

SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR SECURE APPLICATION DISTRIBUTION

BACKGROUND

1. Field

[0001] This disclosure relates generally to providing executable applications and, in some non-limiting embodiments or aspects, to systems, methods, and computer program products for secure application distribution.

2. Technical Considerations

[0002] There are numerous technical challenges and considerations to deliver a trusted application to untrusted devices (such as end-user devices) in a tightly-controlled manner. This arises from serving applications from distributed systems (e.g., multiple servers located in geographically dispersed datacenters), seeking constant uptime, and requiring a consistency between application code and data structured across a full lifecycle of interactions with the served code. Partial rollouts of new versions, infrastructure errors, and other like events can compromise consistency and accuracy.

SUMMARY

[0003] According to non-limiting embodiments or aspects, provided is a system comprising at least one processor programmed or configured to: receive, from a requesting system, an initial request for an application package; determine application data based on the initial request; generate a cryptographic token based on the application data; generate at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL; communicate the at least one link to the requesting system; and serve the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

[0004] In non-limiting embodiments or aspects, the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof. In non-limiting embodiments or aspects, the application data comprises an application version and an origin URL, wherein serving the application package comprises: generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL. In non-limiting embodiments or aspects, generating the cryptographic token comprises encrypting the application data with at least one encryption key. In non-limiting embodiments or

aspects, serving the application package comprises: decrypting the cryptographic token in response to the requesting system accessing the at least one link; modifying the content path based on the decrypted cryptographic token; and processing the modified content path. In non-limiting embodiments or aspects, the at least one processor is further programmed or configured to: communicate, to the requesting system, a data integrity value generated based on the application package.

[0005] According to non-limiting embodiments or aspects, provided is a computer-implemented method comprising: receiving, from a requesting system, an initial request for an application package; determining application data based on the initial request; generating a cryptographic token based on the application data; generating at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL; communicating the at least one link to the requesting system; and serving the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

[0006] In non-limiting embodiments or aspects, the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof. In non-limiting embodiments or aspects, the application data comprises an application version and an origin URL, wherein serving the application package comprises: generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL. In non-limiting embodiments or aspects, generating the cryptographic token comprises encrypting the application data with at least one encryption key. In non-limiting embodiments or aspects, serving the application package comprises: decrypting the cryptographic token in response to the requesting system accessing the at least one link; modifying the content path based on the decrypted cryptographic token; and processing the modified content path. In non-limiting embodiments or aspects, the method further comprises communicating, to the requesting system, a data integrity value generated based on the application package.

[0007] According to non-limiting embodiments or aspects, provided is a computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: receive, from a requesting system, an initial request for an application package; determine application data based on the initial request;

generate a cryptographic token based on the application data; generate at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL; communicate the at least one link to the requesting system; and serve the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

[0008] In non-limiting embodiments or aspects, the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof. In non-limiting embodiments or aspects, the application data comprises an application version and an origin URL, wherein serving the application package comprises: generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL. In non-limiting embodiments or aspects, generating the cryptographic token comprises encrypting the application data with at least one encryption key. In non-limiting embodiments or aspects, serving the application package comprises: decrypting the cryptographic token in response to the requesting system accessing the at least one link; modifying the content path based on the decrypted cryptographic token; and processing the modified content path. In non-limiting embodiments or aspects, the program instructions further cause the at least one processor to: communicate, to the requesting system, a data integrity value generated based on the application package.

[0009] Other non-limiting embodiments or aspects will be set forth in the following numbered clauses:

[0010] Clause 1: A system comprising at least one processor programmed or configured to: receive, from a requesting system, an initial request for an application package; determine application data based on the initial request; generate a cryptographic token based on the application data; generate at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL; communicate the at least one link to the requesting system; and serve the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

[0011] Clause 2: The system of clause 1, wherein the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof.

[0012] Clause 3: The system of clauses 1 or 2, wherein the application data comprises an application version and an origin URL, wherein serving the application package comprises: generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL.

[0013] Clause 4: The system of any of clauses 1-3, wherein generating the cryptographic token comprises encrypting the application data with at least one encryption key.

[0014] Clause 5: The system of any of clauses 1-4, wherein serving the application package comprises: decrypting the cryptographic token in response to the requesting system accessing the at least one link; modifying the content path based on the decrypted cryptographic token; and processing the modified content path.

[0015] Clause 6: The system of any of clauses 1-5, wherein the at least one processor is further programmed or configured to: communicate, to the requesting system, a data integrity value generated based on the application package.

[0016] Clause 7: A computer-implemented method comprising: receiving, from a requesting system, an initial request for an application package; determining application data based on the initial request; generating a cryptographic token based on the application data; generating at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL; communicating the at least one link to the requesting system; and serving the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

[0017] Clause 8: The computer-implemented method of clause 7, wherein the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof.

[0018] Clause 9: The computer-implemented method of clauses 7 or 8, wherein the application data comprises an application version and an origin URL, wherein serving the application package comprises: generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL.

[0019] Clause 10: The computer-implemented method of any of clauses 7-9, wherein generating the cryptographic token comprises encrypting the application data with at least one encryption key.

[0020] Clause 11: The computer-implemented method of any of clauses 7-10, wherein serving the application package comprises: decrypting the cryptographic

token in response to the requesting system accessing the at least one link; modifying the content path based on the decrypted cryptographic token; and processing the modified content path.

[0021] Clause 12: The computer-implemented method of any of clauses 7-11, further comprising: communicating, to the requesting system, a data integrity value generated based on the application package.

[0022] Clause 13: A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: receive, from a requesting system, an initial request for an application package; determine application data based on the initial request; generate a cryptographic token based on the application data; generate at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL; communicate the at least one link to the requesting system; and serve the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

[0023] Clause 14: The computer program product of clause 13, wherein the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof.

[0024] Clause 15: The computer program product of clauses 13 or 14, wherein the application data comprises an application version and an origin URL, wherein serving the application package comprises: generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL.

[0025] Clause 16: The computer program product of any of clauses 13-15, wherein generating the cryptographic token comprises encrypting the application data with at least one encryption key.

[0026] Clause 17: The computer program product of any of clauses 13-16, wherein serving the application package comprises: decrypting the cryptographic token in response to the requesting system accessing the at least one link; modifying the content path based on the decrypted cryptographic token; and processing the modified content path.

[0027] Clause 18: The computer program product of any of clauses 13-17, wherein the program instructions further cause the at least one processor to: communicate, to

the requesting system, a data integrity value generated based on the application package.

[0028] These and other features and characteristics of the present disclosure, as well as the methods of operation and functions of the related elements of structures and the combination of parts and economies of manufacture, will become more apparent upon consideration of the following description and the appended claims with reference to the accompanying drawings, all of which form a part of this specification, wherein like reference numerals designate corresponding parts in the various figures. It is to be expressly understood, however, that the drawings are for the purpose of illustration and description only and are not intended as a definition of the limits of the disclosed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0029] Additional advantages and details are explained in greater detail below with reference to the non-limiting, exemplary embodiments that are illustrated in the accompanying schematic figures, in which:

[0030] FIG. 1 is a schematic diagram of a system for secure application distribution according to some non-limiting embodiments or aspects;

[0031] FIG. 2 is a flow chart for secure application distribution according to some non-limiting embodiments or aspects; and

[0032] FIG. 3 is a schematic diagram of example components of one or more devices according to some non-limiting embodiments or aspects.

DETAILED DESCRIPTION

[0033] For purposes of the description hereinafter, the terms “end,” “upper,” “lower,” “right,” “left,” “vertical,” “horizontal,” “top,” “bottom,” “lateral,” “longitudinal,” and derivatives thereof shall relate to the embodiments as they are oriented in the drawing figures. However, it is to be understood that the embodiments may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings and appendix, and described in the following specification, are simply exemplary embodiments or aspects of the disclosed subject matter. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting.

[0034] No aspect, component, element, structure, act, step, function, instruction, and/or the like used herein should be construed as critical or essential unless explicitly

described as such. Also, as used herein, the articles “a” and “an” are intended to include one or more items and may be used interchangeably with “one or more” and “at least one.” Furthermore, as used herein, the term “set” is intended to include one or more items (e.g., related items, unrelated items, a combination of related and unrelated items, and/or the like) and may be used interchangeably with “one or more” or “at least one.” Where only one item is intended, the term “one” or similar language is used. Also, as used herein, the terms “has,” “have,” “having,” or the like are intended to be open-ended terms. Further, the phrase “based on” is intended to mean “based at least partially on” unless explicitly stated otherwise.

[0035] As used herein, the term “account identifier” may include one or more primary account numbers (PANs), tokens, or other identifiers associated with a customer account. The term “token” may refer to an identifier that is used as a substitute or replacement identifier for an original account identifier, such as a PAN. Account identifiers may be alphanumeric or any combination of characters and/or symbols. Tokens may be associated with a PAN or other original account identifier in one or more data structures (e.g., one or more databases, and/or the like) such that they may be used to conduct a transaction without directly using the original account identifier. In some examples, an original account identifier, such as a PAN, may be associated with a plurality of tokens for different individuals or purposes.

[0036] An “application program interface” (API) refers to computer code or other data stored on a computer-readable medium that may be executed by a processor to facilitate the interaction between software components, such as a client-side front-end and/or server-side back-end for receiving data from the client. An “interface” refers to a generated display, such as one or more graphical user interfaces (GUIs) with which a user may interact, either directly or indirectly (e.g., through a keyboard, mouse, etc.).

[0037] As used herein, the term “communication” may refer to the reception, receipt, transmission, transfer, provision, and/or the like of data (e.g., information, signals, messages, instructions, commands, and/or the like). For one unit (e.g., a device, a system, a component of a device or system, combinations thereof, and/or the like) to be in communication with another unit means that the one unit is able to directly or indirectly receive information from and/or transmit information to the other unit. This may refer to a direct or indirect connection (e.g., a direct communication connection, an indirect communication connection, and/or the like) that is wired and/or wireless in nature. Additionally, two units may be in communication with each other

even though the information transmitted may be modified, processed, relayed, and/or routed between the first and second unit. For example, a first unit may be in communication with a second unit even though the first unit passively receives information and does not actively transmit information to the second unit. As another example, a first unit may be in communication with a second unit if at least one intermediary unit processes information received from the first unit and communicates the processed information to the second unit.

[0038] As used herein, the term “computing device” may refer to one or more electronic devices configured to process data. A computing device may, in some examples, include the necessary components to receive, process, and output data, such as a processor, a display, a memory, an input device, a network interface, and/or the like. A computing device may be a mobile device. As an example, a mobile device may include a cellular phone (e.g., a smartphone or standard cellular phone), a portable computer, a wearable device (e.g., watches, glasses, lenses, clothing, and/or the like), a personal digital assistant (PDA), and/or other like devices. A computing device may also be a desktop computer or other form of non-mobile computer.

[0039] As used herein, the terms “client” and “client device” may refer to one or more client-side devices or systems (e.g., remote from a transaction service provider) used to initiate or facilitate a transaction (e.g., a payment transaction). As an example, a “client device” may refer to one or more POS devices used by a merchant, one or more acquirer host computers used by an acquirer, one or more mobile devices used by a user, and/or the like. In some non-limiting embodiments or aspects, a client device may be an electronic device configured to communicate with one or more networks and initiate or facilitate transactions. For example, a client device may include one or more computers, portable computers, laptop computers, tablet computers, mobile devices, cellular phones, wearable devices (e.g., watches, glasses, lenses, clothing, and/or the like), PDAs, and/or the like. Moreover, a “client” may also refer to an entity (e.g., a merchant, an acquirer, and/or the like) that owns, utilizes, and/or operates a client device for initiating transactions (e.g., for initiating transactions with a transaction service provider).

[0040] As used herein, the term “payment device” may refer to a payment card (e.g., a credit or debit card), a gift card, a smartcard, smart media, a payroll card, a healthcare card, a wristband, a machine-readable medium containing account information, a keychain device or fob, an RFID transponder, a retailer discount or

loyalty card, a cellular phone, an electronic wallet mobile application, a personal digital assistant (PDA), a pager, a security card, a computing device, an access card, a wireless terminal, a transponder, and/or the like. In some non-limiting embodiments or aspects, the payment device may include volatile or non-volatile memory to store information (e.g., an account identifier, a name of the account holder, and/or the like).

[0041] As used herein, the term “server” may refer to or include one or more computing devices that are operated by or facilitate communication and processing for multiple parties in a network environment, such as the internet, although it will be appreciated that communication may be facilitated over one or more public or private network environments and that various other arrangements are possible. Further, multiple computing devices (e.g., servers, point-of-sale (POS) devices, mobile devices, etc.) directly or indirectly communicating in the network environment may constitute a “system.” Reference to “a server” or “a processor,” as used herein, may refer to a previously-recited server and/or processor that is recited as performing a previous step or function, a different server and/or processor, and/or a combination of servers and/or processors. For example, as used in the specification and the claims, a first server and/or a first processor that is recited as performing a first step or function may refer to the same or different server and/or a processor recited as performing a second step or function.

[0042] As used herein, the term “transaction service provider” may refer to an entity that receives transaction authorization requests from merchants or other entities and provides guarantees of payment, in some cases through an agreement between the transaction service provider and an issuer institution. For example, a transaction service provider may include a payment network such as Visa® or any other entity that processes transactions. The term “transaction processing system” may refer to one or more computer systems operated by or on behalf of a transaction service provider, such as a transaction processing server executing one or more software applications. A transaction processing server may include one or more processors and, in some non-limiting embodiments or aspects, may be operated by or on behalf of a transaction service provider.

[0043] Non-limiting embodiments or aspects of the disclosed subject matter are directed to systems, methods, and computer program products for secure application distribution that address the technical problems facing existing systems. For example, non-limiting embodiments may avoid requiring a persistent state (local or distributed)

to serve an application to end-user devices (e.g., such as web browsers on account holder mobile devices). Through the unique arrangements described herein, non-limiting embodiments permit global distribution of a complex static web application package (such as but not limited to an application formed from multiple strongly-coupled files like JS (Java Script), CSS (cascade style sheet), HTML (hypertext multimedia language), and/or the like). Moreover, the unique embodiments described herein permit service of an application from multiple datacenters without any particular datacenter being designated as primary or secondary. In non-limiting embodiments, only certain entities (such as approved merchants) can request the “major.minor” version of the application (e.g., where the request is formatted with major revision and minor revision version numbers). Non-limiting embodiments prevent service of older versions of applications requested by merchants, which might contain bugs or vulnerabilities, and provide such merchants with access to integrity tags for served files (e.g., such as those that they pull themselves from a Content Distribution Network (CDN)). These improvements provide for enhanced efficiency, reduced usage of bandwidth and number of communications, reduced usage of computing resources, and additional improvements and advantages as described herein.

[0044] In electronic payment processing networks, merchant systems (e.g., such as merchant web servers) may request a payment application and/or authentication application for conducting a payment transaction. In such examples, merchants may have the application package request hardcoded into webpages and/or scripts. Thus, merchants have to update this code based on updates to the application version, which may be for security, efficiency, and/or the like. Non-limiting embodiments describe herein provide a system in which merchants do not have to take affirmative actions to update an application package request.

[0045] Referring to FIG. 1, shown is a system 1000 for secure application distribution according to non-limiting embodiments or aspects. A receiving system 100 may include one or more computing devices in communication with a data storage device including application data 102. In non-limiting embodiments, the receiving system 100 may include, for example, a transaction processing system in an electronic payment processing network. In non-limiting embodiments, the receiving system 100 may include, for example, a payment gateway, a service provider system (e.g., such as a map service), one or more application distribution servers, and/or any other system arranged in a networked environment and capable of processing requests.

[0046] The application data 102 may be stored local or remote to the receiving system 100. The application data 102 may include one or more software applications, such as web applications, to be provided as a service to one or more requesting computing devices (e.g., a requesting system 104). The application data may also identify network locations of assets, such as application distribution servers 106, 110 and corresponding application data 108, 112. The system 1000 may include multiple application distribution servers 106, 110 arranged in different geographic locations. Although two application distribution servers (106, 110) are shown in FIG. 1 for purposes of illustration, it will be appreciated that in non-limiting embodiments, any number of servers and/or application distribution servers may be utilized and may be deployed in one or more locations, such that one or multiple servers may be deployed at one or multiple locations. Each application distribution server 106, 110 may be in communication with a respective data storage device including application data 108, 112. In some non-limiting embodiments, receiving system 100 may also serve as an application distribution server.

[0047] With continued reference to FIG. 1, a requesting system 104 may communicate an initial request for an application package to the receiving system 100 and/or another system in communication with the receiving system 100. In non-limiting examples, the requesting system 104 may be a merchant system (e.g., such as a merchant web server) that is requesting a payment application to be incorporated into a merchant webpage for engaging in transactions with consumers. However, it will be appreciated that non-limiting embodiments may be used in various other scenarios and by various other parties.

[0048] In non-limiting embodiments, the initial request for an application package may be communicated by the requesting system 104 through an API or another communication interface provided by the receiving system 100. The initial request may be in the form of a message that identifies the requesting party, identifies the requested application package, identifies a version or system of the requesting party, and/or includes any other data that may be used to determine a corresponding application package to provide the requesting system 104. For example, although the version number may be in the request in some non-limiting embodiments, one or more data parameters, features, and/or the like in the request may additionally or alternatively be used to identify a corresponding application package. Based on the initial request, the receiving system 100 or other computing device may determine

corresponding application data from the application data 102 stored in the data storage device. The application data may include data corresponding to an application package, such as an application version, an expiration data, an origin URL, a full or partial application package, and/or the like.

[0049] In non-limiting embodiments, the receiving system 100 or another computing device generates a cryptographic token based on the application data 102 that it previously determined. The cryptographic token may be generated by inputting one or more data elements, such as the application data itself, into a token generation algorithm that, for example, hashes the inputted data and/or encrypts the inputted data with at least one encryption key. Once a token is generated, the receiving system 100 or another computing device generates at least one link, including a URL or another type of pointer, pointing to at least one asset of an application (e.g., an application package or a portion thereof) stored as application data (e.g., as application data 102, 108, and/or 112) in different locations. In some examples the link may be a hyperlink (e.g., provided in hypertext) and/or embedded in an image or button. The URL may include the cryptographic token in its path (e.g., <https://pointer.service/#XXXXXXXX>, where #XXXXXXXX is the cryptographic token).

[0050] The receiving system 100 or another computing device may communicate the link to the requesting system 104 and/or any other computing device capable of accessing the assets through the link. For example, the requesting system 104 may automatically follow the link, may display the link, may store the link for later use, and/or the like. In response to a computing device following the link, the corresponding application package may be served to the computing device. For example, the receiving system 100 or another computing device may, in response to receiving a request to access the link (e.g., in response to resolving an HTTP request for a domain), decrypt the cryptographic token from the link, modify the content path of the link based on the decrypted token, and then process the modified content path to provide access to the assets.

[0051] For example, in an example scenario in which the link is <https://pointer.service/#XXXXXXXX> and the cryptographic token is #XXXXXXXX, the token may be decrypted into application data that is used to generate a new link by replacing the token with other data. In some examples, a look-up table may be used to identify the network location of the asset (e.g., application package). In other examples, the decrypted token itself may be the network location or part thereof (e.g.,

#XXXXXXXX may decrypt to a file path of “/v3/package.zip” which completes the URL as “https://pointer.service/v3/package.zip”). In some non-limiting embodiments, the application data that is encrypted as the token may include an application version and an origin URL, which may be located in a header field of an HTTP response packet sent from the receiving system 100 or another computing device to the requesting system 104.

[0052] Referring now to FIG. 2, shown is a flow chart for secure application distribution according to non-limiting embodiments or aspects. The steps shown in FIG. 2 are for example purposes only. It will be appreciated that non-limiting embodiments may involve additional steps, fewer steps, different steps, and/or a different order of steps. In some non-limiting embodiments or aspects, a step may be performed automatically in response to the completion of a previous step (e.g., may be performed without user intervention upon the completion of a previous step). At step 200, a request for an application package may be received from a client computing device of a requesting entity. For example, the initial request may include an API call, where the requesting party (e.g., such as a merchant) expresses an intent to use a specific semantic version of an application (e.g., “I want to use APP_X at version 2.3”), which may be expressed as { “application”: “APP_X”, “version”: “2.3” } or the like. The requesting party may specify the web application by “major.minor” version information.

[0053] The initial API call may be received by a system such as a transaction processing system, payment gateway system, application distribution server, and/or the like. For example, in non-limiting embodiments, a transaction processing system may expose the API for requests and may be in further communication with one or more application distribution servers, which may be part of or separate from the transaction processing system, to serve the application package. At step 202, application data is determined based on the request. For example, if a request asks for version 2.3 of “APP_X”, it may be determined, and the API response may indicate, a search of all released APP_X versions resulted in multiple versions available (e.g., [2.3.0, 2.3.1, 2.3.2, 2.3.3]). In an example in which a transaction processing system receives the request, the decision made by the transaction processing system controls which version of the request web application the merchant will launch. This allows the transaction processing system to serve a desired (e.g., best, most recent, most secure, globally available, and/or the like) version of the web application in response

to the merchant request. Thus, by ensuring that all application distribution servers have the version of the web application bundle available, requests of that asset can be serviced by any datacenter and/or application distribution server.

[0054] At step 204, a cryptographic token is generated based on the application data. For example, a cryptographic token may be generated by hashing, encrypting, or performing any other cryptographic process on or with the application data or some portion thereof. The use of a cryptographic token may obfuscate the internal semantic version resolution from the requesting party when communicated in a content path (e.g., URL) and to enable other advantages and improvements discussed herein. As an example, Authenticated Encryption (AE) with Associated Data (AD) encryption (AEAD encryption) may be used to encrypt the application data. AE is used to prevent any attacks involving injecting other data into ciphertext and provides assurance that the decrypted token can be ultimately trusted. AD is used to additionally ensure that token relates to the asset requested (e.g. “sdk.js” is AD in the above example). The keys used may be, in some examples, long-living keys with no specific rotation requirements. Keys may be propagated to a content delivery network (CDN) well in advance to allow serving of any asset request from any datacenter. In non-limiting embodiments, the cryptographic token allows time-bound instructions to be provided to a merchant or other requesting system to consistently serve only the application package that the receiving system determines should be available.

[0055] At step 206, a link is generated that includes the cryptographic token. The link may include a URL with a content path including the token (e.g., [https://files.visa.com/APP_X/\[token\]](https://files.visa.com/APP_X/[token])). At step 208, the link is communicated to the requesting system in a response. The response may further indicate an adherence to a contract or standard (e.g., the 2.3 SemVer contract) and data usable by a web browser of the requesting party to verify the authenticity of the response. The response may include the following structured data, as an example: { “appUri”: “https://files.visa.com/APP_X/[2.3.3]”, “files”: { “sdk.js”: “sha384-tsQFqpERLhBV...1”, “tools.js”: “sha384-aZajjp4Bqd...2” } }. This enables the requesting system to load the latest version of the “APP_X” web application by using the provided data to build HTML script tags, such as the following examples:

```
<script src="https://files.visa.com/APP_X/[2.3.3]/sdk.js" integrity=" sha384-tsQFqpERLhBV...1" crossorigin="anonymous"></script>
```

```
<script src="https://files.visa.com/APP_X/[2.3.3]/tools.js" integrity=" sha384-
aZajjp4Bqd...2" crossorigin="anonymous"></script>
```

[0056] The “[2.3.3]” in the URLs above are used for example purposes to convey the resolution of “major.minor” request into a most recent “major.minor.patch” response. In non-limiting embodiments and as described herein, the value of “[2.3.3]” in the URLs above is a cryptographic token to obfuscate the internal semantic version resolution from the requesting party and to enable other advantages and improvements discussed herein. For example, a URL may be:

`https://files.visa.com/microform/PlbbtM1Qqik8ZAoYrjNQoGDPtKzj++l/sdk.js`

where “PlbbtM1Qqik8ZAoYrjNQoGDPtKzj++l” is the cryptographic token. Replacement of plaintext (e.g., [major.minor.patch]) with an encrypted token provides several advantages. For example, in some non-limiting embodiments, only a transaction processing system may be enabled to create a cryptographic token, preventing other actors from making unsolicited asset requests to a CDN including multiple application distribution servers. Further, since only a transaction processing system (or other entity) has the key, any successful decryption constitutes a valid request to retrieve assets, and the decrypted token (e.g., data structure) can be trusted as instructions to the CDN to serve content in a specific way. Moreover, this arrangement allows for a transaction processing system to temporally restrict requests by adding expiry values, to augment responses by providing extra instructions (e.g., such as Cross-Origin Resource Sharing (CORS) or Content Security Policy (CSP) headers), and to implement business logic at the asset distribution point, without the need for complex server-side state. This further preserves erroneous use of assets by a requesting party (e.g. by mixing versions of assets, either by consumer error or by change introduced by a producer such as an ongoing new version rollout in a typical application roll-out) and restricts available versions of the web application by time, specific consumer, geography, and/or the like.

[0057] In non-limiting embodiments, the response (e.g., including the link with cryptographic and/or other data) may be communicated in one or more HTTP packets. The packets may include HTTP headers tailored for an individual execution to improve application security. For example, CSP or CORS headers (e.g. which may be required in some examples when pulling assets with subresource integrity (SRI) attribute) can limit malicious use or attacks against the application. In some examples, content tokens may be used to carry instructions on headers augmentation in a stateless

manner. For example, the released token may contain: {v=2.3.3, o=checkout.merchant.com} and the following headers may be served:

URL	Response Header
/TOKEN/sdk.js	Access-Control-Allow-Origin: checkout.merchant.com
/TOKEN/iframe.html	Content-Security-Policy: frame-ancestors https://checkout.merchant.com

[0058] In non-limiting embodiments, information about a domain of the requesting party (e.g., a merchant domain) may be used to supplement an initial API call. For example, if the intent is to “use microform in version 2.3 on my checkout.merchant.com domain”, the call may include { “application”: “microform”, “version”: “2.3”, “domain”: “checkout.merchant.com” }, as an example.

[0059] In response to a system accessing the link at step 210, a transaction processing system or other computing device may decrypt the cryptographic token from the URL at step 212. In an example where the token is a hash rather than an encrypted token, the hash may be referenced from a table where it is mapped to application data. A cryptographic token may be decrypted to, for example, “{v=2.3.3, e=1669040722}” where v is the version and e is token expiry value. If a token is expired the receiving system may notify the requesting system and/or determine the appropriate version. At step 214, the decrypted data (e.g., a string of data or portion thereof) may be used to modify the content path of the URL that included the cryptographic token, replacing the cryptographic token with the application data. At step 216, the application packaged may be served by one or more application distribution servers and datacenters in a CDN.

[0060] Typical CDNs may have rollout periods during which assets are replicated across all servers, and during this period data consistency is not guaranteed. This period of inconsistency can be challenging in existing systems. Non-limiting embodiments described herein solve this problem by deferring resolution of the “latest version” of a given bundle until replication is complete. This guarantees that each application package (e.g., bundle) will have all its assets available globally (e.g., across multiple application distribution servers), in a consistent manner, for the duration of a transaction or other task/event being performed. By storing information about a release schedule (e.g., information about what specific time point the 2.3.4 version should be served to replace version 2.3.3) in communication with the system that processes the initial API call (e.g., such as a transaction processing system),

application packages may be distributed well ahead before they are started to be consumed by a client computing device. For example, assuming for purposes of example that such a cut-off time is at 10:00 and maximum application lifecycle is 15 minutes, the following table of timing, requests, and responses illustrates how an application package may be globally distributed without interruption during new version rollouts, even though the rollouts may not be atomic (e.g., half the server cluster at once):

<i>Time</i>	<i>Client</i>	<i>Request</i>	<i>Response</i>	<i>Decrypted token</i>
09:59	A	API Init "2.3"	Token AAA	v=2.3.3,e=10:14
10:00	B	API Init "2.3"	Token BBB	v=2.3.4,e=10:15
10:03	A	GET AAA/x.html	2.3.3/x.html	
10:03	B	GET BBB/x.html	2.3.4/x.html	
10:15	A	GET AAA/y.html	401 error	
10:15	B	GET BBB/y.html	2.3.4/y.html	

[0061] In non-limiting embodiments, requesting systems (such as merchants) are prevented from running an application again if a newer version was released. Non-limiting embodiments provide a fully stateless solution from the perspective of the application distribution servers.

[0062] Referring now to FIG. 3, shown is a diagram of example components of a device 400 according to non-limiting embodiments or aspects. Device 400 may correspond to at least one of the computing devices shown and described in connection with FIG. 1. In non-limiting embodiments or aspects, such systems or devices may include at least one device 400 and/or at least one component of device 400. The number and arrangement of components shown in FIG. 3 are provided as an example. In non-limiting embodiments or aspects, device 400 may include additional components, fewer components, different components, or differently arranged components than those shown in FIG. 3. Additionally, or alternatively, a set of components (e.g., one or more components) of device 400 may perform one or more functions described as being performed by another set of components of device 400.

[0063] As shown in FIG. 3, device 400 may include bus 402, processor 404, memory 406, storage component 408, input component 410, output component 412, and communication interface 414. Bus 402 may include a component that permits communication among the components of device 400. In non-limiting embodiments

or aspects, processor 404 may be implemented in hardware, firmware, or a combination of hardware and software. For example, processor 404 may include a processor (e.g., a central processing unit (CPU), a graphics processing unit (GPU), an accelerated processing unit (APU), etc.), a microprocessor, a digital signal processor (DSP), and/or any processing component (e.g., a field-programmable gate array (FPGA), an application-specific integrated circuit (ASIC), etc.) that can be programmed to perform a function. Memory 406 may include random access memory (RAM), read only memory (ROM), and/or another type of dynamic or static storage device (e.g., flash memory, magnetic memory, optical memory, etc.) that stores information and/or instructions for use by processor 404.

[0064] With continued reference to FIG. 3, storage component 408 may store information and/or software related to the operation and use of device 400. For example, storage component 408 may include a hard disk (e.g., a magnetic disk, an optical disk, a magneto-optic disk, a solid state disk, etc.) and/or another type of computer-readable medium. Input component 410 may include a component that permits device 400 to receive information, such as via user input (e.g., a touch screen display, a keyboard, a keypad, a mouse, a button, a switch, a microphone, etc.). Additionally, or alternatively, input component 410 may include a sensor for sensing information (e.g., a global positioning system (GPS) component, an accelerometer, a gyroscope, an actuator, etc.). Output component 412 may include a component that provides output information from device 400 (e.g., a display, a speaker, one or more light-emitting diodes (LEDs), etc.). Communication interface 414 may include a transceiver-like component (e.g., a transceiver, a separate receiver and transmitter, etc.) that enables device 400 to communicate with other devices, such as via a wired connection, a wireless connection, or a combination of wired and wireless connections. Communication interface 414 may permit device 400 to receive information from another device and/or provide information to another device. For example, communication interface 414 may include an Ethernet interface, an optical interface, a coaxial interface, an infrared interface, a radio frequency (RF) interface, a universal serial bus (USB) interface, a Wi-Fi® interface, a cellular network interface, and/or the like.

[0065] Device 400 may perform one or more processes described herein. Device 400 may perform these processes based on processor 404 executing software instructions stored by a computer-readable medium, such as memory 406 and/or

storage component 408. A computer-readable medium may include any non-transitory memory device. A memory device includes memory space located inside of a single physical storage device or memory space spread across multiple physical storage devices. Software instructions may be read into memory 406 and/or storage component 408 from another computer-readable medium or from another device via communication interface 414. When executed, software instructions stored in memory 406 and/or storage component 408 may cause processor 404 to perform one or more processes described herein. Additionally, or alternatively, hardwired circuitry may be used in place of or in combination with software instructions to perform one or more processes described herein. Thus, embodiments described herein are not limited to any specific combination of hardware circuitry and software. The term “programmed or configured,” as used herein, refers to an arrangement of software, hardware circuitry, or any combination thereof on one or more devices.

[0066] Although embodiments have been described in detail for the purpose of illustration, it is to be understood that such detail is solely for that purpose and that the disclosure is not limited to the disclosed embodiments or aspects, but, on the contrary, is intended to cover modifications and equivalent arrangements that are within the spirit and scope of the appended claims. For example, it is to be understood that the present disclosure contemplates that, to the extent possible, one or more features of any embodiment or aspect can be combined with one or more features of any other embodiment or aspect.

WHAT IS CLAIMED IS:

1. A system comprising at least one processor programmed or configured to:
 - receive, from a requesting system, an initial request for an application package;
 - determine application data based on the initial request;
 - generate a cryptographic token based on the application data;
 - generate at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL;
 - communicate the at least one link to the requesting system; and
 - serve the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.
2. The system of claim 1, wherein the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof.
3. The system of claim 1, wherein the application data comprises an application version and an origin URL, wherein serving the application package comprises:
 - generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL.
4. The system of claim 1, wherein generating the cryptographic token comprises encrypting the application data with at least one encryption key.
5. The system of claim 4, wherein serving the application package comprises:
 - decrypting the cryptographic token in response to the requesting system accessing the at least one link;

modifying the content path based on the decrypted cryptographic token;
and
processing the modified content path.

6. The system of claim 1, wherein the at least one processor is further programmed or configured to:

communicate, to the requesting system, a data integrity value generated based on the application package.

7. A computer-implemented method comprising:
receiving, from a requesting system, an initial request for an application package;

determining application data based on the initial request;

generating a cryptographic token based on the application data;

generating at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL;

communicating the at least one link to the requesting system; and

serving the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

8. The computer-implemented method of claim 7, wherein the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof.

9. The computer-implemented method of claim 7, wherein the application data comprises an application version and an origin URL, wherein serving the application package comprises:

generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL.

10. The computer-implemented method of claim 7, wherein generating the cryptographic token comprises encrypting the application data with at least one encryption key.

11. The computer-implemented method of claim 10, wherein serving the application package comprises:

decrypting the cryptographic token in response to the requesting system accessing the at least one link;

modifying the content path based on the decrypted cryptographic token; and

processing the modified content path.

12. The computer-implemented method of claim 7, further comprising:

communicating, to the requesting system, a data integrity value generated based on the application package.

13. A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to:

receive, from a requesting system, an initial request for an application package;

determine application data based on the initial request;

generate a cryptographic token based on the application data;

generate at least one link comprising at least one asset URL, the at least one asset URL comprising the cryptographic token in a content path of the at least one asset URL;

communicate the at least one link to the requesting system; and

serve the application package from at least one server to the requesting system in response to the requesting system accessing the at least one link.

14. The computer program product of claim 13, wherein the application data comprises at least one of the following: an application version, an expiration data, an origin URL, or any combination thereof.

15. The computer program product of claim 13, wherein the application data comprises an application version and an origin URL, wherein serving the application package comprises:

generating at least one HTTP packet comprising a header field, the header field comprising the application version and the origin URL.

16. The computer program product of claim 13, wherein generating the cryptographic token comprises encrypting the application data with at least one encryption key.

17. The computer program product of claim 16, wherein serving the application package comprises:

decrypting the cryptographic token in response to the requesting system accessing the at least one link;

modifying the content path based on the decrypted cryptographic token;
and

processing the modified content path.

18. The computer program product of claim 13, wherein the program instructions further cause the at least one processor to:

communicate, to the requesting system, a data integrity value generated based on the application package.