

1 **SYSTEM AND METHOD FOR PRODUCT REGISTRATION**

2 3 **FIELD OF THE INVENTION:**

4 **[0001]** The present invention relates generally to the production of products having
5 sensitive data therein.

6 **DESCRIPTION OF THE PRIOR ART**

7 **[0002]** A device that participates in a cryptographically secure communication system,
8 will typically have some type of unique and immutable information that was injected into the
9 device at the time of manufacturing. This information could be a cryptographic key, a shared
10 secret or some other data that may be cryptographically bound to an inherently unique
11 attribute of the device. Such information may be generally referred to as a "key", and the
12 injection of information may be generally referred to as "keying" the device or "key
13 injection".

14 **[0003]** The purpose of injecting the keys is to ensure that the device is accepted as an
15 authentic participant of a secured communication system at some point in the future, after the
16 device has been distributed. However, the producer of the device will often wish to ensure
17 that devices are manufactured legitimately and thus wishes to protect the keys that are
18 injected into the devices. The producer will typically aim to protect the keys in order to
19 protect future revenue, since authentication of the keys may be used to provide conditional
20 access to the secure system and its content etc. The injected key is also important as it
21 enables a customer or user of the device to avoid tedious procedures required to register the
22 device.

23 **[0004]** The device may be granted such conditional access to the system based on
24 cryptographic authentication that the key is trusted. This trust is based on the fact that it is
25 exceptionally difficult to reproduce the trusted data outside of the manufacturing process.
26 Systems that provide conditional access include, e.g., satellite television and radio, those
27 systems that continuously broadcast information but wish to control access to their content
28 and thus revenue for providing such content. These systems rely on the manufacturing
29 process and the Original Equipment Manufacturer (OEM), in particular, key injection, to
30 provide a root of trust for the devices, and ultimately for the entire secure communication
31 system.

1 [0005] Keys that are injected into the devices are sometimes of a standard format and
2 purchased from a governing body, for example, High Definition Content Protection (HDCP)
3 keys, which are used to protect data as it is sent over a cable from your PC to your monitor
4 among other things. The governing body thus also has an interest in ensuring that the keys
5 distributed to the device's producer are protected and not lost. This creates a liability for the
6 producer, thus increasing the importance for protecting the injected keys. In some cases, the
7 producer can be fined for losing or copying keys and if they acquire a reputation for
8 negligence when handling keys, the governing body may restrict or severe the distribution of
9 the keys. Maintaining this relationship is often important to the producer, especially when
10 the keys are of a standard format needed for the device to be compatible with other devices
11 and/or infrastructure. In this case, without being able to use a particular key, the device will
12 not work as intended.

13 [0006] In a modern business climate comprising ever-increasing device complexity and
14 sophistication, it is common for individual parts to be manufactured and keyed by one
15 manufacturer for later assembly by another manufacturer. In such a situation there exists
16 certain security implications when the producer of the device or the owner of the
17 communication system is not the device manufacturer. It can therefore be paramount for a
18 device producer to ensure the integrity of the manufacturing systems that are responsible for
19 the integrity of the producer's device.

20 [0007] When considering the integrity of the manufacturing process, of particular
21 concern are issues related to the confidentiality of secret information that is used to
22 manufacture the device, as well as ensuring that the manufacturer correctly reports the
23 identities and the number of units manufactured to the producer. Ideally, the producer of the
24 device should try to obtain assurances that a manufacturer is not creating and distributing
25 "grey" or "black" market parts or devices. For example, a manufacturer that ships a certain
26 number of keyed products back to the producer, but still has leftover keys, may then
27 manufacture and sell devices with those extra keys. The producer has thus lost revenue since
28 the manufacturer is the one who profits from the sale. Other actions such as cloning or theft
29 of keys may also arise, which is difficult to detect and control when the keying process is
30 outsourced. In some cases, keys could be published on the Internet to enable users to gain
31 access to a conditional access system without paying for such a service.

1 [0008] Traditionally, a producer that is concerned about securing the information
2 injection stage at a manufacturing site has little choice but to implicitly trust that a
3 manufacturer is operating in a manner that gives due consideration to the producer's device
4 and system security. Protective mechanisms are generally naïve, in that keying information is
5 typically bulk encrypted and sent to the manufacturer, where, upon arrival, all of the keying
6 information is decrypted at once, and the manufacturer is then trusted not to compromise the
7 bulk of information.

8 [0009] One method to restrict access to keying information is to use an on-line client-
9 server mechanism. With such a mechanism in place, the client at the manufacturer's facility
10 would be connected to a network, and would make requests for keying information on a per-
11 device basis, to a remote key-providing server under the control of the producer.

12 [0010] There are a number of problems with implementing a manufacturing system that
13 relies on an off-site, remotely networked server, that provides keying information on such a
14 just-in-time basis. The foremost problem is that an off-site server can not guarantee a
15 minimal service level or response time to the manufacturing line if it uses a public shared
16 packet-switched network. To prevent problems in the manufacturing line, a certain level of
17 service in terms of latency and through-put is optimal. Given modern manufacturing
18 realities, where production lines exist in remote jurisdictions relative to the producer, such
19 guaranteed network availability can be prohibitively expensive.

20 [0011] A manufacturing facility will typically not begin a production run without all of
21 the necessary materials on hand, including data materials. Otherwise, the risk to production
22 line delays would be too high. Any keying system used by a manufacturer, should be able to
23 substantially guarantee service availability and provide a suitable response. This requires
24 local availability of all data resources and keying information before commencement of a
25 production run.

26 [0012] Given that all data resources must be locally available to a production line,
27 possibly existing on computer systems, and media that is not under direct control of the
28 producer; the producer must consider how to ensure the confidentiality of any secret keying
29 information.

1 **[0013]** Enough data should be locally available to the manufacturer, in order to
2 commence and complete a production run. In the event that the producer discovers
3 unauthorised and contractually objectionable behaviour by the manufacturer, the producer
4 should also consider how to prevent such a rogue manufacturer from producing grey or black
5 market product, after the termination of a contract.

6 **[0014]** Another problem related to cloning stems from overproduction, a specific type of
7 cloning operation, which is of particular concern to producers of silicon chips.
8 Overproduction can occur when the producer of an integrated circuit (IC) outsources
9 manufacturing of their IC designs to one or more third party manufacturing companies. The
10 purpose of outsourcing certain or all manufacturing steps is to lower production costs by
11 selecting a third party that can offer the best price for performing a particular stage in the
12 manufacturing process. For example, a fabless design house (e.g. a producer) may wish to
13 contract overseas manufacturing facilities to build chips that they have designed. Such
14 overseas manufacturing facilities are often chosen as they are able to produce electronic
15 components relatively inexpensively.

16 **[0015]** However, outsourcing generally increases the risk that a particular contractor may
17 overproduce product, that they have been contracted to build, in order to supply a grey
18 market. For example, if the contracted manufacturer acts in bad faith and over produces ICs
19 from the design provided by the producer, but does not inform the producer that such
20 overproduction occurs, the extra product is available to be sold in a grey market channel as
21 “counterfeit” or “cloned” ICs. This allows the third party manufacturers to realize extra
22 revenues and earnings at the expense of future product demand and revenues for their
23 customer, namely the producer/designer.

24 **[0016]** The above may occur because, in such scenarios, often the producer does not ever
25 handle the product aside from receiving engineering samples at the beginning of the
26 production phase. Accordingly, at each stage of the manufacturing process, subsequent to
27 design, there is an opportunity to steal parts and product. In some cases, employees of a good
28 faith contract manufacturer may be thieves. “Yield shrinkage” may occur, where an
29 employee steals product right off of the manufacturing line. This can be detrimental to not
30 only the producer and contract manufacturer, due to lost revenue, but also to the relationship
31 between the producer and the manufacturer for conducting future business.

1 [0017] It is therefore an object of the present invention, to obviate or mitigate the above-
2 described disadvantages.

3 SUMMARY OF THE INVENTION

4 [0018] The present invention provides for separating the addition of sensitive data to a
5 product between separate entities for inhibiting grey market product due to overproduction
6 and yield shrinkage.

7 [0019] In one aspect, a method is provided for for controlling a production process for a
8 product comprising including in a design for the product, a module having a mathematical
9 transform for intercepting and transforming data flow in the product, the mathematical
10 transform requiring a key to be operable and the product requiring successful operation of the
11 mathematical transform to be operable; arranging the production process into a plurality of
12 stages, each of the plurality of stages capable of being performed by respective ones of a
13 plurality of entities; dividing the key into a plurality of portions of sensitive data and
14 distributing each of the plurality of portions to respective ones of the plurality of entities; and
15 distributing a mask for producing the product according to the design to a first one of the
16 plurality of entities to enable the first one of said plurality of entities to add a first portion of
17 sensitive data to said key; wherein said product is operable upon adding all of said plurality
18 of portions of sensitive data.

19 [0020] In another aspect, a method is provided for controlling operation of a product
20 comprising intercepting data provided by a first portion of the product; inputting the data into
21 a mathematical transform; applying the mathematical transform to the data using a key to
22 generate an output, the mathematical transform requiring the key to be operable; and
23 providing the output to a second portion of the product; wherein the product requires
24 successful operation of the mathematical transform to be operable and the mathematical
25 transform comprises a plurality of portions of sensitive data each of which are added during
26 production of the product in a plurality of stages.

27 [0021] In yet another aspect, a registration module included in a product is provided for
28 controlling operation of the product comprising a mathematical transform for intercepting
29 and transforming data flow in the product; and a key stored in a memory, the mathematical
30 transform requiring the key to be operable, the product requiring successful operation of the

1 mathematical transform to be operable; wherein the key comprises a plurality of portions of
2 sensitive data each of which are added during production of the product in a plurality of
3 stages.

4 **[0022]** In yet another aspect, a system is provided for controlling a production process for
5 a product comprising a controller at a first entity, the first entity being responsible for the
6 production process; and a plurality of servers at a plurality of process stages, the plurality of
7 servers being configured for receiving sensitive data from the controller in a secure manner
8 and for sending log reports to the controller; the controller being configured for distributing a
9 mask for producing the product to a first one of the plurality of servers to enable the first one
10 of the plurality of servers to add a first portion of sensitive data to a key in the product at a
11 first process stage, the key being used to operate a mathematical transform in the product, the
12 product requiring successful operation of the mathematical transform to be operable; and
13 wherein at least one other of the plurality of servers is configured for adding at least one other
14 portion of sensitive data to the key in at least one other process stage.

15 BRIEF DESCRIPTION OF THE DRAWINGS

16 **[0023]** An embodiment of the invention will now be described by way of example only
17 with reference to the appended drawings wherein:

18 **[0024]** Figure 1 is a schematic block diagram of a remote device registration system;

19 **[0025]** Figure 2 is a schematic representation of the graphical user interface (GUI)
20 illustrated in Figure 1;

21 **[0026]** Figure 3 is a schematic representation of a distribution image;

22 **[0027]** Figure 4 is a flow chart illustrating a key injection and reporting procedure;

23 **[0028]** Figure 5 is a flow chart illustrating a provisioning procedure;

24 **[0029]** Figure 6 is a flow chart depicting a credit instruction procedure;

25 **[0030]** Figure 7 illustrates a mapping scheme for another embodiment supporting
26 multiple products;

1 **[0031]** Figure 8 illustrates an example of a filtered log report; and

2 **[0032]** Figure 9 is a block diagram illustrating another embodiment of a remote device
3 registration system.

4 **[0033]** Figure 10 is a schematic block diagram of an embodiment for key injection using
5 multiple stages in a manufacturing process.

6 **[0034]** Figure 11 is a schematic representation of a mask incorporating a registration
7 module for separating key injection stages using the embodiment of Figure 10.

8 **[0035]** Figure 12 is a schematic representation of a stage shown in the embodiment of
9 Figure 10.

10 **[0036]** Figure 13 is a flowchart showing steps taken in producing a product.

11 **[0037]** Figure 14 is a schematic block diagram of an example product produced from the
12 mask shown in Figure 11.

13 DETAILED DESCRIPTION OF THE INVENTION

14 **[0038]** Referring to Figure 1, a remote device registration or trusted key injection system
15 is generally denoted by numeral 10. A producer 12 of a device 22 utilizes the services of a
16 separate entity, in this case an outside manufacturer 14, for the injection of unique and
17 immutable information into the devices 22. The information may be a cryptographic key, a
18 shared secret, or some other data that may be cryptographically bound to an inherently unique
19 attribute of the device 22 and will hereinafter be referred to as a “key”. The step of injecting
20 the key into a device 22 will hereinafter be referred to as “keying” or “key injection”.

21 **[0039]** The producer 12 utilizes a controller 16, which is a computer system that is
22 remote to the manufacturer’s facility. The controller 16 includes a hardware security module
23 (HSM) 11. The HSM 11 is a protected device used by the controller 16 to perform
24 cryptographically secure operations, such as encryption, decryption and signing. The HSM
25 11 may be tamper resistant (e.g. physically difficult to access) or may be tamper reactive (e.g.
26 erases data if tampered with). The controller 16 is responsible for packaging and conveying
27 keys and other information to the manufacturer 14 as well as for monitoring the distribution

1 and usage of the keys by the manufacturer 14. The producer 12 typically obtains bulk
2 quantities of keys (not shown) from an outside source such as a governing body, e.g.
3 producer of HDCP keys. The keys are stored in a data storage device 15 until they are to be
4 distributed to a particular manufacturer 14. The controller 12 and its operations can be
5 monitored, modified and thus controlled by an operator using a graphical user interface (GUI)
6 13. The GUI 13 is typically a software application that is displayed and interacted with using
7 a personal computer (not shown).

8 [0040] The controller 16 is connected to a server 18 residing at the manufacturer 14
9 through a pipeline 23. The pipeline 23 includes two forward communication channels,
10 namely a control channel 26 and a distribution channel 25, and a backward channel 24. The
11 control channel 26 is used by the controller 16 to meter the number of keys that the
12 manufacturer 14 may use by sending credit instructions. The distribution channel 25 is used
13 by the controller 16 to distribute protected blocks of keys to the manufacturer 14. The back
14 channel 24 is used by the system 10 to make the controller 16 aware of key usage for
15 reporting and auditing purposes. The channels 24, 25 and 26 may be arbitrary communication
16 channels and are not required to be either reliable or secure. Reliability and security over the
17 channels 24, 25 and 26 are provided using a combination of technical mechanisms and
18 processes/procedures. For example, if a message sent over the forward channel 26 to the
19 module 18 does not decrypt because it is corrupt, a user may phone an operator of the system
20 controller module 16, and have them send the message again.

21 [0041] The manufacturer 14 utilizes one or more server 18, which is a computer system
22 that is local to the manufacturer's facility and whose activities are monitored and metered
23 through messages sent by the controller 16. The server 18 also reports back to the controller
24 16 over the back channel 24. The server 18 includes an HSM 28 that is similar to the HSM
25 11 utilized by the controller 16. The HSM 28 stores a protected credit pool 30 which dictates
26 how many keys the manufacturer 14 may use. Use of the keys is metered by the controller 16
27 by monitoring data reported by the server 18, and adding or subtracting from the credit pool
28 30 accordingly. The credit pool 30 is an abstract concept representing the number of keys
29 that may be decrypted by the HSM 28 before the server 18 must request and obtain more
30 keys from the controller 16. The controller 16 distributes keys to the server 18 over the

1 distribution channel 25, and the server 18 will store the keys in a local data storage device 17
2 as will be explained more fully below.

3 [0042] The manufacturer 14 utilizes one or more equipment 20 used to inject the
4 cryptographic keys into the devices 22. Typically keying occurs during a testing phase of the
5 manufacturing process, and thus the equipment 20 is often a testing machine on an assembly
6 line. The equipment 20 includes a key agent 21 which is typically a software program or
7 toolkit that is loaded into the equipment 20 used to administer key injection at the application
8 side. The key agent 21 communicates with the server 18 to request and obtain keys as they
9 are needed. Typically, the server 18 will provide enough keys to the key agent 21 so as to not
10 disrupt the timing of the production process. However, the server 18 will not provide an
11 unnecessary number of keys so as to restrict the usage of the keys until keying approval is
12 provided by the controller 16 as metered through the credit pool 30.

13 [0043] Typically, the key agent 21 will have threshold levels that indicate when a new
14 batch of keys are needed by that particular equipment 20, so as to not disrupt production.
15 Since the controller 16 is typically not in constant communication with the server 18, the
16 controller 16 may adjust its parameters to ensure that enough keying material is made
17 available to the equipment 20 through the server 18, while ensuring that not too much key
18 data is released by the server 18, before the controller 16 can obtain key usage reports from
19 the server 18 as will be explained in greater detail below.

20 [0044] The key agent 21 will preferably include an application program interface (API)
21 that runs on the equipment 20 to enable an operator of the equipment itself to request keys,
22 either manually or in an automated fashion. The key agent 21 is used to provide a level of
23 protection for data passing between the server 18 and the equipment, and may be thought of
24 as a simplified secure sockets layer (SSL) connection between the server 18 and equipment
25 20. It will be appreciated that resources permitting, the key agent 21 may also be
26 implemented using an SSL connection between itself and the server 18. The key agent 21 is
27 also responsible for generating report records as keys are used, that are sent back to the server
28 18 for reporting purposes.

29 [0045] The controller 16 is the command center for monitoring and metering key
30 injection by the manufacturer 14. In order to control keying from a remote location, the GUI

1 13 is used by an operator to monitor and configure each manufacturer 14, server 18, and
2 equipment 20 that is under the control of the controller 16. An example GUI 13 is shown in
3 Figure 2. The GUI 13 is divided into a server window 200, a controller window 204 and an
4 equipment window 202. The server window 200 includes a list of the manufacturers 14 and
5 thus the servers 18 that are controlled by the controller 16. The particular controller 16 is
6 indicated in the controller window 204. The operator can select a particular manufacturer
7 (e.g. manufacturer A as shown in Figure 2), and the equipment 20 that is associated with the
8 manufacturer is displayed in the equipment window 202.

9 [0046] In the example shown in Figure 2, the server at manufacturer A comprises a
10 window offering information regarding server 1, server 2 and server 3. Each server has
11 certain data associated with it. For instance, as shown in Figure 2, each server includes a
12 progress bar showing their available storage space, available credit and number of keys
13 available for each of keytype 1 and keytype 2. Each tester window also displays log
14 information, such as dates on which previous reports were processed, previously reported
15 credit, previous refill amount, and data regarding missing log records. The server windows
16 also provide the operator with options 214 and 216 for remotely configuring and disabling
17 the server 18 from the controller 16.

18 [0047] The controller 16 has the capability of remotely configuring the servers 18. This
19 allows the controller 16 to change key types, add or delete key types and control other
20 configuration options. This is preferably accomplished by sending configuration messages,
21 along the control channel 26, to the server HSM 28. The HSM 28 may evaluate the
22 configuration messages, whereby some configuration messages alter the behaviour of the
23 HSM 28, and other configuration messages are sent to the server 18. Configuration messages
24 sent to the server 18 via the HSM 28, using this method, can help to ensure that the server 18
25 attains configuration instructions that are trusted and known to originate from the controller
26 16.

27 [0048] The controller 16 may remotely configure the system 10 at the server level or the
28 equipment level through the key agent 21. The controller 16 can also force polls of the
29 servers 18 and can adjust the intervals for regular polling. Typically, the servers 18 are
30 polled at a fixed interval, and the controller 16 can use a forced poll to obtain information
31 between the intervals as needed. For example, with a one day interval, the controller 16 may

1 need data to report to an administrator intraday, and thus can force a poll of all servers to
2 obtain such data. The GUI 13 may also include a controller email option allowing the
3 controller 16 to automatically contact an administrator in extreme circumstances, such as
4 decryption or distribution failure at critical production runs.

5 **[0049]** Each key that is distributed to the server 18 and injected by equipment 20 into
6 device 22 triggers certain log records at certain events. The GUI 13 can be used to search,
7 sort, compile and analyze the log records and to view a custom or standard report 400 as
8 shown in Figure 8. In this example, there are three primary log records that are generated. A
9 key to server log 402 is generated when a key is distributed by the producer 16 to a server 18,
10 a key to agent log 404 is generated by the HSM 28 at the point where it releases a key to the
11 key agent 21, and a key injection log 406 will be generated by the key agent 21 upon
12 injection of the key. Each log record may include any number of identifying information,
13 including ID types, time stamps, manufacturer, equipment etc. In the example report shown
14 in Figure 8 the report 400 illustrates a key to server log 402, key to agent log 404 and key
15 injection log 406 for a key having a sequence ID = 001. These records may then be used to
16 track the life cycle of the key having such a sequence ID number. It will be appreciated that
17 the report 400 may include any number of records and may be filtered based on any suitable
18 field. For example, a report 400 showing all keys injected on May 3rd by tester 2 at
19 manufacturer A could be compiled by filtering accordingly, a date field and a location field.

20 **[0050]** Referring now to Figure 3, the producer 16 may package a bulk set of keys in a
21 secure data transmission using a distribution image 40 that is to be sent to the server 18,
22 preferably using encryption. The distribution image 40 enables the producer to include keys
23 for multiple products destined for multiple servers 18 in one transmission. Each server 18 is
24 then able to decrypt and obtain a certain number of keys, but only after authorization has
25 been received by the HSM 28, from the controller 16 via the control channel 26. The image
26 40 is a collection of data records, each record contains a type 58, ID 60, size 54 and data 56
27 field. Where data 56 will typically contain the key data of an arbitrary size identified by size
28 54. Type 58 and ID 60 fields are used by the HSM 28 to identify the key data, possibly being
29 used to filter certain keys, depending on the HSM's 28 configuration, as previously instructed
30 via the control channel 26. Keys may be encapsulated such that the implementation does not
31 care what a key really looks like to the target. This makes it flexible and extensible without

1 requiring a redesign for each new key type. The wrapper should contain a type, size and
2 unique ID, the body is abstract. The wrapper may also contain elements to support more
3 advanced features like logging or variable assignment into the abstracted image.

4 **[0051]** The image 40 is encrypted with an image key 42. The image key 42 is used by
5 the server 18 to decrypt the image 40 and obtain the keys. The image key 42 is itself
6 encrypted for each server 18 and stored as a server header 48. A collection 44 of server
7 headers 48 are stored in a main header 46. To decrypt the image 40 and obtain the keys, the
8 header 48 is chosen by the server 18 and is decrypted by the HSM 28 to obtain the image key
9 42. The image key 42 is then used to decrypt the image 40.

10 **[0052]** As noted earlier, the distribution images 40 may be used to support multiple
11 products. Referring also to Figure 7 a mapping of product types and data blocks is shown.
12 For example, the producer 16 has three products, namely gamma utilizing key 1 (having filter
13 tag 1), beta utilizing key 2 (having filter tag 2) and an accompanying configuration block
14 (also having filter tag 2), and alpha utilizing key 1, key 2 and the configuration block. The
15 image 40 may include bulk quantities of keytype 1 and keytype 2, and the gamma and beta
16 products may be less sophisticated than the alpha product. Producer 16 can package a single
17 image 40 with data for, e.g. fifty (50) of each block, whereby a certain tester (e.g. tester 1)
18 has permission to manufacture, and thus may obtain fifty (50) of filter tags 1 and 2 for
19 producing fifty of product alpha. Another tester (e.g. tester 2) may at the same time have
20 permission to manufacture and thus obtain fifty (50) of filter tag 1 from the image 40, to
21 produce fifty of product beta, and fifty (50) of filter tag 2 to produce product gamma. An
22 image 40 may contain all of the keying data, possibly including multiple type of keys, to
23 produce a single product of any product type. A tester identifies to the server 18 the type of
24 product, or product model, that it is being programmed. This model information is sent to the
25 HSM 28 with the encrypted image 40, so that when the HSM 28 decrypts the image 40, the
26 key data 50, can be filtered and only the key data needed to program the identified product
27 model is ever release by the HSM 28 to the tester. Therefore, the producer 12 can support
28 multiple products with a single image 40 whilst taking steps to ensure that the manufacturer
29 14 can only manufacture the products that they are supposed to be manufacturing.

30 **[0053]** Since the image 40 can support multiple products, the log records are used to track
31 the actual key injection performed at the tester, which will be explain more fully below. By

1 tracking the log records, a producer 16 can attempt to detect if, e.g., a manufacturer 14
2 returns 50 of product gamma instead of 50 of product alpha (which they have been paid to
3 produce) whereby they could also have sold 50 of product beta on a grey or black market.
4 Such a discrepancy may or may not be malicious but in any case can be reasonably identified.

5 **[0054]** A typical life cycle of a key from its distribution over distribution channel 25 until
6 the HSM 28 reports to the controller 16 over back channel 24, is shown in Figure 4. The
7 highlighted blocks in Figure 4 represent those steps performed by secure modules, namely the
8 HSM 11 and the HSM 28. The controller 16 first obtains a bulk quantity of standard keys
9 from an outside supplier. The controller 16 then passes the keys to the HSM 11, and the
10 HSM 11 encrypts blocks of keys, each block containing a measured quantity of a certain
11 keytype. It will be appreciated that the keys may also be bulk encrypted into blocks having
12 more than one key type. The controller 16 then stores the bulk encrypted keys in the storage
13 device 15 until it receives an order or other command indicating that a block of keys is to be
14 distributed.

15 **[0055]** When the producer 16 distributes a block of keys, it first obtains a bulk encrypted
16 block and passes this block to the HSM 11. The HSM 11 decrypts the block and re-encrypts
17 the block of keys for transmission with the image key 42. The image key 42 is then itself
18 encrypted for each server 18 to producer the individual headers 48. These headers 48 are
19 stored in the group 44 of the main header 46. At this point, the HSM 11 generates a key to
20 server log 402 for the keys that have been re-encrypted for distribution. The log 402 is stored
21 locally at the producer 12 for later analysis. The re-encrypted block of keys is then
22 distributed over the distribution channel 25 to the server 18.

23 **[0056]** The server 18 passes the encrypted block of keys that are included in the image 40
24 to the HSM 28, and the HSM 28 then decrypts the image 40. The HSM 28 first selects its
25 particular header 48 from the group 44 and decrypts the image key 42. The image key 42 is
26 then decrypted to obtain the keys from the image 40. The image 40 is then preferably
27 validated, e.g., using a secure hashing algorithm, MAC, or digital signature, and filtered. The
28 HSM 28 also then re-encrypts each key that is obtained from the image 40 for storage. The
29 server 18 then stores the re-encrypted keys locally for later use by the equipment 20. It will
30 be appreciated that authenticity of the images 40 is assumed based on the unique symmetric
31 distribution keys k_{s1} and k_{s2} shared between the controller 16 and server 18. The messages

1 shared therebetween, can be considered authentic once a successful integrity check is
2 performed, e.g. after a sha-2 digest compare.

3 **[0057]** When the controller 16 receives a request from the equipment 20 for a certain
4 number of keys (e.g. N keys), the HSM 28 is given N keys to decrypt. A key to agent log
5 record 404 is then generated for each of the N keys that is decrypted by the HSM 28 and the
6 keys are passed to the equipment 20 for injection. At this point, the keys are "in the clear"
7 and are thus ready for injection.

8 **[0058]** The equipment 20 injects each of the N keys and the key agent 21 generates a key
9 injection log record 406 for each key that is injected. The HSM 28 will continually obtain the
10 key to agent log records 404 and key injection log records 406 and preferably concatenates
11 these records into a master log report R that is sent back to the controller 16 over the back
12 channel 24.

13 **[0059]** The individual logs are preferably concatenated into a binary file, that identifies
14 the date that the file was produced. The reports R are preferably encrypted by the HSM 28
15 with encryption key k_1 and returned to an application running on the server 18 to be sent
16 over the back channel 24. The controller 16 may then decrypt the report R and validate the
17 individual logs (e.g. 402, 404, 406). Each log may be tagged with a monotonically
18 synchronous number. If all the record ID values, put together, are not a contiguous set, then
19 the operator of the controller 16 will know where to track the missing logs in the sequence.

20 **[0060]** As explained above, the controller 16 had previously stored a number of key to
21 server log records 402 for the N keys when they were distributed. Therefore, the controller
22 16 expects at some time in the future to receive the report R that completes the lifecycle for
23 each key to indicate that the keys that were originally distributed have been decrypted and
24 injected into the correct device, by the correct server 18. The controller 16 is thus able to
25 evaluate log reports as they are provided. The controller 16 can then determine if any action
26 should be taken, such as intervening in the manufacturing operation (e.g. stop distribution), or
27 providing more keys. The controller 16 may also require further information before
28 distributing a further block of keys. In this way, the controller 16 can meter the distribution
29 and only provide more keys if the manufacturer is operating in good faith and has
30 consistently provided accurate log reports.

1 **[0061]** The log records (e.g. those shown in Figure 8) enable a producer to spot
2 discontinuities in the sequence of ID numbers. For instance, if a number of keys have been
3 distributed but have not reported a key to agent or key to injection log, the manufacturer may
4 have lost that key. This could indicate grey or black market activity. In another scenario, the
5 report R may include a key to agent log 404 but not a key injection log 406 for a particular
6 key. This may indicate that the problem originated at the particular equipment requesting the
7 key rather than the manufacturer 14 itself. Therefore, the manufacturer 14 may also use the
8 log reports R for auditing purposes and to identify internal malicious activity so as to
9 maintain its relationship with the producer 12. The life cycle of each key requires a report
10 record at each critical stage where the key is operated on. Therefore, the producer 12 has the
11 necessary information to identify where problems have arisen and to direct efforts towards
12 correcting or eliminating such problems. Preferably, the log records include information
13 pertaining to not only a sequence number for the key, but also the key type. In this manner,
14 the producer 12 can also determine if alpha products were commissioned, yet gamma and
15 beta products may have been produced.

16 **[0062]** The log reports provide the information to both deter malicious or unethical acts
17 by the manufacturer 14 and provide the means to evaluate the integrity of the existing
18 manufacturers 14 and tools to provide evidence of any undesirable activity. The use of
19 tangible evidence in detecting undesirable activity allows the producer 12 to confront the
20 manufacturer 14 with something more than a suspicion, which, in a case where the illicit
21 activity is occurring at the tester level (e.g. by an employee and not the company itself), may
22 salvage an important relationship between the producer 12 and the manufacturer 14.

23 **[0063]** In addition to distribution, the controller 16 uses the control channel 26 to control
24 the credit pool 30 and thus meter the key injection stage. A credit instruction procedure is
25 shown in Figure 6. The HSM 28 must consume credit from the credit pool 30 when
26 decrypting a distribution image 40 and obtaining keys. Over time, the credit pool 30 will
27 diminish and need to be replenished with a credit instruction file sent by the controller 16.

28 **[0064]** The controller 16 only sends one control message C to the server 18 at a time over
29 control channel 26. One of the preferably required files contained in this message is a credit
30 instruction file. The file can be an encrypted set of data for a specific server 18 that is
31 decrypted by the HSM 28, to a credit instruction. The credit instruction contains, e.g., the

1 serial number of the HSM 28 and/or server 18, the server's token ID, a sequence number,
2 new credit amount, and configuration data, that has all been signed by the controller 16.

3 [0065] Upon receiving the control message C, the HSM 28 decrypts the credit instruction
4 data from the control message C, and validates the signature. The HSM 28 also validates the
5 serial number and token ID as its own, if applicable. A validation of the sequence number is
6 then performed. The sequence number should be greater than the sequence internally stored
7 in the HSM 28. Once validated, the HSM 28 will update its internal sequence number and set
8 the value of the credit pool 30 to the credit value in the credit instruction.

9 [0066] The HSM 28 will then process any configuration messages in the control message
10 C to update its internal configuration, in order to enable the controller 16 to push
11 configuration data to the server 18, such as updates for filtering rules, keying information,
12 credit rules etc., as explained above in relation to the GUI 13. Configuration data can be
13 intended for the HSM 28, an application running on the server 18 or even the key agent 21.
14 The HSM 28 looks for configuration messages of a defined type to process them.
15 Configuration messages can be marked as private or public, and access thereto would then be
16 controlled by the HSM 28.

17 [0067] A credit report Cr is the server's response to processing a credit instruction in a
18 control message C. The credit report Cr may contain the serial number and token ID of the
19 HSM 28, the current sequence value, the current value of the credit pool 30, number of refills
20 to date, and an error code that is set to zero if no errors occurred during credit instruction
21 processing.

22 [0068] The credit report Cr is preferably signed by the HSM 28 using its signing key k_2 .
23 The report Cr is then encrypted for the controller 16 using the controller's public encryption
24 key k_3 . The report Cr is then sent to the controller 16 and stored with the log reports R for the
25 above described auditing purposes.

26 [0069] Prior to distributing keys, the producer 12 and the manufacturer 14 may undergo a
27 provisioning procedure to initialize the HSMs and the server 18. The provisioning procedure
28 is shown in Figure 5. The HSM 28 produces and sends a provisioning request message P to
29 the controller 16. This message P preferably contains the serial number of the HSM 28 being
30 used by the server 18. The HSM 28 generates the two cryptographic key pairs k_1 , k_2 (e.g.

1 RSA key pairs or preferably using elliptic curve cryptography (ECC)), one (k_1) for receiving
2 encrypted messages and another (k_2) for signing outgoing messages. Preferably, the
3 manufacturer 14 is cryptographically bootstrapped in a physically controlled environment
4 during this exchange of key pairs k_1 and k_2 .

5 [0070] When the controller 16 receives the provisioning request from the server 18, it
6 passes the request to the HSM 11 who checks the integrity of the message and then assigns
7 the manufacturer 14 a "token ID". Two keys, preferably symmetric keys k_{s1} and k_{s2} (e.g.
8 Advanced Encryption Standard (AES) keys), are generated. These keys are to be used by the
9 controller 16 and server 18 to protect the distribution images 40 on the distribution channel
10 25 and the log reports R on the backward channel 24 as mentioned above.

11 [0071] The HSM 11 then generates a provisioning response message P' that, for example,
12 contains the assigned token ID, public keys of the HSM's encryption and signing key pairs k_3
13 and k_4 respectively, the distribution and backward channel symmetric keys k_{s1} and k_{s2} , some
14 initial configuration data, and a hash digest for integrity. Similar to the provisioning request
15 message P, it is assumed that the provisioning response message P' is handled within a
16 physically controlled environment (e.g. using HSM protection).

17 [0072] The provisioning response message P' may then be sent to the server 18, and the
18 server 18 may then perform initialization operations upon receiving its first provisioning
19 request. The structure of the provisioning response may contain a member that decrypts to a
20 separate structure that contains symmetric keys for the forward and backward channel
21 communications between the controller 16 and server 18. It shall be noted that these keys are
22 distinct for each HSM 28 (and thus each server 18), and are not shared amongst a group of
23 HSMs. Once the provisioning procedure is complete, a normal exchange of distribution
24 images 40 and control messages C may commence.

25 [0073] In another embodiment, shown in Figure 9, the system 10 may be retrofitted to
26 existing solutions that have been implemented by the manufacturer 14 for protecting the key
27 injection stage. In the embodiment shown in Figure 9, like elements are given like numerals
28 with the suffix "a". For example, a manufacturer 14, may have equipment 20a that already
29 includes a scrambler 74 for converting a string "BCA" to "ABC", where the device 22 is
30 wired to accept ABC as the injected key. In this manner, if the key "BCA" is stolen or

1 misplaced, it will not work for the device 22a because the scrambling has not occurred.
2 These attempts at protecting a key, although easy to implement, are typically naive and may
3 not provide a suitable level of protection. By accommodating for such protection, the system
4 10 may then be retrofitted to the equipment 20a without undoing an existing solution that has
5 already been implemented. Accordingly, additional cost to manufacturer 14 for
6 implementing system 10 can be avoided. The retrofit may be implemented until a complete
7 redesign is warranted, at which time the arrangement shown in Figure 1 may be used.

8 **[0074]** In order to accommodate existing solutions, the system 10 stores a set of signed
9 objects 72 at the server 18, which are a collection of executable files that are associated with
10 particular equipment 20a and perform the existing solution subsequent to the HSM 28a
11 releasing a key, and prior to key injection. In this way, the key is altered to accommodate the
12 existing solution without the equipment 20a being aware. As shown in Figure 9, the
13 controller 16a would first need access to the executable file (exe) 70 that is used by the
14 equipment 20a to provide the existing solution. The controller 16a would then pass the exe
15 70 to the HSM 11a. The HSM 11a would then sign the exe 70 and pass the signed exe 70 to
16 the HSM 28a, and the HSM 28a may then store the signed exe 70 as a signed object 72. In
17 operation, when the equipment 20a requests a new batch of keys, the server 18a will validate
18 the exe against the exe's signature, that is stored in the HSM 28a. Once the server 18a has
19 verified the exe 72, it will send the exe keys to be scrambled.

20 **[0075]** For example, equipment 20a requires a key BCA to feed to scrambler 76 in device
21 22a so that the key ABC is injected to product alpha. The HSM 28a determines that product
22 alpha has a signed object exe A, for modifying key ABC. The signed object exe A is
23 verified, and applied to key ABC resulting in scrambled key BCA. The scrambled key BCA
24 is then sent to equipment 20a, and the scrambler 76 modifies key BCA so that it injects key
25 ABC. The equipment 20a does not realize that the key BCA (that it received) was stored by
26 the server 18a in a protected form as ABC. It will be appreciated that the key stored by the
27 server 18a may also be in a form such as CAB, which is then modified to read BCA for
28 scrambling to then be converted to ABC for injection. Such a case may arise when key CAB
29 is the standard form and must be modified to suit an existing solution where CAB would not
30 be accepted as the key. Therefore, the signed objects 72 will contain any program required to

1 accommodate the existing solution implemented by equipment 20a, and the example
2 provided above is solely for illustrative purposes.

3 [0076] The signed objects 72 also inhibit malicious code from being loaded into the
4 server 18a for modifying the keys prior to injection, since the signed executables are typically
5 verified for the keys to be released to the machine prior to being applied to a key. The system
6 10 can thus provide an increased level of security whilst accommodating an existing solution.

7 [0077] Therefore, by utilizing a remote system controller 16 separate from the server 18,
8 the producer 12 is able to monitor the activities of the manufacturer 14, and meter credit
9 through the HSM 28. The producer 16 is thus able to govern the injection of keying
10 information on the devices 22, in order to ensure that the manufacturer 14 correctly reports
11 the identities and the number of units manufactured for the producer 12. This enables the
12 producer 12 to have assurances that a manufacturer 14 is not creating and distributing grey or
13 black market products or devices 22.

14 [0078] With the above procedures and system 10 in place, a producer 12 can monitor
15 production at a manufacturer 14. The producer 12, using the credit instructions in the control
16 messages C, can meter the production of devices 22 by adding or removing available credit
17 for use by the manufacturer 14.

18 [0079] It will be appreciated that the system 10 is not limited to one manufacturer 14 as
19 shown in Figure 1, nor is each manufacturer 14 limited to one set of equipment 20. The
20 system 10 is also not to be limited to the use of a single controller 16. The HSM 28 is most
21 preferably trusted hardware in order to protect key values and the integrity of the credit pool
22 30. Moreover, keying information contained in the distribution image 40 does not
23 necessarily have to be keying information, but can also be any data element that requires
24 confidentiality and authenticity. A requirement for keying data is typical of a system 10
25 which wishes to enforce granularity of device activation.

26 [0080] In an alternative arrangement, exemplified in Figures 10-14 and described in
27 greater detail below, overproduction may be inhibited by introducing a separation of duties
28 within the silicon or device production process. Typically, a producer 12 will contract out the
29 various stages of production (design, manufacture, testing etc.) to multiple contractors. In
30 general, separation of duties involves purposefully separating production stages, in particular

1 manufacturing stages, for silicon chips or other products, so that the end product has been
2 handled or "touched", by each subcontractor, in order for the end product to be fully
3 functional. Since grey markets are typically supplied by a single point of failure, or a single
4 bad-faith contractor in the production process chain, forcing a set of contractors to operate in
5 sequence implies that two or more subcontractors must collude against the producer 12, in
6 order to supply a grey market with non-crippled sub-components or devices. The end
7 product, and it's sub-components, should complete all stages to be fully functional. In
8 general, the risk of attack against the producer 12 is drastically reduced when multiple sub-
9 contractors are required to collude in order to steal.

10 **[0081]** During the production of silicon wafers, the wafer fabrication process requires
11 several steps, each of which can be sub-contracted to third party manufacturing contractors.
12 For example, a fabless semiconductor company that designs a chip (e.g. producer 12), may
13 distribute the design in an electronic data file representation to these third party contract
14 producers. These data files can include various abstractions of the behavioural, logical, and
15 physical design description contained in both standardized and proprietary description
16 languages. These files can also contain production details, that include but are not limited to,
17 design constraints and attributes, process parameters, and sequencing instructions for the
18 manufacturing processes involved. The purpose of these data files is to define for each
19 subcontractor the design processes and parameters required for each subcontractor to
20 complete their part of the production process within the guidelines set forth by the fabless
21 semiconductor company. The complete set of these data files, when considered in a
22 comprehensive manner, can be considered a complete description of the intended silicon
23 production process from a behavioural design abstraction to tested and packaged integrated
24 circuit components

25 **[0082]** An example of a semiconductor solid state product production process may
26 proceed as follows, making reference to Figure 10. First, a fabless semiconductor company,
27 also known as a fabless design house (e.g. producer 12) subcontracts the production of
28 finished IC chips to a semiconductor foundry company. In order to convey the intended
29 circuit description defined in an originating design description 90, the fabless semiconductor
30 company delivers either a register level design description (RTL code), produced at an RTL
31 design stage 100; or a gate level netlist design description, produced at stage 101; to the

1 semiconductor foundry company. In turn, the semiconductor foundry company then
2 subcontracts the physical design, also known as the transistor level design, to a third party
3 physical design firm, who receives the abstract RTL code or netlist from the foundry
4 company and returns a physical design description in the form of an electronic data file
5 during stage 103.

6 **[0083]** The semiconductor foundry company may then take the electronic design file that
7 represents the physical design, and deliver the electronic design file to a third party mask
8 designer. The mask designer receives the physical design files and builds, e.g., photomasks,
9 according to the parameters defined by the physical design file during stage 104. A mask 90'
10 is thus produced according to the originating design definition 90, after undergoing several
11 design stages in this example.

12 **[0084]** The photomasks are then delivered to the semiconductor foundry. The
13 semiconductor foundry uses the photomasks to fabricate the silicon wafers during stage 105
14 by repeated use of the photomasks to layer, pattern, dope, and heat the circuit elements onto
15 the wafers. This step in the process may be called wafer fabrication. After wafer fabrication,
16 each wafer contains multiple die in the shape of many squares or rectangles on the wafer.
17 Each die is a complete integrated circuit, which eventually becomes an individual chip. Each
18 wafer typically contains hundreds to thousands of die, depending on the size of each die.

19 **[0085]** Once the die are fabricated onto the wafer, the semiconductor foundry company
20 conducts wafer sort during stage 106, which is used to determine the yield of the wafer, or
21 how many die have avoided defects. The wafer sort process may also provide feedback on
22 the process technology and parameters as they relate to the yield of the wafer. Typically, at
23 this stage 106, the semiconductor foundry company then subcontracts out the packaging and
24 testing of the packaged parts. At the packaging house, the wafers are divided (e.g. "cut") into
25 individual die. The known "bad" die from the wafer sort process have been marked and are
26 discarded at this time. Each die is packaged in molded plastic during stage 107, and the pins
27 are bonded to the pads in the die. At this time the parts are sent to a final testing stage 108 at
28 yet another location. During the final testing stage 108, the "good" chips (i.e. those which
29 have passed) are sent back to the semiconductor foundry for final distribution during stage
30 109 back to the fabless semiconductor manufacturer or to the customer.

1 **[0086]** In summary, the exemplary semiconductor IC product production process for an
2 originating design description 90, can be broken down into the following stages: 1) RTL
3 design stage 100; 2) Gate level Netlist design stage 101; 3) Physical/Transistor design stage
4 103; 4) Mask manufacture stage 104 (e.g. photomask); 5) Wafer fabrication stage 105; 6)
5 Wafer sort stage 106; 7) Die packaging stage 107; and 8) Final testing stage 108. This is only
6 one representative example and it will be appreciated that any permutation of the eight steps
7 outlined above can be subcontracted, combined or be performed by the originator of the
8 design (e.g. producer 12). For example, there exists merchant semiconductor manufacturers
9 that do not typically subcontract any of the above stages. On the other hand, there also exists
10 fabless design houses as discussed above, that take an abstract idea or design or specification
11 for a chip and subcontract all stages. Also, certain subcontractors can provide all services for
12 every stage, while others are capable of only providing one service.

13 **[0087]** As such, it will be appreciated that any or all of the above stages may occur at a
14 single facility and that in other applications there may also be more or less stages, i.e. an
15 arbitrary N number of stages depending on the production process, application etc.

16 **[0088]** The product 22 includes some form of sensitive or immutable information that is
17 to be included in its design, and preferably cannot operate without such sensitive information.
18 The producer 12 subcontracts in this example (directly to the foundry and indirectly to their
19 subcontractors), eight third party manufacturing entities that perform specific stages in the
20 overall production of product 22.

21 **[0089]** In order to prevent an incomplete product or sub-components from being diverted
22 to a grey market 110 at one of the production entities 100-108, a "separation of duties" is
23 applied. The separation of duties is a division of production and data programming duties of
24 each production process stage, such that all duties are performed by the intended contractor in
25 the intended order, which are necessary to complete production of an un-crippled product 22.

26 **[0090]** In this example, a sensitive task such as the injection of cryptographic data is
27 divided into a plurality of stages, each of which is carried out by a distinct production entity
28 (e.g. design, manufacturing, testing etc.), during a distinct production process stage. In order
29 to separate the sensitive task(s) in this example, the producer 12 incorporates a registration

1 module definition 92 into the design description 90 such that a registration module 92' is
2 included in the mask 90' (see Figures 10 and 11), which is then used to fabricate wafers.

3 [0091] The module 92' is used such that a mathematical transformation intercepts critical
4 signals and/or data flows within the final product 22, such as a boot signal, and if the
5 mathematical transformation cannot operate, the product 22 is crippled. The mathematical
6 transformation is preferably a cryptographic transformation that makes use of invertible
7 cryptographic mathematical transforms based on the injected keying data, the transform
8 preferably being intractable as defined by the keying data. For example, in one embodiment,
9 use of Exclusive-OR (XOR) operations may be made for performance reasons, however use
10 of XOR operations is not a requirement.

11 [0092] In order for the mathematical transformation to operate, it may be registered
12 through incremental injections or additions of critical data, such as portions of cryptographic
13 keying data, at each stage of the production process. In this way, if a wafer produced at stage
14 105 is overproduced, and supplied to grey market stages 110 as shown in Figure 10, the
15 resultant product 112 is crippled, in this case because it has not received all of the
16 cryptographic data that is required to properly operate the product 22.

17 [0093] Preferably, as shown by way of example in Figure 10, the key injection system 10
18 described above and shown in Figures 1-9 may be used to distribute, meter, audit and solicit
19 reporting of the key injection stages at each manufacturing step. In this case, even if all
20 entities are in collusion to distribute grey market product, the producer 12 will be able to
21 detect this activity due to incomplete log reports, and if necessary inhibit the distribution of
22 further keying data. It will be appreciated that alternatively, system 10 may be used at any
23 number of stages and need not be used at each or any stage at all.

24 [0094] For example, the wafer production stage 105 may utilize the system 10 but not
25 any other stage. However, since preferably each manufacturing stage will include some form
26 of testing procedure, it is beneficial to incorporate system 10 into such testing. The producer
27 12 in this scenario would at least expect data during the second stage. It will also be
28 appreciated that the registration module 92' in the mask 90' (and definitions 92 and 90
29 thereof respectively) may be used without relying on the system 10 and may rely on each
30 manufacturing stage to implement a portion of the keying process. In any of these situations,

1 by splitting responsibilities, no one entity has the necessary information, on their own, to
2 successfully supply grey markets with product or sub-components.

3 [0095] A mask 90' produced from the design definition 90 is shown in greater detail in
4 Figure 11. As discussed above, the registration module definition 92 may be incorporated
5 into any design definition 90, which may implements a set of instructions or lines of code
6 etc., that will, in part, insert the registration module 92' within a path (preferably one that is
7 critical to the product's operation) between one portion of the customer code 120 and another
8 portion of the customer's code 122. Data that enters the module 92' along path 124 is applied
9 to a cryptographic transform 128 and is output to the other portion 122 along path 126. The
10 output present at path 126 will preferably only be usable if the cryptographic transform 128 is
11 successfully applied to the data input at path 124.

12 [0096] The cryptographic transform 128 preferably works with a memory 130, processor
13 132 and cryptographic key 134 in order to perform its operation. The memory 130, processor
14 132 and cryptographic key 134 are configured, preferably using the key injection systems 10
15 present at each production process stage. The memory 130 also includes another
16 cryptographic key 131, which, in general, comprises keying material that is accumulated at
17 each stage, preferably through injection using a key injection system 10 as shown in Figure
18 10. Preferably, the key 134 is used at injection time to ensure that the material being
19 accumulated in memory 130 to compose the key 131 is authentic. The key 134 may be a
20 public key, and may or may not be needed. For example, the module 92 may work without
21 the key 134, at the potential risk of some classes of attack that may or may not be relevant to
22 the particular producer 12.

23 [0097] In general, the sensitive data used by module 92' is split into portions, each
24 portion being added to key 131 at appropriate production process stages, e.g. those performed
25 once the mask 90' has been produced. For example, one technique would be to inject digital
26 signatures with message recovery at each stage in the production process. The key 134 may
27 be used to validate the digital signature, in doing so; the validated digital signature produces a
28 message that could be used in a key derivation scheme, with existing data in memory 130, to
29 derive a cryptographic key 131. Another example, would be to employ a key sharing
30 technique, where pieces of the cryptographic key 131 are added to memory 130 at various
31 production stages. When the final production process stage has been completed, the memory

1 130 contains enough data, so that the key sharing technique can be used to re-compose the
2 cryptographic key 131.

3 [0098] An example of a generic production process stage 102 is schematically shown in
4 Figure 12. As noted above, the producer 12 preferably utilizes system 10 for distributing
5 keying data and for monitoring reports generated when keying occurs. Key injection into a
6 silicon chip typically occurs at wafer test, or during a post packaging test. In this example,
7 the production stage 102 includes a server 18 and key agent 21 operating with testing
8 equipment 20. The stage 103 also includes a production process step 139, e.g. apparatus for
9 producing a silicon wafer. The production process step 139 uses the output of the previous
10 process step, which has been distributed thereto, to produce a partially completed Product₊
11 140, e.g. a silicon wafer. The subscript "+" in this example is used to represent the current
12 "portion" of sensitive data that has been applied to the product 22, where, preferably, an
13 additional portion of the sensitive data for this stage 102 is injected using the key agent 21 of
14 equipment 20. At this point, the Product₊ may not be fully operational, for the reason that the
15 transform 128 does not have all the necessary information to perform its operation. The
16 Product₊ is then available to be distributed to the next production stage.

17 [0099] Figure 13 provides a flow chart showing an example production process that
18 includes two distinct manufacturing stages (i.e. $N=2$) and where the producer 12 is
19 responsible for all design/production stages up to the production of the mask 90'. At step
20 500, the producer 12 determines the number of stages, and thus the number of portions of
21 keying data that will be injected, in this example, $N=2$. At step 502, the producer 12
22 preferably establishes a key injection system 10 that links each manufacturing stage to itself
23 over the channels 24, 25, and 26. As discussed above with reference to Figure 1, the
24 producer 12 may use a single controller 16 to communicate with multiple servers 18. In this
25 example, the producer 12 would distribute, monitor and receive log records from two servers
26 18.

27 [00100] At step 504, the producer 12 incorporates a registration module definition 92 into
28 the design description 90 and produces a mask 90' comprising such a registration module
29 92'. The mask 90' is then distributed to the first manufacturer for implementing stage 1 of
30 this exemplary 2-stage manufacturing process at step 506, and stage 1 is executed at step 508.
31 For example, the first manufacturer will produce a wafer, creating chips that conform to the

1 mask 90'. During wafer test, the manufacturer will then program some partial keying
2 material into memory 130. This portion of the sensitive data is inserted at step 510, and the
3 sever 18 would preferably report to the producer at step 512 using the mechanisms outlined
4 above. Alternatively, stage 1 may not handle the injection of any sensitive data, and this
5 operation may then be solely executed during stage 2.

6 **[00101]** Once the first portion of the keying data is programmed to the chip or product, the
7 product contains only partial keying information, not sufficient to operate properly. The
8 partially registered product is represented by Product₁, wherein the subscript 1 represents the
9 first portion as described above. The partially produced, partially programmed Product₁ is
10 then distributed to stage 2 at step 514, for execution at step 516. The next manufacturer, at
11 step 518, then injects a second portion of key data. For example, at step 518, the second
12 manufacturer may program additional keying information, or may derive cryptographic
13 keying information using partial key data stored in memory 130 during step 510 and new key
14 data from the system 10 used at step 518. This derivation step could be based on a hash, or
15 possibly a more sophisticated key sharing technique. Preferably, at step 520, the second
16 manufacturer reports back to the producer 12, indicating that the second key portion was
17 successfully injected. The producer 12 may now possess two log records indicating that the
18 key data has been successfully inserted, and can use this information to monitor its records.
19 It will be appreciated that stage 2 may be any one or a combination of production process
20 steps subsequent to the wafer fabrication performed in stage 1. For example, stage 2 may
21 include wafer sort, die packaging, and final testing, where, during final testing, the second
22 key portion is added.

23 **[00102]** Once the second portion of the keying data is inserted, the product 22, in this
24 example, is completely operable, and thus completely registered (e.g. tested and packaged
25 IC), and in Figure 13 is represented by Product₁₂, wherein the subscript 12 represents the
26 complete set of key data, namely data portion 1 and data portion 2. The Product₁₂ then
27 continues to a distribution channel at step 522 where it eventually arrives at the customer as a
28 working product at step 524.

29 **[00103]** As also illustrated in Figure 13, if, for example, the first manufacturer, or an
30 employee thereof, attempts to distribute grey market product at step 526, through an alternate
31 distribution channel at step 528, a crippled product would be provided to the customer at step

1 530, since the Product, only contains the first portion of the key data, and thus the transform
2 128 cannot perform its operation. Therefore, although the testing, packaging etc. may be
3 performed at grey market stage 2, the additional keying data is not provided, and thus the
4 product 530 is fully manufactured, but not completely registered, rendering it crippled. It
5 will be appreciated that the module 92 is preferably implemented such that anti-tampering
6 means are considered and implemented.

7 [00104] Referring now to Figure 14, a schematic example of a finished customer product
8 22a, incorporating a module 192 is shown, wherein module 192 is a logical manifestation of
9 the physical layout for module 92' shown in Figure 11. In Figure 14, like numerals may be
10 given the suffix "a" for clarity. The product 22a, using the implementation of module 92'
11 (e.g. 192) is able to apply the cryptographic transform 128a, being part of an enforcement
12 block 150, to the product's critical data path between code 120a and 122a. The path is
13 decoded through the transform 128a so that the customer's logic 122a can properly function.
14 In this example, a verification 132a, which is an implementation of processor 132, is
15 performed. The verification 132a uses a one-time programmable (OTP) memory 130a and an
16 identity portion 134a, which is an implementation of the key 134 of Figure 11. The key 134a
17 and memory 130a are injected with sensitive data using, e.g. the procedure outlined in Figure
18 13. It will be appreciated that the product 22a is only one implementation incorporating the
19 logic provided by module 92' (e.g. as module 192), and that the example shown in Figure 14
20 is for illustrative purposes only.

21 [00105] Although the above has been described with reference to certain specific
22 embodiments, various modifications thereof will be apparent to those skilled in the art.

Claims

1. A method of controlling a production process for a product (22,22a), said method comprising:
 - including in a design (90) for said product, a module (92') having a mathematical transform (128,128a) for intercepting and transforming data flow in said product, said mathematical transform requiring a key (131) to be operable and said product requiring successful operation of said mathematical transform to be operable;
 - arranging said production process into a plurality of stages of production, each of said plurality of stages of production capable of being performed by respective ones of a plurality of entities;
 - dividing said key into a plurality of portions and distributing each portion to a respective one of said plurality of entities; and
 - initiating a first stage of production according to said design, said first stage of production comprising a first one of said plurality of entities adding a first portion of said key to said product;wherein said product is operable upon all of said plurality of portions being added to said product.
2. The method according to claim 1, further comprising distributing an inoperable product (22,22a) comprising said first portion of said key (131) to at least one other entity to perform one or more subsequent stages whereby subsequent portions of said key are added to said product to complete said key thereby producing an operable product.
3. The method according to claim 1 or claim 2, further comprising establishing a communication channel (23) between a producer (12) and said plurality of entities for securely distributing said portions.
4. The method according to claim 3, further comprising each of said plurality of entities providing a log report (R) to said producer (12) upon adding a respective portion to said product.
5. The method according to claim 3 or claim 4, wherein said producer generates said design (90) and is responsible for dividing said key (131).

6. The method according to any one of claims 1 to 5, wherein said product is an integrated circuit (IC).
7. The method according to claim 6, wherein a definition of said design is used by said first one of said plurality of entities to produce a silicon wafer.
8. The method according to any one of claims 1 to 7, wherein said mathematical transform is a cryptographic transform and said key is a cryptographic key.
9. The method according to claim 8, wherein said cryptographic transform uses at least one exclusive-or (XOR) operation.
10. A system for controlling a production process for a product (22), said system comprising a controller (16) having a processor and memory, said memory storing computer executable instructions that when executed by the processor cause a plurality of servers (16) at a plurality of process stages to perform the method according to any of the preceding claims.
11. A method of controlling operation of a product, said method comprising:
 - intercepting data provided by a first portion (120a) of said product;
 - inputting said data into a mathematical transform (128, 128a);
 - applying said mathematical transform to said data using a key (134a) to generate an output, said mathematical transform requiring said key to be operable, said key having been made operable by a plurality of portions thereof being added by respective entities during respective stages of production; and
 - providing said output to a second portion (122a) of said product;wherein said product is operable if said mathematical transform is operable.
12. The method according to claim 11, wherein said data is a boot signal.
13. The method according to claim 11 or claim 12, wherein said mathematical transform is a cryptographic transform and said key is a cryptographic key.
14. The method according to claim 13 wherein said cryptographic transform uses at least one exclusive-or (XOR) operation.

15. The method according to any one of claims 11 to 14, wherein said first portion of said product comprises protected logic, and said second portion of said product comprises unprotected logic.

16. A registration module (92') for a product (22,22a), said registration module for controlling operation of said product, said registration module comprising a memory for storing a key (131) and computer executable instructions for performing the method according to any one of claims 11 to 15.

17. A product comprising the registration module according to claim 16.

18. The product according to claim 17 comprising an integrated circuit.