large cloud operators, AWS, GCP, Azure, Facebook, and global telephony
17 carriers AT&T, NTT, Verizon, T-Mobile, the HyperSphere is not a party to or
18 beneficiary of the business transactions it hosts. Instead, merchants fund the resources by
19 paying only for the HyperNodes involved in carrying their data or executing their
20 prescribed tasks. Aside from a small transaction fee for processing certain types of
21 transactions, the HyperSphere does not receive compensation or set the price of services
22 executed on its platform. The cost of services in the HyperSphere is instead determined
23 by a negotiation between the HyperNode owners, i.e. the resource providers, and the
24 merchant or service-provider seeking network, storage, or computing resources.

25 In one embodiment, these tasks are assigned and distributed to available (online)
26 and willing HyperNodes through an artificial intelligence (AI) based marketplace
27 matching merchant's demands to various resource provider's capacity and capabilities. In
28 another embodiment, the tasks to be performed are specified and the participating
29 HyperNodes agreeing to perform such services are assigned in a digital contract called a
30 HyperContract. In a third embodiment, HyperContracts are electronically negotiated in
31 an AI-based HyperSphere Marketplace. HyperNodes serving HyperContracts may

comprise one of four tiers of HyperSphere Resource Providers classified by their performance, reliability, capacity, and availability, comprising:

- ***Tier-1 Resource Providers:*** Tier-1 Resource Providers comprise highly reliable, high capacity, high bandwidth cloud communication, cloud computing, and cloud storage suppliers. Tier-1 providers offer premium performance at the highest HyperMetal payment prices, taking HyperSphere marketplace job contracts, i.e. HyperContracts, with a guaranteed service contract payment of HyperMetal and with no HyperMetal performance bonus. Tier-1 resources include AWS, Azure, GCP, etc. Tier-1 resources operate on a business model of “guaranteed performance, guaranteed payment” converting HyperMetal to HyperCoins upon job completion, and most likely exchanging HyperCoins immediately for fiat currency. Statistically, Tier-1 resource providers are expected to service performance demanding applications including premium HyperSphere merchants offering commercial pay-for-service businesses, and real-time applications sensitive to latency and propagation delay such as voice and live video.
- ***Tier-2 Resource Providers:*** Tier-2 Resource Providers comprise high capacity and high bandwidth computing platforms and Internet providers lacking global omnipresence or guaranteed availability. Tier-2 resources offer high-throughput cloud connectivity, high speed cloud-connected computing, and high- capacity storage at competitive, commercially aggressive rates compared to Tier-1 Resource Providers. Examples include local ISPs, cable operators, university server farms, and re- purposed high capacity Bitcoin miners. HyperContracts for Tier-2 Resource Providers negotiated in the HyperSphere marketplace comprise a combination of fixed and bonus HyperMetal compensation. HyperMetal bonus compensation varies, depending on market competition and a provider’s performance. Upon receiving payment in HyperMetal for fulfilling a HyperContract, the received HyperMetal is automatically converted to HyperCoins, the HyperSphere’s fungible cryptocurrency. Although Tier-2

HyperNodes can hold HyperCoins for speculative investment, Tier-2 HyperNodes are more likely to immediately reuse HyperCoins for contracting services or to cash them out to avoid fluctuations in HyperCoin cryptocurrency prices. Alternatively, Tier-2 resource providers may elect to convert HyperCoins into HyperMetal through a tandem currency-exchange process comprising the steps of (i) exchanging HyperCoins to fiat currency at the current market rate, and subsequently (ii) purchasing HyperMetal at the current quant (bit rate) cost. HyperCoins cannot be used to purchase HyperMetal directly without becoming subject to onerous government regulations regarding ‘options contracts’.

10

- ***Tier-3 Resource Providers:*** Tier-3 Resource Providers comprise opportunistic providers comprising wireline connected or high uptime devices such as personal computers, small servers, private online storage farms, and HyperNode enabled routers (i.e. HyperSpots). Tier-3 resources are used for HyperSphere Merchants less sensitive to communication latency such as online shopping sites, regional services, online back and storage of photos, files, media contents, etc. Tier-3 resources may also be used philanthropically to support students, researchers, startups, and entrepreneurs lacking access to computing resources.

HyperContracts for Tier-3 Resource Providers negotiated in the HyperSphere marketplace a mix of contract and bonus HyperMetal compensation with a greater possibility for guaranteed HyperMetal compensation during periods of high demand. Tier-3 Resource Providers often may comprise individuals or small companies wishing to supplement their income or lower their operating costs. Upon receiving payment in HyperMetal for fulfilling a HyperContract, the received HyperMetal is automatically converted to HyperCoins, the HyperSphere’s fungible cryptocurrency. Although Tier-3 HyperNodes can exchange HyperCoins to fiat currency at any time, they are less likely to “cash out” upon payment than Tier-1 or Tier-2 providers, and are more likely to hold them for investment purposes, risking a capital loss for the upside potential of the cryptocurrency.

31

- **Tier-4 Resource Providers:** Tier-4 Resource Providers comprise mobile devices such as cell phones, tablets, notebook computers, and automobiles that offer only limited-duration connectivity to the HyperSphere cloud. Such devices are primarily used for increasing the HyperSphere’s communication mesh density, especially in peer-to-peer networks and in rural or dense urban areas where access to cell towers may be limited or obstructed. HyperContracts for Tier-4 Resource Providers negotiated in the HyperSphere marketplace primary comprise bonus HyperMetal compensation automatically converted into HyperCoins, the value of which varies in accordance with periods and localities of high demand. Although Tier-4 HyperNodes can exchange HyperCoins to fiat currency at any time, they are less likely to “cash out” upon payment than other tiers of providers, and are more likely to hold them for investment purposes, risking a capital loss for the upside potential of the cryptocurrency.

HyperSphere Resource Providers can leverage their cell phones, notebooks, personal computers, and company computers to make income from devices that would otherwise sit idle at least two-thirds of every day. Resource Providers include everyday ordinary people wishing to augment their paychecks or cut their phone and utility bills, Small companies can improve their profitability and improve cash flow as well as access global customers otherwise not available to them. Gamers can share their resources when they are not playing in exchange for access to a global network of computing power and challengers to enhance their gaming experiences when they are active. Bitcoin miners can find an alternative use for their hardware investments as the profit potential of the Bitcoin market continuously declines. Universities can share their research computers with other institutes to balance workloads, reduce costs, and improve peak performance. Large companies can improve their loading through access to HyperSphere clientele.

HyperSphere merchants are independent companies using the HyperSphere to deliver products and service to their clients. In one embodiment, the HyperSphere Marketplace comprises an AI (artificial intelligence) based transactional ecosphere matching merchant demand against cloud-connected suppliers, i.e. where HyperSphere resource providers perform tasks and execute contracts for HyperSphere merchants. The

merchants in turn, provide services for their clients including business-to-business (B2B) and business-to-consumer (B2C) support. Since the HyperSphere is a platform and not a company, consumers are unable to use or to access the HyperSphere except through a HyperSphere merchant. Commercial merchants are able to compete more effectively using the HyperSphere than the Internet because they can access greater resources with higher performance at a lower cost.

Any individual or corporation wishing to participate in HyperSpheric e-commerce can actively engage in the HyperSphere as a merchant, a resource provider, or both, with no investment or infrastructure expense. As a HyperSphere merchant, businesses can immediately access the potentially limitless resources of the HyperSphere, engaging in international businesses and global marketing at competitive costs. As a HyperSphere resource provider, a computer network operator, disenfranchised Bitcoin miner, regional ISP provider, cable network operator, personal computer, or even smartphone owner, can earn profit by contributing communication and computing assets to the HyperSphere on a full-time or part-time basis. Customers of HyperSphere merchants vary with the services a merchant offers. Customers of HyperSphere merchants vary with the services a merchant offers. Businesses supported by the HyperSphere include the following:

HyperSphere Digital Currencies

Digital tokens used by the HyperSphere involve two types of cryptocurrency – HyperMetal transactional tokens, and HyperCoin utility tokens. The purpose of HyperMetal and HyperCoins is to facilitate a means by which merchants and resource providers can transact e-commerce in the HyperSphere.

HyperCoin Cryptocurrency:

HyperCoins, the HyperSphere's tradable cryptocurrency and utility token, is a fully fungible medium of commerce with capability of...

- Being traded, i.e. bought or sold, in banks or independent digital currency exchanges,

- Being minted from HyperMetal by resource providers (HyperNodes) as earned compensation for completing tasks and fulfilling HyperContracts, or
- Being used to engage and pay resource providers (HyperNodes) for completing tasks and fulfilling HyperContracts, the cryptocurrency being recycled into new HyperCoins (having new digital cryptographic identities).

HyperCoins bought or sold in banks independent digital currency exchanges and optionally used in the HyperSphere, are intrinsically volatile and subject to supply and demand market dynamics. Conversely, HyperMetal, purchased from the HyperSphere directly, is intended exclusively for contracting services in the HyperSphere, and cannot be sold, bartered, or hypothecated. Risk adverse merchants should employ HyperMetal to contract resource providers. Risk adverse resource providers minting HyperCoins as compensation, should immediately use the cryptocurrency to purchase services from other HyperSphere merchants or immediately exchange their HyperCoins for fiat currency in a digital currency exchange.

Holding on to a HyperCoin involves economic risks. Aside from their utility in the HyperSphere, the value of a HyperCoins as an investment vehicle is unknown, as it depends on market dynamics... The conversion rate between international currency, aka “fiat” currency, and HyperCoins is subject speculation and currency fluctuations. Should a purchaser hold HyperCoins for an extended duration there is no guarantee the HyperCoins price may appreciate or that the HyperCoins holder may be able to sell their HyperCoins for a price higher than they purchased them. In one embodiment of this invention, in order to sell HyperCoins, a purchaser or token holder must be an active participant in the HyperSphere requiring downloading of a HyperNode onto one or more of their computing or communication devices and actively connecting to the HyperSphere for a prescribed duration. Absent an actively connected HyperNode, the associated HyperCoins will be disabled for eligibility for sale or exchange until the criteria is met. In other embodiments, a transaction fee is levied, but where there is no requirement for a user to own an active HyperNode.

HyperMetal Token:

1 The non-tradable internal cryptocurrency of the HyperSphere, HyperMetal utility
2 tokens, serve a single purpose– to create and execute HyperContract transactions and to
3 pledge reward payments for successful task and micro-task completion. HyperMetal may
4 be purchased by fiat currency from the HyperSphere or through authorized digital
5 currency exchanges (DCEs). HyperMetal may not be sold or bartered. The purchase price
6 of HyperMetal (in fiat currency) is set by the network’s bit cost. The bit-cost is the
7 normalized weighted average cost for 1st Tier resource providers, unrelated to the trading
8 price of HyperCoins. Because HyperMetal can only be used to pledge compensation for
9 HyperContracts and is not tradable, the purchase price of HyperMetal is relatively stable
10 and not subject to speculation driven price volatility.

11 Detached from trader induced price volatility, merchants and resource providers
12 can accumulate and hold HyperMetal to fund a company’s HyperSpheric operations
13 without risk of speculators’ driving up costs, destabilizing their business and supply
14 chain. And because the HyperMetal’s purchase price is set by 1st Tier supplier bit rates,
15 purchasing HyperMetal is commercially equivalent to guaranteeing access to the highest
16 performance networks. Discounted by the buying power of the HyperSphere’s growing
17 economy, merchants can access resources at prices they would otherwise be unable to
18 procure on their own.

19 Since HyperMetal is purely transactional and used exclusively within the
20 HyperSphere, it is not subject to speculation or currency fluctuations thereof. HyperMetal
21 can only be converted back into HyperCoins by engaging in e-commerce in the
22 HyperSphere, i.e. where a HyperSphere Merchant uses HyperMetal to hire a service from
23 a HyperSphere Merchant. The purchase price of HyperMetal is determined not by the
24 trading price of HyperCoins, but by the actual bit-rate cost of Tier 1 resource providers
25 also referred to herein as “quants”, a measure of computing and communication bit-rate
26 cost.

27 As such the exchange rate from fiat currency to the HyperCoins may vary, as will
28 the exchange rate of HyperCoins needed to purchase HyperMetal, but the net exchange
29 rate between international fiat currency and HyperCoins will vary only in the real bit-rate
30 cost or a quant of cloud services. If the HyperCoins appreciates, fewer utility tokens will
31 be needed to purchase a fixed quantity of HyperMetal transactional tokens. Conversely, if

1 the trading price of HyperCoins drops, more utility tokens will be needed to procure the
2 same quantity of transaction token. Regardless of the market price of HyperCoins, the
3 equivalent purchase price of HyperMetal in international currency remains relatively
4 constant, essentially varying proportionally with the global cost of electrical power.

6 ***Dual Token Economy:***

7 The HyperSphere economy as disclosed concurrently operates on two forms of
8 digital tokens or virtual currency – HyperMetal used within the HyperSphere marketplace
9 used by HyperSphere merchants to procure resources and schedule tasks, and
10 "HyperCoins", used to compensate and reward HyperSphere resource providers for
11 completing jobs in a timely manner and for engaging to the HyperSphere economic
12 environment. Only by completing tasks can HyperSphere resource providers convert, i.e.
13 "mint", HyperMetal into HyperCoins. In the HyperSphere, *minting*, the process of
14 completing tasks and converting HyperMetal into HyperCoins, is analogous to *mining* in
15 other cryptocurrencies except that it involves two digital currencies, one created, the
16 other destroyed. Unlike mining of Bitcoins, minting of HyperCoins is based on a resource
17 provider completing a task, job, or contract for a merchant. Minting is not based on
18 gambling or gaming.

19 The commercial value of one of the HyperSphere's unique innovations – the
20 concept of a dual virtual currency system, is that the HyperCoins can float to any value
21 without affecting the real cost of engaging in e-commerce within the HyperSphere. Risk
22 adverse merchants can eliminate current concerns by concurrently purchasing
23 HyperCoins and simultaneously converting them to HyperMetal, thereby completely
24 eliminating the risk of HyperCoin currency fluctuations from their purchases. Merchants
25 wishing to gamble on HyperCoin appreciation may hold the HyperCoin before redeeming
26 it for HyperMetal, in the hope that their HyperMetal purchasing power will improve if
27 the HyperCoin trading price increases (it may not). In this manner, the HyperSphere can
28 satisfy speculative and conservative business owners and merchants.

30 **Digital Currency Generation**

As shown in **FIG. 49**, HyperCoins can be generated in two ways, either through *minting* 960, the process of converting HyperMetal into HyperCoins, or by recycling 961 (melting) HyperCoins into new HyperCoins. In this process, the payment pledge of HyperContract 964 is ratably apportioned among participating HyperNodes then used to synthesize new HyperCoins. If payment is pledged as HyperMetal 970a, HyperContract 964 transfers and locks 971a the pledge onto the HyperNode 962 owner's HyperMetal blockchain 965a. After contract execution and consensus, the pledge is unlocked 972a and recorded as a debit 973 on the HyperNode 962 owner's personal HyperMetal blockchain 965a, and entered as a corresponding unlocked credit 974 onto the HyperNode 962 owner's HyperCoin blockchain 965b.

In the case of HyperCoin recycling 961, the pledge 970b is entered onto the HyperCoin blockchain 966 of HyperNode 963 and held until HyperContract 964 completion. Thereafter, the HyperCoin is melted 975, i.e. reissued with a new cryptographic code, recycling the old coin into a new one 975. In either minting 960 or recycling 961, participating HyperNodes 962 or 963 automatically generate new HyperCoins 975 at the time of contract completion. Once generated, the HyperCoins can be sold, transferred, or moved into wallets.

Conjunctive BC Generation:

One unique feature of the HyperSphere is its ability to support e-commerce, online shopping, business transactions, and money wires without the need of banks or credit cards. Instead, the HyperSphere utilizes its own network-embedded virtual currency, i.e. digital tokens, to eliminate the cost, complexity, delays, and inefficiencies of transacting e-commerce using international currencies, money wires, and conventional credit-card based payment mechanisms. The generation of these digital tokens occurs autonomously and conjunctively as data packets move through the HyperSphere's unique dynamic network. In one embodiment, unique cryptographic codes are distributed to participating HyperNodes as data moves through the network.

During network operation, a cryptographic code comprising a transient or temporary blockchain is modified with each data hop in the cloud. This verification code

1 used to substantiate a HyperNodes participation in carrying network traffic or performing
2 a task is referred to as a HyperNode hop code or HHC. Because token generation
3 accompanies the network performing real work, no energy is wasted on meaningless
4 mining operations. Instead digital tokens are “minted” as a reward for resource providers
5 completing tasks, jobs, and contracts. This novel Proof-of-Performance (PoP) mechanism
6 requires less than one-trillionth of the energy consumed by mining Bitcoins, representing
7 the world’s first eco-friendly digital token generation. Once issued, the sale or transfer of
8 tokens is chronicled in published blockchain ledgers, easily verifiable by a jury of peers.
9 To initiate new HyperContracts, digital tokens or a portion thereof, are recycled, i.e. used
10 to pay for a new transaction, job, or contract, during which the previous token is melted,
11 and its blockchain retired.

12 HyperNode hop code (HHC) token generation is not exclusively limited to
13 HyperCoin and HyperMetal generation and transactional processing. The same methods
14 can also be used to generate counterfeit-proof digital tokens for merchants using the
15 HyperSphere. This feature – creating, distributing, and verifying digital tokens as a
16 service for merchants using HyperNode Hop Codes, or tokens-as-a-service (TaaS) is
17 unique to the HyperSphere. Because the generation method involves the synthesis of
18 either a transient blockchain (tBC) or a perpetual blockchain (BC), the feature of TaaS
19 can be considered as a special version of a blockchain-as-a-service (BaaS). HyperSpheric
20 TaaS custom-generated tokens can be used for a variety of purposes including reward
21 tokens for customer allegiance; discount coupons for purchases; reward mileage and
22 perks for frequent travelers; gaming rewards for game performance; preferred tickets,
23 backstage passes, and meet-and-greet VIP perks for artist fan club members, shopper
24 rewards for completing information requests or credit card applications, and more.

27 ***HyperCoin Minting:***

28 To prevent the fraudulent generation and falsified validation of cryptocurrency,
29 the HyperSphere does not employ mining to generate new coins. Instead of trusting an
30 unknown miner and a potentially corruptible jury-of-peers to validate Proof-of-Work
31 solutions of numerical and cryptographic puzzles, the HyperSphere synthesizes

cryptocurrency through a reliable internal process executed by its network of HyperNodes. This generation method is not observable to outside observers or subject to packet sniffing, and is unrelated to PoW nonce-hash puzzle solving. In its unique implementation, cryptocurrency generation occurs adjunctively as an intrinsic part of network operation during data transport in execution of HyperContracts without requiring additional energy or effort.

In operation, the HyperSphere utilizes multi-tree DyDAG blockchains to generate cryptocurrency and to record ownership. The DyDAGs may comprise ‘perpetual’ or ‘transitory’ blockchains depending on their purpose and application. *Perpetual blockchains* (BC) establish ownership by linking extant cryptocurrency to personal CA-certificates using an identity-trust-chain lineage derived from a corresponding parental certificate.

Transitory blockchains or ‘tBC’ are, in contrast, temporary distributed ledgers used to execute HyperContracts, synthesize cryptocurrency, and ratably apportion compensation to participating resource providers. Unlike the permanence of a perpetual blockchain, once the task of a transitory blockchain is completed its blockchain is destroyed. In this manner perpetual blockchains do not get burdened carrying unnecessary and irrelevant blocks of minutiae. All cryptocurrency synthesis in the HyperSphere starts with a HyperContract, a business agreement between resource providers and their clients— service providers and merchants. Each HyperContract comprises a *job specification* and a *reward pledge* describing the compensation reserved for resource providers participating in the contract’s successful execution. HyperContract pledges can be made in two different cryptocurrencies, either HyperMetal or HyperCoins.

The process of *minting*, converting a HyperMetal pledge into HyperCoins is shown in **FIG. 50**. As shown, the vertical lines represent perpetual blockchains 970a and 970b; 971a and 971b; 972a and 972b; and 973a and 973b held by HyperNode owners 970c through 973c and signed by the owner’s corresponding CA-certificate 970d through 973d based identity-trust-chain. The roles of participating HyperNodes 970c through 973c are represented by their metamorphic function performed respectively, as either name server [NS], authority [A], or task [T] nodes for job execution, or authority [A] nodes participating as observers in jury-of-peer consensus. Each participating node has two

DyDAG perpetual blockchains--one set for HyperMetal transactions 970a, 971a, 972a, and 973a, the other for HyperCoin transactions 970b, 971b, 972b, and 973b.

In the DyDAG matrix shown, new blocks are appended onto these perpetual blockchains, in sequence ordered from top to bottom and time stamped accordingly. In the same illustration, horizontal arrows represent transitory blockchains tBC 976, 978e, 978f, 982, and 983a through 983d. Transitory blockchain are impermanent--executed sequentially, they modify perpetual blockchains and are subsequently discarded. As depicted, the minting process of HyperCoin generation occurs sequentially from top to bottom with tasks executed left to right. As listed in the order shown in the HyperContract, these processes involve the following milestones:

- HyperContract pledging 975 at time t_{mp}
- HyperContract task execution 977 over the duration Δt_t
- HyperContract consensus 979 at time t_c
- HyperCoin minting 981 at time t_g

In HyperContract pledging 975 at time t_{mp} , the decentralized HyperSphere Marketplace successively concludes contract negotiation, at which time the merchant sponsor distributes the HyperMetal pledges (e.g., 985d) to the committed contract participants, recording the pledge onto their HyperMetal blockchains 970a, 971a, 972a, and 973a as a pending transaction without actually transferring the HyperMetal. In this manner, the pledge acts as a blockchain version of an escrow by locking the currency to prevent double spending. HyperContract execution 977 occurs over a period of time, i.e. during the interval Δt_t when the HyperNodes execute a series of tasks (or subtasks) in accordance with the HyperContract's job specification. During data transit and micro-task execution, each HyperNode is delivered a cryptographic receipt, a transitory blockchain, e.g., 978e, containing a series of hashed blocks containing *HyperNode hop codes* or HHCs of the HyperNodes before it.

As shown in **FIG. 51**, starting with HyperContract 994a, HyperNodes 991, 992a, 992b and 992c autonomously generate cryptographic codes 994b through 994e as part of a data packet's SDNP based routing instructions. Upon completion of their work, each

HyperNode adds its own cryptographic block to the transitory blockchain. The HyperNode then forwards the new longer blockchain onward to the next node, which in turn repeats the process. In this manner, each HyperNode has irrefutable evidence of its participation. For example, task node |T| 992b receives transitory blockchain tBC₁ containing cryptographic code 994c, as shown having a value {tBC₀ + h(HHC₁)}, and exclusively receives the HyperNode hop code HHC₂ 995c corresponding to the transport of data packet 983a, and then after processing forwards a revised transitory blockchain tBC₂ 994d having a value {tBC₁ + h(HHC₂)} onto the next HyperNode 992c. In this manner, a string of self-consistent blocks is generated excluding the possibility of imposters because they do not have access to the incoming transient blockchain or the HyperNode hop code HHC of the last data transport. Expressed algebraically in terms of SDNP network generated HyperNode hop codes HHC_x 995b, 995c, 995d and cryptographic hash function $h(x)$ then:

$$tBC_j = h(HHC_j) + tBC_{j-1}$$

In this process, the HyperContract itself forms the initial block of a transitory blockchain used in HyperContract execution, or tBC₀ = $h(HC')$ 994a. As data packets pass through HyperNodes 991, 992a, 992b, 992c in succession, a copy of the transient blockchain tBC_j 994b, 994c, 994d, 994e is deposited, i.e. written onto the HyperNode's HC blockchain with the transient blockchain growing in length as the job is executed. Upon reaching the terminus node, the final full-length transitory blockchain tBC_f 994e is returned to the HyperContract initiator to confirm task completion. The full-length tBC_f blockchain is concurrently forwarded to the jury-of peers specified in the HyperContract for checking.

Returning to **FIG. 50**, at time t_c , consensus by a jury-of-peers confirms contract execution 979 using a RBOS (replicant blockchain observer segment), e.g., 980d, to facilitate inspection without the possibility of backtracing. Upon confirmation of HyperContract completion, peer review, and consensus, every participating HyperNode owner is awarded compensation according to their contribution using a copy of the transitory blockchain tBC₀, tBC₁, and tBC₂ establishing, i.e. 'proving' their performance.

1 Once confirmed, the transitory blockchain is automatically converted, i.e. *minted*, into a
2 specified number of HyperCoins 984 and recorded on the HyperNode owners' perpetual
3 HyperCoin blockchains 970b, 971b, 972b and 973b. Cryptocurrency synthesis where
4 participating nodes prove their contribution in performing real tasks is referred to as
5 *Proof-of-Performance* or PoP.

6 Upon proving a HyperNode's performance 981 at time t_g the HyperMetal is
7 unlocked (using an unrestricted code) 985 and a new code is generated containing a hash
8 of the HyperMetal pledge and the transitory blockchains tBC 983a, 983b, 983c and 983d,
9 proving a valid peer-reviewed origination. Graphically, the minting process is depicted as
10 an unlocking 985 and a debit 986 from the HyperMetal perpetual blockchain BC 970a,
11 971a, 972a, and 973a and a corresponding credit 987 onto the HyperCoin perpetual
12 blockchain BC 970b, 971b, 972b and 973b of the same blockchain owners. HyperCoins,
13 once minted, may be converted to international fiat currency or used in the HyperSphere
14 to solicit resources, a process equivalent to recycling.

15 In HyperCoin recycling, shown in **FIG. 52**, a HyperContract pledge is made in
16 HyperContract 975 in HyperCoins (not in HyperMetal). The pledge, transferred by
17 transient blockchain 988, is recorded on the HyperNode's perpetual HC blockchain 970b,
18 971b, 972b and 973b as a pending transaction (without actually transferring the
19 HyperCoin). Locking 989 the HyperCoin into a digital escrow at the time of contract
20 negotiation is especially important to prevent double spending, especially since the
21 HyperCoin is fungible and tradable as a liquid asset. Task execution 977 and juror
22 consensus 979 for recycling HyperContract execution occurs in the same manner as a
23 minting HyperContract, except at the time of contract completion t_g 981a when new
24 HyperCoins 984a are generated.

25 In the recycling process, the cryptographic identities of the original pledged
26 HyperCoins are destroyed, (in HyperSphere taxonomy "melted") then re-minted as new
27 HyperCoins with new digital identities (depicted as a debit 998 and concurrent credit 999
28 on the HC perpetual blockchain 970b, 971b, 972b, and 973b). Like newly minted
29 cryptocurrency, recycled HyperCoins employ a digital identity based on a cryptographic
30 hash value $h(x)$ derived from HyperNode hop codes HHC_j and the original
31 HyperContract HC. The HyperSphere's recycling process is entropic (lossy), not

conservative, as the quantity of generated HyperCoins re-minted by HyperNodes is less than the number of HyperCoins pledged in the HyperContract, $\#HC_{new} < \#HC_{pledge}$, naturally reducing the number of HyperCoins in circulation by reuse attrition.

Other Blockchain Processes

Unlike in cryptocurrency mining, which only pays miners lucky enough to solve an arduous puzzle before others can, in Proof-of-Performance all HyperNodes participating in a successful HyperContract execution receive a contractually guaranteed return as minters. And because it occurs adjunctively with SDNP network operation, minting and recycling HyperCoins essentially consumes no more electrical energy than performing communication or e-commerce itself. In essence, HyperSpheric cryptocurrency synthesis wastes virtually no energy at all. To fully appreciate cryptocurrency generation and transactional processing in the HyperSphere and how these processes differ from Bitcoin, Ethereum, and conventional blockchain applications, it is insightful to consider a device's system architecture. As shown in **FIG. 53**, a computer or communication device supports software applications using an 'operating system' (OS) such as Windows, MacOS, Linux, Android or iOS. The operating system is hosted on a platform comprising hardware and drivers 1001, typically including multiple CPUs, memory, and device connections. An operating 'kernel' 1002 provides resource scheduling and task management for the OS acting an interface, i.e. a liaison between the hardware and an applications environment (referred to here as a application environ 1003 to avoid ambiguity). The application environ hosts a variety of software including APIs, UI/UX, database, business, email, VoIP-messengers, remote access gateways, IoT, Web apps, and more. Most apps today are network enabled, facilitating Internet-of-Everything (IoE) connectivity.

As depicted, the operating kernel 1002 interacts directly with both applications environ 1003 and the underlying hardware platform 1001. The kernel 1002 also interacts with the communication protocol stack, especially via Application Layer-7, and at layer 1.5 (the interfacial quasi-layer existing between PHY Layer-1 and Data Link Layer-2). In operation, signals received by PHY Layer-1 are passed up to Layer-2 and concurrently transferred to the OS kernel for job scheduling. In turn, the kernel schedules tasks

1 through its interaction with Application Layer-7 to support software running atop the
2 device OS (in the application habitat). In this mechanistic explanation, it is insightful to
3 distinguish the primary communication role of Application Layer-7 (in the SDNP or
4 TCP/IP protocol stack) from the functions of computer application programs (running
5 within the OS's application environ). Specifically, Application Layer-7 data packets
6 provide high-level network connectivity to specific applications but are incapable of
7 operating independently from the OS-hosted applications. In that sense the application
8 environ 1003 sits atop the OSI protocol stack immediately above Application Layer-7. In
9 the parlance of layers of abstraction, Application Layer-7 supports software running in
10 the application environ above it and the software relies on Layer-7 supplied information
11 for support. To function, the software and the data packet payloads must match in type,
12 syntax, version, etc. For example, without database software installed on a device SQL
13 instructions received on Layer-7 will go unrecognized and unanswered.

14 Data packets carrying Hypertext Transfer Protocol (HTTP) content for
15 distributed, collaborative, and hypermedia information over the Web is completely
16 useless without a browser application able to interpret HTML or XML. Similarly, in
17 conventional cryptocurrency blockchain, transactions received as Layer-7 payloads
18 cannot modify or append new blocks onto an existing blockchain without corresponding
19 application support. All conventional blockchain and cryptocurrency transactions occur
20 entirely within the app environ of the host OS, not part of the protocol stack. As a
21 network portal to the HyperSphere, HyperNodes 1005 span the SDNP protocol stack and
22 OS app environs, communicating directly with the Network & Transport Layers-3 and -4,
23 with SDNP Application Layer-7, and with its API and UI/UX in the OS apps environ
24 1003.

25 In blockchain processing, the HyperSphere is wholly unique, generating HHC
26 cryptographic HyperNode hop codes as part of Network Layer-3 and using this
27 information in a blockchain processor or 'BCP' 1007, a network-connected software
28 engine used in blockchain generation and transactional processes. The BCP 1007 then
29 supports blockchain apps including BaaS (Blockchain as a Service) 1008a and various
30 blockchain apps 1008b. Although BCP 1007, BaaS 1008a, and BC apps 1008b are used
31 to facilitate HyperMetal and HyperCoin transactions, the processing engines can also be

1 employed as a service to HyperSphere users creating custom cryptocurrencies or
2 tokenization of service provider businesses.

3 The HyperSphere’s multi-layer cryptocurrency generation is entirely unique and
4 easily distinguished by conventional blockchains processed entirely as an application
5 running in the OS app-envirom above Layer-7 in the OSI protocol stack. For this (and
6 innumerable other reasons), it is more accurate to refer to such conventional blockchain
7 processors as “apps” rather than “protocols” or “networks.” The HyperSphere’s BCP
8 1007, in contrast, can truly be considered a protocol because it exists as part of the SDNP
9 protocol stack, i.e. operating as a network-native operation both in minting new
10 cryptocurrencies or when conducting e-commerce transactions. Semantics aside, because
11 BCP 1007 operation is HyperSphere network-native, blockchain processing is rapid–
12 limited only by the speed of peer consensus rates for transactional validation.

13 Despite its rapid process capabilities, the HyperSphere’s cryptocurrencies are
14 difficult to counterfeit because they employ cryptographic hop codes unique to SDNP
15 network operation not observable from the OSI Session, Presentation, or Application
16 Layers 5, 6 and 7. These codes include a combination of HyperContract information
17 (including the hash of the pledge and a timestamp) and their own unique sequence of
18 HyperNode hop codes. Moreover, because of its purely internal coin generation and a
19 cloaked (undisclosed) jury-of-peers, cryptocurrency counterfeiters are unable to match or
20 predict network-generated blockchain content.

21
22 Aside from its network-native blockchain processing and short-length DyDAG
23 blockchains, another method to improve transactional speed involves the unique use of
24 blockchain defragmentation. In a manner similar to defragging a hard disk drive (HDD),
25 in the process of *blockchain defragmentation* available cryptocurrency is moved to the
26 end of the blockchain at some regular schedule, e.g. at the end of every transaction or
27 every day. By re-locating liquid currency near the bottom of the DyDAG blockchain,
28 subsequent transactional verification requires only very short RBOS segments for
29 confirmation, speeding validation and preventing backtracing altogether.

30 As depicted in **FIG. 54**, during blockchain processing new blockchains are
31 appended only to the end of the chain regardless as to whether they constitute a credit or

1 a debit of assets. Blockchain assets are processed using a last-in first out or LIFO process,
2 where the last acquired coins are used first to minimize transaction time. As shown, at
3 time t_2 currency added at time t_1 (credit 1010) is consumed (shown as credit debit pair
4 1011a and 1011b). At time t_3 an even earlier deposit (credit 1012a) must be found and
5 confirmed to facilitate a current debit 1012b. At time t_6 , the asset needed to fund a current
6 liability (debit 1014b) could involve identifying a deposit from far in the past (credit
7 1014a), i.e. comprising a fragmented transaction 1019 resulting in a long RBOS and slow
8 transaction resolution.

9 The solution to this conundrum is to clean up the blockchain “as you go,”
10 meaning to remove defragmented assets at a convenient time when other transactions are
11 not occurring and speed is not critical. The defragmentation process, shown in the
12 sequence from times t_4 to t_6 , involves identifying stranded assets and relocating these
13 assets to the end of the blockchain. Since added blocks are permanent, there is no means
14 by which to change an earlier entry. Instead, the BC defrag process involves adding
15 “zero” to the blockchain, by recording a debit-credit pair 1015 as shown at time t_4 .
16 During validation, the new debit 1015b will cancel the earlier deposit (credit 1014a),
17 resulting a new asset (credit 1015c) relocated to the end of the chain as shown at time t_5 .
18 Then, when a payment (debit 1016) is made at time t_6 , the asset (credit 1015c) is already
19 located at the end of the blockchain and a compact rapid transaction can occur. Another
20 element of blockchain management is the use of auxiliary blockchains.

21
22 While in communal blockchain arbitrary files are appended onto the main
23 blockchain, with the HyperSphere’s use of DyDAG blockchains, as shown in **FIG. 55**,
24 content can be implemented as an auxiliary sidechain 1021 without disturbing the
25 integrity of the main blockchain 1020.

26 Without the ability to write arbitrary blocks onto the main blockchain, users are
27 prevented from contaminating transactional blockchains with objectionable or illegal
28 content. Instead, the main blockchain 1020 records only a pointer 1022 linking it to an
29 auxiliary sidechain 1021, supporting entries 1023a, 1023b other than cryptocurrency
30 transactions useful for documentation purposes. If the documentation supports a
31 transaction 1024a, such content can be included in an RBOS 1025 for a validation check

1 by independent jurors 1026, and a pledge payment 1028 can be made to the jurors
2 therefor.

3 Once the sidechain is complete, it terminates and records a second entry on the
4 main blockchain, establishing a firm chronology of events by pointers 1027a and 1027b
5 without recording the actual content 1024a. Because the second entry (pointer 1027b)
6 occurs at a different dynamic state than the first (pointer 1027a), the DyDAG sidechain
7 does not form a cyclic loop. The same auxiliary sidechain mechanism can be used for
8 documentation unrelated to cryptographic transactions, and can even be used to invoke
9 subroutine calls of executable code via BC apps. These processes may optionally record
10 updates as to a subroutine's process status on the main blockchain while continuing
11 processes in parallel to the spatiotemporal state of the blockchain, thereby enabling the
12 prospect of executing a *Turing complete* process.

13 ***Secure Wallet:***

14 HyperSphere accounts refer to the device assets, devices, and HyperWallets
15 owned by a particular parental CA-certificate. In particular, HyperSphere accounts of a
16 specific owner include their devices, their registered HyperNodes or HyperNode cluster,
17 their HyperMetal perpetual blockchain, their HyperCoin perpetual blockchain, and their
18 HyperWallets. Any number of intermediary CA-certificates may be used to digitally sign
19 and verify ownership of these elements. All HyperNode income earned by a HyperSphere
20 account owner will reside on their personal HyperCoin blockchain unless transferred into
21 one of several HyperWallets. Additionally, HyperWallets may hold cryptocurrency other
22 than HyperCoins including private company tokens using the HyperSphere as a
23 Blockchain-as-a-Service (BaaS). Aside from minting HyperCoins, all asset transfers in
24 and out of a HyperSphere account occur through OT³ proxy mediator.

26 ***Purchases & OT³ Proxies:***

27 As described previously, all asset transfers in and out of a HyperSphere account
28 are executed using a special transitory blockchain referred to as a one-time-transaction
29 token or OT³ proxy. The proxy exists only during a transaction after which the mediator
30 and its records are irrevocably dissolved. In particular, to prevent theft or backtracing
31 during sale of HyperCoins or when using HyperCoins as payment for online or point-of-

1 sale purchases, no direct blockchain access to the owner's blockchain is allowed. Instead,
2 a two-step transfer process is employed where first the blockchains are moved onto a
3 One-Time Transaction Token mediator or OT³ proxy then in a second step the
4 cryptocurrency is transferred from the proxy to the merchant or buyer in exchange for
5 goods or currency (crypto or fiat). During all OT³ proxy mediated transactions, the first
6 step requires the payor, the HyperCoin holder, to request moving a specified number of
7 HyperCoins from their account or HyperWallet to the OT³ proxy.

8 The process, shown in **FIG. 56**, commences by the requestor opening an OT³
9 transfer HyperContract. The HyperContract then identifies a jury-of-peers (e.g. jurors
10 1051 with a blockchain 1044) and creates a replicant blockchain observer segment
11 (RBOS) 1043 from the owner's HyperCoin blockchain (or HyperWallet) 1040 of
12 sufficient length to verify the payor holds adequate assets to execute the requested
13 transaction. Once verified, the requested HyperCoins are debited from the owner's
14 perpetual HyperCoin blockchain 1040 and credited onto the transitory OT³ blockchain
15 1041. Because the payor 1050 cannot see the cloaked jurors 1051, they are unable to
16 execute a 51%, cyberbot, or Sybil attack to engage in double spending because they don't
17 know the jurors who are checking the RBOS blockchain. Similarly, the payor 1050
18 cannot subvert or corrupt the 1040 RBOS data.

19 The next step is to confirm the sincerity of the payee, either the merchant selling
20 good and services, or the HyperCoin purchaser. This can be accomplished in person for
21 POS transactions, through an escrow agent (for real property) or by time-locking the OT³
22 proxy's release till the transaction settles, e.g. until the validity of a Bitcoin payment can
23 be confirmed. After the transactional integrity is confirmed the OT³ proxy 1041 transfers
24 the HyperCoin digital code to the merchant or buyer (e.g. paid for a rideshare (arrow
25 1042a), paid to a POS terminal (arrow 1042b), or transacted online (arrow 1042c)), and
26 the proxy is closed. In this manner through the OT³ proxy neither party directly interacts
27 and is unable to commit fraud or backtracing. The proxy mediator also speeds
28 transactional resolution because the slower blockchain verification and transfer process
29 can precede the actual e-commerce transaction. Lastly, the OT³ proxy limits the total
30 assets at risk for transactional fraud because the HyperSphere account holder never
31 exposes their personal HyperCoin blockchain or HyperWallet.

1
2 ***Replicant Blockchain Observer Segments (RBOS):***

3 To manage transactional integrity while preventing personal identity theft or
4 leakage from an account owner's blockchain, the HyperSphere uses a unique and
5 inventive method referred to as a *replicant blockchain observer segment* (RBOS),
6 introduced here for the first time. Comprising a limited length copy of a host's
7 blockchain, the RBOS is sufficiently long to authorize a transaction but too short to
8 enable backtracing of prior history or inadvertently result in privacy leakage.

9 An example of the use of an RBOS 1063 for juror consensus in HyperSpheric
10 transactions is shown in **FIG. 57**. Any given transaction can employ more than one
11 RBOS to support any size jury-of-peers. After a transaction's completion, its
12 corresponding RBOS 1063 is destroyed and the hashed blockchain recorded, protecting
13 privacy while ensuring transaction integrity and traceability while preventing double
14 spending.

15
16 ***Cloaked Jurors:***

17 Another unique feature of the HyperSphere is its ability to establish *ad hoc*
18 tunneling communication, i.e. dynamic single-hop VPNs, between a HyperNode source
19 portal and a remote portal. The purpose of these private tunnels is to divert traffic away
20 from subnets suffering QoS degradation from congestion and to avert cyberattacks on the
21 network or on blockchain transactions such as DoS or Sybil attacks. The method can also
22 be used to ensure hypersecure communication over uncontrolled Last Mile links.

23 As shown in **FIG. 58**, once a HyperNode inter-portal tunnel 1082 is established
24 data can flow using direct routing to the remote portal 1081 unprocessed by intermediate
25 nodes, much like an express train passes through local train stops without stopping (or
26 even slowing down). Application of HyperNode tunnel communication is especially
27 valuable in repelling cyber-assaults. For example, in the event of a rapid rise in localized
28 network congestion where a denial-of-service-attack is suspected, the node under attack
29 can temporarily suspend incoming packet support (or optionally open a queue buffer),
30 establish a tunnel 1082 beyond the reach of attacked device or subnet, then reestablish all
31 ongoing sessions redirecting traffic with other nodes 1084a to and from the remote portal

1 1081. While this response methodology won't prevent DoS from delaying the
2 establishment of new incoming calls and sessions, it allows the surrounded node to
3 establish open new links to safety on a priority basis, even when the source node 1080 is
4 surrounded by cyberbots 1083a.

5 Because cyberbots 1083 lack the dynamic security credential to interpret the
6 SDNP protocol, they cannot trace the location of a remote HyperNode portal 1081.
7 HyperNode tunneling is especially valuable in protecting cryptocurrency transactions to
8 avoid blockchain attacks such as Sybil, 51%, and DoS methods. By specifying cloaked
9 jurors in a HyperContract, the transacting parties are unaware of which HyperNodes are
10 performing asset and transaction consensus validation. Moreover, by connecting to
11 cloaked jurors through a HyperNode tunnel 1082, their inter-portal communication is
12 privileged and not subject to metadata surveillance and hacking by other network nodes.
13 HyperNode tunneling is automatically executed by any HyperNode seeking exceptional
14 transactional security or upon detecting a DoS assault. Once an attack is detected, tunnel
15 traffic is assigned priority over all local traffic. Ongoing sessions are reinitiated through
16 the remote node without any knowledge of the cyberbot attackers

17 The HyperSphere also supports tunneling executed on an end-to-end basis. Unlike
18 inter-portal tunneling, in end-to-end tunneling the communicating parties exchange
19 cryptographic keys prior to and unrelated to the opening of a session or placing a call.
20 Ideally, the keys can be exchanged between two devices in person without ever
21 employing an intervening network. The application of end-to-end encryption facilitates
22 personal privacy in the HyperSphere independently from the SDNP's security protocols.

23 Multidimensional properties of the HyperSphere include the following:
24

- 25 • Using quantum computing for brute force code breaking of a data packet is
26 rendered meaningless by the incomplete data contained within the packet, lacking of
27 metadata useful in identifying related datagrams in a sea of network traffic.
- 28 • Using quantum computing for brute force code breaking of a data packet
29 doesn't help in decrypting other data packets since security credentials and concealment
30 algorithms are dynamic, changing faster than they can be broken.

- 1 • The HyperSphere's metamorphic HyperNode's are stateless, meaning they
2 forget what they have done immediately after they execute any task, leaving no record
3 to inspect.
- 4 • Data transport occurs over a meshed network, secured on a hop-by-hop basis
5 using dynamic concealment methods, meaning there are no master keys able to inspect
6 data traffic, content, or even metadata.
- 7 • Since routing is dynamic, traveling through the network at near the speed of
8 light, a hacker's intervention (also traveling at the same speed) can never catch the
9 packet it is chasing. By the time the hacker's packet arrives at a HyperNode, the state of
10 the DyDAG meshed routing has changed, metaphorically it's like reading yesterday's
11 weather report.
- 12 • The application of DyDAG transitory blockchains (tBC) in HyperContract
13 execution are stateless– destroyed after each task is completed, so supercomputing
14 cannot be used to break a record that has already been destroyed.
- 15 • DyDAG perpetual blockchains (BC) are privacy protected by a multi-tree
16 identity-trust-chain using pseudonymous identities, meaning no means exists to link the
17 pseudonymous owner to their true identity CA-certificate.

18
19 In general, the HyperSphere employs time and state-based dynamic changes in its
20 network operations, packet transport, and security credentials to greatly reduce the
21 probability of a successful intrusion into the SDNP spatiotemporal meshed network or
22 against HyperSpheric transactional processing. This does not mean that any given
23 datagram might not be code-broken, but that the damage of the attack is limited because
24 of the packet's limited content, short lifespan, and lack of contextual metadata.

25 26 ***Sequential Quantum Key:***

27 To recover lost root CA-certificate and restore corrupted account identities, the
28 HyperSphere includes, as a last resort, a unique cryptographic device introduced herein as
29 a sequential quantum key or SQK. The SQK, properly decoded, gives its owner the
30 ability to open and restore their root CA-certificate to reclaim rightful ownership of
31 corrupted accounts. Built on the principle of the *quantum observer effect*, which states

1 that by very act of watching, an observer affects the observed reality, in a sequential
2 quantum key, not only must the contents of the key be faithfully reproduced, the
3 sequence in which each cell is viewed and entered must be executed in a precise way
4 (metaphorically as a multidimensional Rubik's cube). If the sequence is executed
5 incorrectly, the proper combination will never appear. The observer effect makes brute
6 force attacks more difficult while allowing users to separate passphrase archiving and
7 sequencing in unrelated media content.

9 **Cryptoeconomics**

10 While there is no generally accepted definition, in the context of this paper we use
11 the term 'cryptoeconomics' in its simplest interpretation to mean e-commerce involving
12 the use of cryptocurrency. In that context, cryptocurrency based trading and e-commerce
13 includes cryptographic currency generation, decentralized financial transactions, juror
14 consensus-based transaction validation, smart contract execution, and crypto-incentives.
15 Over the last decade, the advent of cryptocurrency, blockchain technology, and smart
16 contracts has had a profound and beneficial impact on business, especially involving fund
17 raising, venture capital, banking, capital decentralization, and challenging corporate
18 hegemony for information control. Despite these accomplishments, the application and
19 commercial acceptance of cryptocurrency and blockchain technology by major business
20 sectors, especially risk adverse ones, has not progressed as anticipated. A number of
21 issues plague today's crypto based economic models, as describe below.

23 ***1) The Problems with Crypto-economy 1.0***

24 All present day cryptocurrency based commerce, i.e. cryptoeconomics 1.0, relies
25 on certain fundamental commonalities and associated deficiencies and vulnerabilities,
26 namely:

- 28 • All transactions occur over the Internet and are subject to inherent security
29 vulnerabilities (e.g. man-in-the-middle attacks, trust attacks, viral infections).

1 • There is no certain way to determine if CA-certificate based identification is
2 stolen or fraudulent or to prevent malware infections (delivered via corrupted
3 certificates).

4 • Nearly one hundred fifty attacks on blockchains are known, with many
5 involving theft, fraud and double spending.

6 • The vast majority of cyber and cryptocurrency crimes go unpunished,
7 especially those involving multinational hacks, in part because legal authorities are ill
8 equipped to deal with cyber-crime, and also because of questions of jurisdiction.

9 • Conspiratorial and cyberbot peer voting control enables a variety of exploits,
10 51% attacks, and Ponzi schemes to be perpetrated on unsuspecting coin holders and
11 traders.

12 • Corrupted smart-contracts can be used to commit fraud or Ponzi schemes
13 without repercussion.

14 • Conventional cryptocurrency generation is by its very design fundamentally
15 energy inefficient and ecologically harmful.

16
17 The last bullet highlights the ecological challenges with crypto-economy 1.0. In
18 essence, Proof-of-Work, the basis for Bitcoin mining was originally created not for
19 cryptocurrency but to thwart denial-of-service attacks by forcing attackers to waste
20 energy and spend money to launch an attack. Unfortunately, the same mechanism is now
21 used to generate every PoW cryptocurrency (and to no surprise) wasting vast amounts of
22 energy. Despite its publication buzz, alternatives such as Proof-of-Stake consensus based
23 cryptocurrencies are not seeing widespread adoption. Aside from the foregoing concerns,
24 cryptoeconomically several major fundamental flaws plague crypto-economy 1.0,
25 including:

26
27 • Proof-of-Work generation of cryptocurrency relies on an artificially created
28 coin shortage (limited production with diminishing rewards) with countdowns fueling
29 price volatility and inviting exploits.

- 1 • Hard forks off of a blockchain intended to ‘subvert’ blockchain integrity
2 with fraudulent blocks creates intense legal controversy between valid transactions
3 competing on the main chain and side forks.
- 4 • Cryptocurrency today doesn’t really do anything– aside from cryptocurrency
5 mining and trading, using hash-nonce puzzle solving is unrelated to e-commerce or in
6 delivering any real product.
- 7 • Long overweight blockchain cryptocurrencies suffer slow (and ever
8 worsening) transaction rates incompatible with today’s need for rapid e-commerce.
- 9 • Repeated use of direct blockchain payments exposes the blockchain or wallet
10 to backtracing and theft.
- 11 • Present day smart contracts require unanimous consensus, thereby limiting
12 the transaction rate serially by the slowest participant.
- 13 • Volatility in PoW cryptocurrencies, especially Bitcoin and Ether, prevent
14 those businesses not interested in trading, from holding cryptocurrencies, forcing them
15 to buy cryptocurrency only at the time of a trade.
- 16 • Successful blockchain attacks occur with tremendous regularity. Because
17 skilled perpetrators gain control over entire blocks of the blockchain, coin owners
18 irrevocably lose access to their own cryptocurrency to an unknown assailant with no
19 legal recourse or means by which to recover their losses or seek justice or even to
20 identify the perpetrator.

21
22 The last two bullet points highlight the risk of the current state of
23 cryptoeconomics, specifically merchants and businesses can’t procure and hold
24 cryptocurrency because (i) it is too volatile in price, and (ii) it is too likely to get stolen if
25 held for extended periods of time. This forces cryptocurrency users to simultaneously buy
26 and use their digital assets only in spot trades or concurrent buy-sell transactions– hardly
27 desirable features for the alleged currency of the future. Another issue is one of currency
28 stability. A relentless appreciation in the average price of cryptocurrency coupled with
29 trading based volatility are particularly disconcerting for businesses considering
30 cryptocurrency as medium for commerce because they destabilize an enterprise’s
31 operating costs. One oft-cited apologue (intended to inspire Bitcoin investors)– the

anecdote of the ‘fifty million dollar pizza’ recounts the allegedly true story of a 2010 purchase of two pizzas for 10,000 Bitcoins, a purchase by November 2017 standards equal to \$50,000,000 per pizza. As a post scriptum, six months later Bitcoin lost 25% of its value. Although such volatility may inspire speculators seeking quick riches, for businesses and enterprises the prospects of a wildly unstable currency is horrific, invariably limiting its utility and preventing any real commercial adoption.

2) *Dual Cryptocurrency e-Commerce*

To avoid the pitfalls of “Crypto-economy 1.0”, the HyperSphere’s cryptoeconomic system employs a uniquely implemented dual cryptocurrency system with personal blockchain ownership, i.e. blockchains not shared with other users. To ensure transactional integrity and to establish trust among peers, the HyperSphere’s blockchain is inextricably linked to other blockchains in a dynamic (DyDAG) multi-tree structure. Cryptoeconomics in the HyperSphere is simple– no mining or hash-nonce puzzle solving is required. Service providers & merchants issue HyperContracts to obtain communication, computing, and storage capability using either fiat currency or using HyperCoins.

As shown in **FIG. 59**, in the HyperSphere 1100 resource providers (HyperNode owners) earn HyperCoins for supplying resources to merchants, either by *minting* 1102a or *melting* (recycling) 1102b. In HyperContracts 1104 for HyperCoin ‘minting’ 1102a, merchants 1103 contractually pledge payment in HyperMetal 1105 using HyperMetal 1105 purchased with fiat currency 1110 (via a HyperSphere exchange 1111). Upon successful contract completion resource providers (HyperNodes 1101) automatically mint the released HyperMetal pledge into HyperCoins 1112, ratably apportioned in accordance with each HyperNode’s relative contribution. Alternatively, in HyperContracts 1104 for HyperCoin ‘recycling’, merchants contractually pledge payment in HyperCoins (using HyperCoins 1112b earned or purchased previously). Upon successful contract completion resource providers melt and recycle original HyperCoins 1112b into new HyperCoins 1112, ratably apportioned in accordance with each HyperNode’s relative contribution.

The newly generated HyperCoins contain new digital cryptographic identities in order to maintain small files and rapid transactions. Through recycling, HyperCoin

holders and consumers have the ability to immediately use earned or purchased HyperCoin cryptocurrency 1112b to directly pay merchants without requiring digital currency exchanges, thereby avoiding expensive exchange fees. In contrast, using minting HyperContracts, merchants can hold or use HyperMetal 1105 to conduct their business without risking the market volatility of HyperCoin trading and speculation.

3) *HyperSphere Cryptoeconomics*

With its unique method of cryptocurrency generation, its dual-cryptocurrency system, and its utility in facilitating substantive e-commerce, the HyperSphere overcomes the fundamental limitations of Crypto-economy 1.0, namely:

- The HyperSphere delivers meaningful e-commerce including cloud-based communication, computing, storage, device-to-cloud connectivity, and e-services over a user-community owned cloud operating in a fully decentralized manner
- HyperSpheric cryptocurrency comprises embedded network-native digital tokens used for engaging in meaningful e-commerce in the HyperSphere without the complications of fiat-currency based transactions
- HyperSpheric cryptocurrency generation does not involve mining– solving useless puzzles to create digitally assets of artificial scarcity
- HyperCoins are automatically earned by any person, company, or entity hosting HyperNodes on their devices (or cluster of devices), with remuneration paid at the successful completion of any HyperContracts in which a HyperNode participates
- HyperNodes are BYOD friendly, allowing business and personal accounts to co-exist on a common platform with no comingling of files or data and requiring no capital investment (aside from personally owned devices)
- Unlike Bitcoin miners, HyperCoin minters aren't required to consume massive amounts of energy to generate cryptocurrency or to risk capital investment on mining equipment with no certain reward or ROI
- Merchants and service providers are able to buy, store and gradually use HyperMetal to fund their HyperSpheric operations with minimal risk of

1 volatility– thereby circumventing HyperCoin price volatility’s impact on network
2 use

- 3 • Risk tolerant traders, investors, and speculators are free to buy and sell
4 HyperCoins without restriction, potentially benefiting from HyperCoin price
5 fluctuations and volatility

6

7 An economic model for the HyperSphere, albeit in simplified form, is represented
8 schematically in **FIG. 60**, mechanistically illustrating economic growth drivers through
9 both capital infusion and through HyperCoin recycling. As shown, the HyperSphere 1100
10 generates economic value through the operation of its HyperNodes– resource providers
11 performing tasks to complete HyperContracts. Economic growth drivers of HyperSpheric
12 operations are manifest in two ways: (i) as an increase in enterprise value ΔEV of
13 merchants and service providers realized through the creation of products or the delivery
14 of services, and (ii) in minting of new HyperCoins (HC) paid to HyperNode owners. This
15 means service providers and resource providers (the community of HyperSphere users)
16 are the economic beneficiaries of HyperSpheric e-commerce, not telecom or social media
17 corporations, and not the HyperSphere platform. Economic gain of the HyperSphere
18 occurs using two processes– minting and recycling (melting).

19 In minting, new capital infusion into the HyperSphere in the form of fiat currency
20 1110 is converted into HyperMetal 1105 and then “amplified” by a HyperNode 1130 to
21 create added value comprising enterprise value ΔEV (a product or service 1131) and
22 HyperCoin value V_{HC} 1132, algebraically as $(\Delta EV + V_{HC})$. At the time of minting,
23 HyperCoins have an immediate HyperCoin value V_{HC} based on the number of coins
24 received HC and the free market’s HyperCoin exchange rate HXR, i.e. whereby $V_{HC} =$
25 $(HC \cdot HXR)$. If the coins 1132a are concurrently sold (arrow 1134) as they are earned by
26 HyperNode 1130, then volatility in the HyperCoin trading price HXR, i.e. its exchange
27 rate into specific country’s fiat currency, is not a factor in the fiat profit earned by the
28 HyperNode’s owner.

29 Illustrated schematically, value creation through minting depends only on the
30 quantity of HyperMetal (HM) 1105 pledged to a HyperNode 1130 (shown as the input to
31 a difference amplifier); and by the amplifying value-gain A_v of the HyperNode minting

new HyperCoins (a reward value specified in each negotiated HyperContract). Since the HyperMetal purchase price in digital currency exchange 1111 is based on the (relatively stable) bit cost of the network, then volatility in HyperCoin trading price has no effect on earned HyperCoins (a behavior represented graphically as common-mode noise not amplified by the difference amplifier). To summarize, in minting, HyperMetal 1105 is converted into a varying number of HyperCoins 1132 (the system's gain) based on supply-demand market dynamics.

The second economic input to the HyperNode 1130 amplifier, 'recycling HyperCoins', does not result in production of new HyperCoins, but instead melts and recycles old coins 1132b into new coins of lesser quantity. Shown as a negative input to the HyperNode 1130, recycling (arrow 1135) decreases the number of HyperCoins in circulation, enabling merchants to engage in commerce without infusing new capital into the HyperSphere. During recycling, the initial quantity of HyperCoins (HC) is reduced to a smaller quantity (β HC) where β ranges from 0% to 100% describing the fraction of HyperCoins retained, i.e. paid to resource providers and remaining in circulation.

The corresponding value change of HyperCoins in circulation resulting from recycling is then given by $\Delta V_{HC} = (HC \cdot HXR) (1 - \beta)$, a portion of which is delivered as products or services to merchants and service providers. The change in enterprise value, while proportional to the change in HyperCoin value $\Delta EV \propto \Delta V_{HC}$, does not obey a fixed ratio, but varies with market conditions. In summary, recycling does not grow the HyperSphere's economy but facilitates e-commerce in recessionary periods. Melting (recycling) also serves as negative economic feedback, cryptoeconomically-improving stability by steadily taking a fraction of HyperCoins out of circulation.

By combining the stabilizing impact of HyperCoin melting with the ability to generate new currency in response to economic growth enables the HyperSphere to support dynamic e-commerce in a stable closed-loop system while remaining insensitive to fluctuations in the HyperCoin's trading price. In times of economic expansion, new capital injections by merchants seeking resources increase the HyperSphere's money supply of both HyperMetal 1105 and HyperCoin 1132 to support growing market demand. In periods of economic recession when new capital infusions dwindle, the HyperSphere economy can naturally contract in response, consuming unused HyperMetal

1 1105 and ultimately recycling HyperCoins 1132b, reducing the coins in circulation and
2 helping stabilize the currency against market sell-offs. In this manner, the HyperSphere
3 cryptoeconomically operates more similarly to a country than it does as a business
4 enterprise.

5
6 ***Community:***

7 Far beyond regulatory and legislative initiatives intended to prohibit monopolistic
8 control of information and discourage the unfair sequester of network bandwidth, the
9 HyperSphere economically represents the quintessential embodiment of ‘net neutrality’.
10 In operation, the HyperSphere’s decentralized cloud network is owned and enabled by
11 companies and everyday citizens hosting its HyperNodes, making the HyperSphere truly
12 the people’s network. In this regard the HyperSphere is wholly unique and incomparable
13 to present day cryptoeconomics.

14 Individuals, companies, and research institutions owning multiple servers,
15 computers, and mobile devices, can download HyperNode portal software onto each of
16 the component devices then link them as a ‘cluster’ into one account owned by a single
17 identity-trust-chain. The advantage of clustering devices is twofold: first, all HyperCoin
18 cryptocurrency proceeds earned by the cluster can be assigned to a single blockchain and
19 used for the group’s greater good. Secondly, communication and tasks executed in
20 clustered devices behave as a single connected device without any need for remuneration
21 paid to the HyperSphere for establishing the secure interconnectivity. In this manner, a
22 company can create and operate a virtual private corporate network within the public
23 HyperSphere network, functioning privately but with the ability to solicit resources or
24 earn cryptocurrency when the opportunity arises. Another opportunity for device
25 clustering is in gaming communities. Commanding significant graphics and computing
26 bandwidth and processing capacity, the global gaming community can also benefit from
27 device clustering, creating *ad hoc* societies for specific games with their own public-
28 private networks running over the HyperSphere. In such cases the HyperSphere can also
29 be used as a tokenization platform for gamers, separate and distinct from HyperCoin and
30 HyperMetal cryptocurrency.

1 The HyperSphere also supports collaboration. For example, two universities, each
2 operating their own HyperNode clusters, can engage one another's assistance and
3 cooperative support through a HyperContract. As such, HyperContracts defining the
4 circumstances of cooperation specify when a university can borrow computing capacity
5 from another, and the compensation to be paid. With the HyperSphere, idle capacity on
6 server farms, Bitcoin mining farms, and supercomputers need not be wasted, but instead
7 may contribute to solving complex computationally intensive challenges facing the planet
8 and its inhabitants. Finally, the HyperSphere is able to support charitable or non-profit
9 organizations, e.g. providing a mechanism for soliciting and collecting support
10 cryptoeconomically. Contributions can be spent on paying for services or be used to fund
11 HyperNode hosting platforms able to generate more cryptocurrency assets than the
12 original contributions, i.e. to 'amplify' charitable contributions for greater benefit.

CLAIMS

1. A method of transmitting packets containing digital data through a cloud, each of the packets comprising a plurality of data segments, the cloud comprising a network of undifferentiated metamorphic nodes, said nodes being hosted on servers or communication devices, the method comprising:

causing an undifferentiated metamorphic node to transform into a first task node;

causing said first task node to receive packets from other task nodes in the cloud;

causing said first task node, after receiving packets from other task nodes, to transmit packets to other task nodes in the cloud, whereby the packets transmitted by said first task node comprise at least a portion of data segments of packets previously received by said first task node;

causing said first task node to perform at least one of the following concealment methods:

scrambling an incoming packet by changing an order of the data segments in the incoming packet in accordance with a scrambling algorithm, said scrambling algorithm being created in accordance with a first state;

encrypting the incoming packet in accordance with an encryption algorithm and an encryption key, said encryption algorithm and said encryption key being created in accordance with said first state; and

causing said first task node to transform back into an undifferentiated metamorphic node after said first task node transmits said packets to said other task nodes.

2. The method of Claim 1 further comprising:

causing said first task node to both scramble and encrypt the incoming packet to produce an encrypted scrambled packet.

3. The method of Claim 2 further comprising:

causing said first task node to transmit the encrypted scrambled packet to a second task node;

causing the second task node to decrypt the encrypted scrambled packet using said algorithm and encryption key in accordance with said first state to recover the scrambled packet;

causing the second task node to encrypt the scrambled packet in accordance with an algorithm and encryption key created in accordance with a second state so as to produce a second encrypted scrambled packet;

causing the second task node to transmit the second encrypted scrambled packet to a third task node;

repeating in other task nodes the process of decrypting an incoming scrambled packet using a prior state to recover a scrambled packet and then re-encrypting the packet with a new state to generate a newly encrypted scrambled packet;

decrypting for a final time an incoming encrypted scrambled packet in a last task node using security credentials in accordance with a prior state in accordance with which the packet was encrypted to recover a scrambled packet; and

unscrambling the incoming packet in the last task node so as to recreate said packet prior to said scrambling in said first task node.

4. The method of Claim 2 wherein said scrambling algorithm is determined by a value of a state, a time, a geographic zone or a combination thereof.

5. The method of Claim 1 further comprising:

causing said first task node to perform a series of concealment operations on data in the packet, said concealment operations comprising scrambling, encryption, and insertion of junk data into the packet;

causing the first task node to transmit the packet on which said concealment operations have been performed to a second task node; and

causing the second task node to perform the inverse of said concealment operations on the packet, said inverse comprising descrambling, decryption, and removal of junk data and being performing in the reverse of the order in which said concealment operations were performed in the first task node,

wherein each of the concealment operations and the inverse of the concealment operations is performed in accordance with an algorithm, key or other security credential, said algorithm, key or other security credential being determined by a value of a state, a time at which said concealment operations were performed in said first task node, a geographic zone or a combination thereof.

6. The method of Claim 5 wherein the first task node accesses the algorithm, key or other security credential from a diffuse data cloud of disaggregated data.

7. The method of Claim 1 wherein a packet is routed through the cloud using the following sequence of steps:

a caller edge device provides an undifferentiated metamorphic node with an address of a callee edge device;

the metamorphic node transforms into an authority node;

the authority node develops a route for the packet through the cloud, the route comprising metamorphic nodes on the route, and informs each of metamorphic nodes on the route tasks to be performed in routing the packet through the cloud;

the authority node transforms back into an undifferentiated metamorphic node;

upon receiving the tasks to be performed from the authority node, each of the metamorphic nodes on the route transforms into a task node; and

after performing the tasks in routing the packet through the cloud, each of the task nodes transforms back into an undifferentiated metamorphic node.

8. The method of Claim 7 wherein a task node receives from the authority node no addresses of nodes on the route other than an address of a next node on the route.

9. The method of Claim 7 wherein, after a task node transforms back into an undifferentiated metamorphic node, no record of the tasks performed by the task node is retained in the undifferentiated metamorphic node into which the task node transforms.

10. The method of Claim 7 wherein, after the authority node transforms back into an undifferentiated metamorphic node, no record of nodes on the route is retained in the undifferentiated metamorphic node into which the authority node transforms.

11. The method of Claim 7 wherein the authority node develops a route for the packet through the cloud by using network propagation delay maps.

12. The method of Claim 7 wherein:

in developing a route for the packet through the cloud the authority node contacts an undifferentiated metamorphic node with a request for addresses of undifferentiated nodes on the route;

the undifferentiated metamorphic node contacted by the authority node transforms into a name server node;

the name server node obtains the requested addresses of undifferentiated metamorphic nodes on the route from a diffuse data cloud and delivers the requested addresses to the authority node; and

the name server node transforms back into an undifferentiated metamorphic node.

13. The method of Claim 7 wherein a client gateway node hosted on the caller edge device initiates a call or session using the following sequence of steps:

the client gateway node contacts an undifferentiated metamorphic node with a request for an address of the callee edge device;

the undifferentiated metamorphic node contacted by the client gateway node transforms into a name server node;

the name server node obtains the requested address of the callee edge device from a diffuse data cloud and delivers the requested address to the client gateway node; and

the name server node transforms back into an undifferentiated metamorphic node.

14. The method of Claim 13 wherein, after the name server node transforms back into an undifferentiated metamorphic node, no record of the addresses obtained by the name server node from the diffuse data cloud is retained in the undifferentiated metamorphic node into which the name server node transforms.

15. The method of Claim 7 wherein:

the caller edge device encrypts data in the packet in accordance with a cryptographic key;

the caller edge device transmits the cryptographic key used to encrypt the data in the packet to the callee edge device; and

the callee edge device uses the cryptographic key transmitted by the caller edge device to decrypt the data in the packet.

16. The method of Claim 1 further comprising:

causing said first task node to scramble an incoming packet by changing an order of the data segments in the packet in accordance with a first scrambling algorithm so as to produce a scrambled packet;

causing said first task node to split said scrambled packet into multiple scrambled packets;

causing said first task node to encrypt each of the multiple scrambled packets so as to produce multiple encrypted scrambled packets and to transmit the multiple encrypted scrambled packets to other task nodes; and

causing a final task node to receive encrypted scrambled packets, decrypt each the received encrypted scrambled packets so as to produce multiple received scrambled packets, and assemble the received scrambled packets thereby to recreate the scrambled packet produced in the first task node; and

causing the final task node to unscramble the scrambled packet produced in the first task node thereby to recover the incoming packet received by the first task node.

17. A method of routing data packets over a communication network, said communication network comprising a plurality of nodes, the method comprising:

subdividing data into one of the following categories:

data used to perform a name server function, said name server function comprising identifying device names and their corresponding dynamic addresses;

data used to perform an authority node function, said authority node function comprising developing routes of the data packets over a network of task nodes; and

data used to perform a task node function, said task node function comprising concealing the content of data packets using concealment algorithms;

providing the nodes with data needed to perform one of said name server function, said authority node function, or said task node function;

fragmenting each of the categories of data into files in accordance with an algorithm, said algorithm being based on a state such as time, wherein a key is required in order to defragment some or all of the files; and

causing a node to perform at least one of the following concealment methods:

scrambling an incoming packet by changing an order of the data segments in the incoming packet in accordance with a scrambling algorithm, said scrambling algorithm being created in accordance with a first state;

encrypting the incoming packet in accordance with an encryption algorithm and an encryption key, said encryption algorithm and said encryption key being created in accordance with said first state.

18. The method of claim 17 further comprising storing the fragmented data in multiple devices, no single device containing sufficient content to reconstruct the data prior to the fragmentation.

19. The method of Claim 17 wherein a gateway node in the network requests a destination address from the data used to perform a name server function.

20. The method of claim 17 wherein a gateway node delivers data used to perform a name server function to an authority node in order to enable the authority node to access data in a diffuse data cloud, including a table of node-to-node hop times.