(12) **United States Patent**

Goodman et al.

(10) **Patent No.: US 12,395,356 B2**

(45) **Date of Patent: Aug. 19, 2025**

(54) **CERTIFICATE BASED SECURITY FOR DECLARATIVE OPERATIONS**

(71) Applicant: **DELL PRODUCTS L.P.**, Round Rock, TX (US)

(72) Inventors: **Bradley Keith Goodman**, Nashua, NH (US); **Joseph Brent Caisse**, Burlington, MA (US)

(73) Assignee: **Dell Products L.P.**, Round Rock, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 258 days.

(21) Appl. No.: **18/309,473**

(22) Filed: **Apr. 28, 2023**

(65) **Prior Publication Data**

US 2024/0235854 A1 Jul. 11, 2024

Related U.S. Application Data

(63) Continuation-in-part of application No. 18/151,769, filed on Jan. 9, 2023.

(51) **Int. Cl.**
  H04L 9/32 (2006.01)
  H04L 9/40 (2022.01)

(52) **U.S. Cl.**
 CPC *H04L 9/3268* (2013.01); *H04L 63/20* (2013.01)

(58) **Field of Classification Search**
  CPC H04L 9/3268; H04L 63/20
  See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2017/0134173 A1*	5/2017	Kern	....	H04L 9/3268
2018/0062860 A1*	3/2018	Gajjala	....	G06F 9/4401
2019/0327315 A1	10/2019	Grooters et al.		
2021/0044579 A1*	2/2021	Nelson-Gal	....	H04L 63/062
2021/0051029 A1	2/2021	Pham et al.		
2022/0131711 A1*	4/2022	Jatti	....	H04L 9/3268

* cited by examiner

Primary Examiner — Sangseok Park

(74) *Attorney, Agent, or Firm* — Larson Newman, LLP

(57) **ABSTRACT**

An endpoint node of a multiple node environment includes a storage and a processor. The storage stores a public key for an owner of the endpoint node. The processor receives a declarative request. The declarative request includes a signed certificate and a job to be performed in the endpoint node. In response to reception of the declarative request, the processor determines that the signed certificate is valid, and begins execution of the job associated with the declarative request. In response to a trigger event, the processor re-checks the signed certificate, and determines whether the signed certificate has been revoked. In response to the signed certificate being revoked, the processor performs an action indicated in a policy in the storage.
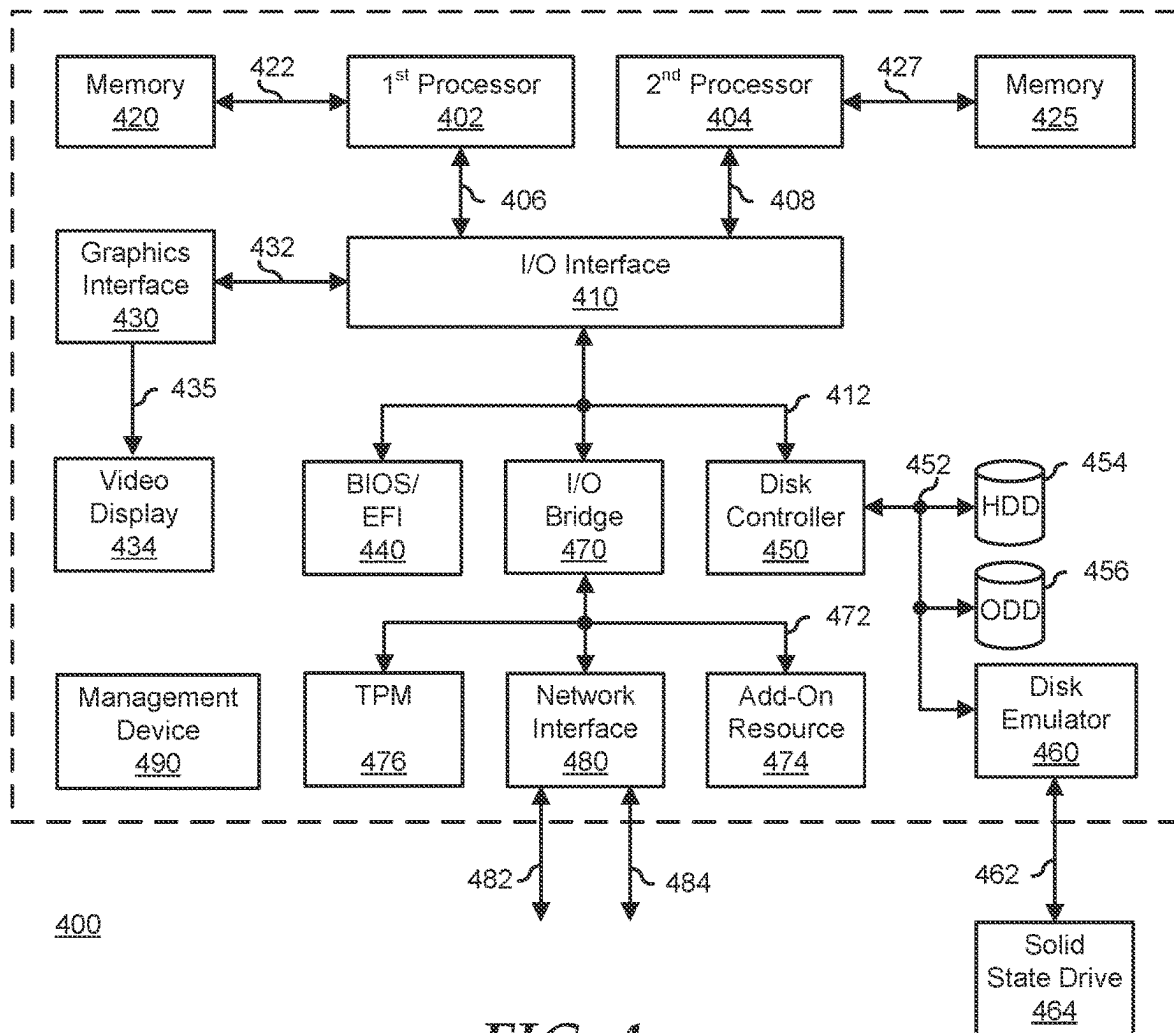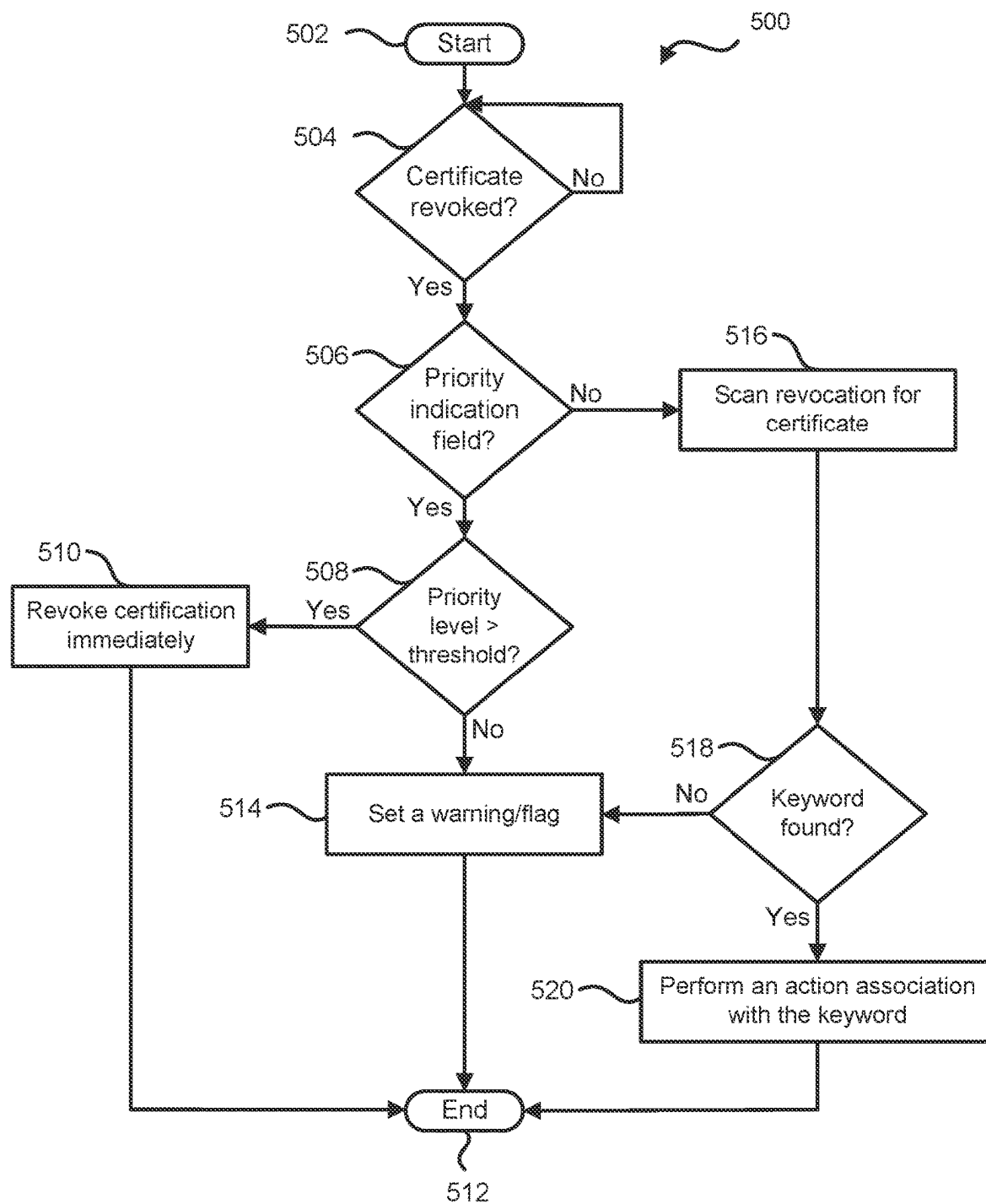
15 Claims, 5 Drawing Sheets
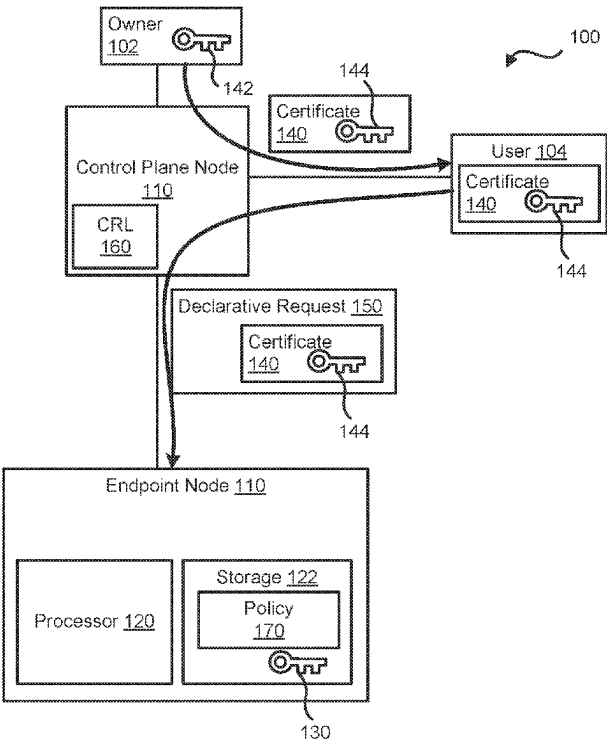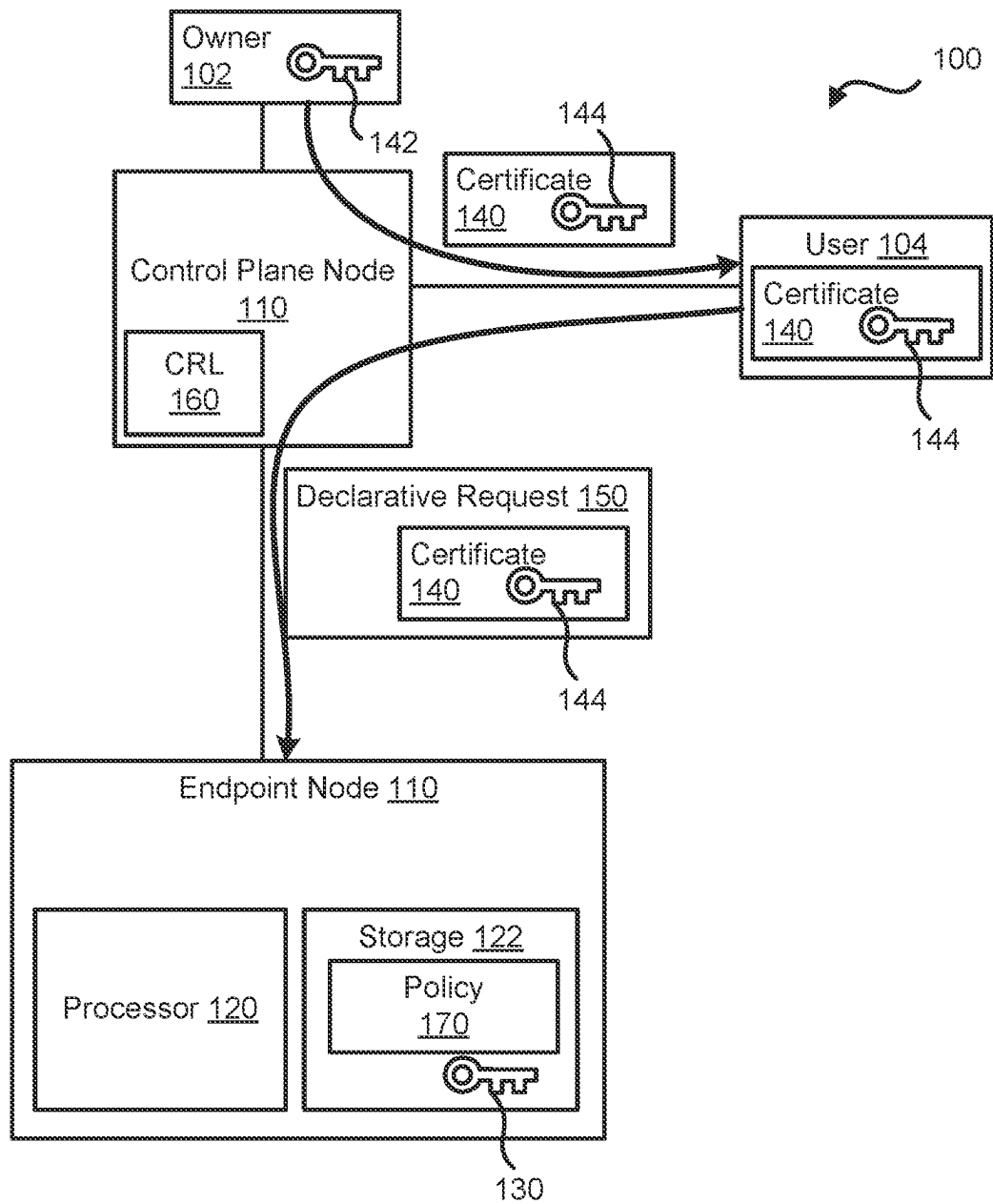
FIG. 1
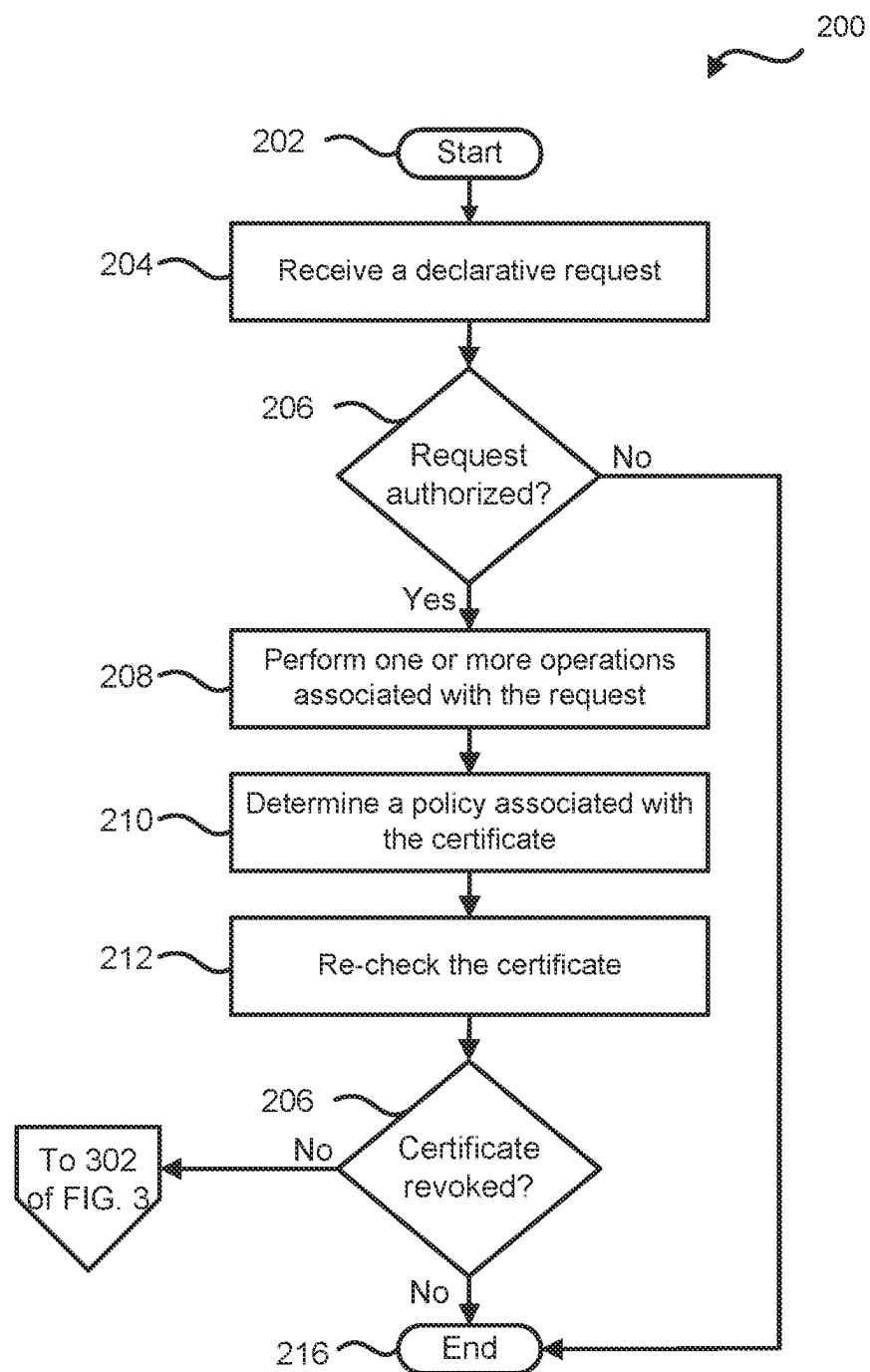
FIG. 2
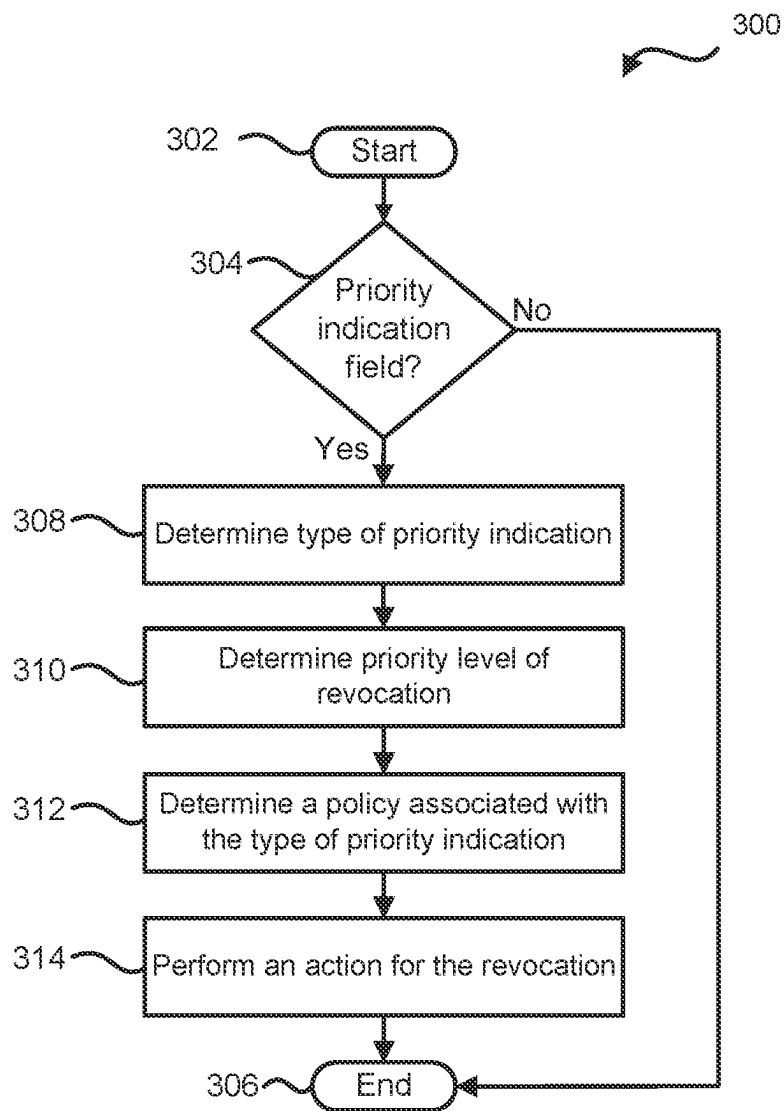
FIG. 3

FIG. 4

FIG. 5

CERTIFICATE BASED SECURITY FOR DECLARATIVE OPERATIONS

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a Continuation-in-Part of U.S. patent application Ser. No. 18/151,769 entitled "CERTIFICATE BASED SECURITY FOR DECLARATIVE OPERA-TIONS," filed on Jan. 9, 2023, the disclosure of which is hereby expressly incorporated by reference in its entirety.

Related subject matter is contained in co-pending U.S. patent application Ser. No. 18/309,522 entitled "CERTIFI-CATE BASED SECURITY FOR DECLARATIVE OPERATIONS," filed Apr. 28, 2023, the disclosure of which is hereby incorporated by reference.

Related subject matter is contained in co-pending U.S. patent application Ser. No. 18/309,551 entitled "CERTIFI-CATE BASED SECURITY FOR DECLARATIVE OPERATIONS," filed Apr. 28, 2023, the disclosure of which is hereby incorporated by reference.

FIELD OF THE DISCLOSURE

The present disclosure generally relates to information handling systems, and more particularly relates to certificate based security for declarative operations.

BACKGROUND

As the value and use of information continues to increase, individuals and businesses seek additional ways to process and store information. One option is an information handling system. An information handling system generally processes, compiles, stores, or communicates information or data for business, personal, or other purposes. Technology and information handling needs and requirements can vary between different applications. Thus information handling systems can also vary regarding what information is handled, how the information is handled, how much infor-mation is processed, stored, or communicated, and how quickly and efficiently the information can be processed, stored, or communicated. The variations in information handling systems allow information handling systems to be general or configured for a specific user or specific use such as financial transaction processing, airline reservations, enterprise data storage, or global communications. In addi-tion, information handling systems can include a variety of hardware and software resources that can be configured to process, store, and communicate information and can include one or more computer systems, graphics interface systems, data storage systems, networking systems, and mobile communication systems. Information handling sys-tems can also implement various virtualized architectures. Data and voice communications among information han-dling systems may be via networks that are wired, wireless, or some combination.

SUMMARY

An endpoint node of a multiple node environment may store a public key for an owner of the endpoint node. A processor may receive a declarative request. The declarative request may include a signed certificate and a job to be performed in the endpoint node. In response to reception of the declarative request, the processor may determine that the signed certificate is valid, and begin execution of the job

associated with the declarative request. In response to a trigger event, the processor may re-check the signed certifi-cate, and determine whether the signed certificate has been revoked. In response to the signed certificate being revoked, the processor may perform an action indicated in a policy in the storage.

BRIEF DESCRIPTION OF THE DRAWINGS

It will be appreciated that for simplicity and clarity of illustration, elements illustrated in the Figures are not nec-essarily drawn to scale. For example, the dimensions of some elements may be exaggerated relative to other ele-ments. Embodiments incorporating teachings of the present disclosure are shown and described with respect to the drawings herein, in which:

FIG. 1 is a block diagram of a multiple node environment according to at least one embodiment of the present disclo-sure;

FIG. 2 is a flow diagram of a method for handling revocations of certificates associated with declarative requests according to at least one embodiment of the present disclosure;

FIG. 3 is a flow diagram of a method for determining a revocation priority level for a certificate of a declarative request according to at least one embodiment of the present disclosure;

FIG. 4 is a block diagram of a general information handling system according to an embodiment of the present disclosure; and

FIG. 5 is a flow diagram of a method for handling revocations of certificates associated with declarative requests according to at least one embodiment of the present disclosure.

The use of the same reference symbols in different draw-ings indicates similar or identical items.

DETAILED DESCRIPTION OF THE DRAWINGS

The following description in combination with the Fig-ures is provided to assist in understanding the teachings disclosed herein. The description is focused on specific implementations and embodiments of the teachings, and is provided to assist in describing the teachings. This focus should not be interpreted as a limitation on the scope or applicability of the teachings.

FIG. 1 illustrates a multiple node environment 100 according to at least one embodiment of the present disclo-sure. Multiple node environment 100 includes an owner node 102 and a user node 104 that may access endpoint nodes 108 through a control plane 110. Endpoint node 108 may include a processor 120 and a storage 122. Control plane 110 may include any suitable type of control-plane, such as a global control plane node, regional control plane node, and local control plane node. In an example, global CP nodes, regional CP nodes, local CP nodes, and endpoint node 108 may be any suitable information handling system, such as substantially similar to information handling system 400 of FIG. 4, wherein each node may include a storage and a processor as described below with respect to FIG. 4. Multiple node environment 100 may include any suitable number of additional components or information handling systems without varying from the scope of this disclosure.

In an example, endpoint node 108 may store an owner public key 130 in storage 122. Owner public key 130 may be utilized by endpoint node 108 to authenticate any received messages or requests. In certain examples, owner

public key **130** may be hard coded within endpoint node **108** or any other suitable node outside of the control plane of multiple node environment **100**. Owner public key **130** may be associated with owner node **102** of endpoint node **108**, such as a company, a user, or the like. In an example, security of owner public key **130** may be increased based on the owner public key being stored within a trusted platform module (TPM) of endpoint node **108**.

In certain examples, owner **102** may perform one or more suitable operations to grant one or more rights or operations to user **104**. For example, owner **102** may provide a signed certificate **140** to user **104** via control plane node **112**. In an example, certificate **140** may be signed by an owner private key **142** and may authorize user **104** to request that one or more services or operations be performed in endpoint **108**. Certificate **140** may include a key signature **144** that was generated from owner private key **142**. In certain examples, user **104** may provide a request **150** including, but not limited to, an imperative request and a declarative request. Request **150** may include certificate **140** and key signature **144**, which in turn may enable endpoint node **108** to authorize or validate the certificate received from user **104**.

In an example, an imperative request or command may involve a particular action to be performed. For example, an imperative request may be for a memory in an endpoint, such as endpoint **108**, to be locked. In response to the imperative request, the endpoint may lock the designated memory. Subsequently, another request may be received to unlock the memory at which point the memory may be unlocked and the imperative request may no longer have any effect.

In an example, a declarative request or command may involve an action to be performed for an extended amount of time. For example, a declarative request may indicate that a memory should be locked. In response to the declarative request, the endpoint may lock the designated memory. Subsequently, another request may be received to unlock the memory at which point the memory may be unlocked. In an example, the declarative request may be different than an imperative request based on the declarative request still being implemented after the subsequent request was performed. For example, if the declarative request is for the memory to be locked and a subsequent request unlocks the memory, the declarative request may cause the memory to be locked again after the subsequent request is no longer being performed. In certain examples, a component, such as a memory, in an endpoint node may have a default state, a declarative request may cause the endpoint to place the component in a declarative state, and an invalidate request may end the declarative request or state so that the component is placed back in the default state.

In an example, declarative requests may live or be active over a long period of time. However, the certificates which predicate the security of these declarative requests may change during the period of time. In this situation, a command associated with a declarative request may be valid at the time of issuance but may later become invalid. In previous information handling systems or endpoint nodes, decisions or actions associated with the invalidation of an outstanding request for a declarative statement whose underlying security credentials have changed may not be straightforward. For example, the invalidation actions may require greater logic to balance security and avoid unintentionally desired behavior.

In certain examples, security of endpoint node, such as endpoint **108**, may be based on execution of commands in an imperative request. In an example, security models may

fundamentally be built on fundamentally deciding if a user is allowed or authorized to request that an operation be performed. This may be an imperative model which is assessed when the operation is requested or performed. However, declarative workloads may be long-lived, with persistent rules, such that these workloads are not imperative commands. In this situation, a simple imperative model does not work.

In an example, a declarative security model may be used for more a user causing an endpoint, such as endpoint **108**, to perform an action. If after reception of a declarative or an imperative request, endpoint **108** may be unplugged then later re-plugged in. Based on endpoint **108** being plugged in, the endpoint may perform different operations based on whether the request is declarative or imperative. For example, if endpoint **108** had received an imperative request, the endpoint would wait for a new command or request before an operation is performed. However, if endpoint **108** had received a declarative request, that declarative request may be durable. Thus, endpoint **108** would continuously perform operations to maintain the desired state indicated in the declarative request.

In certain examples, multiple node system **100** may implement a certificate-based security. In an example, certificates may be a way for one trusted entity, such as owner **102**, to convey trust to another user, such as user **104**. For example, endpoint **108** may include owner public key **130** for owner **102** stored within storage **122** and as such the endpoint may be able to authenticate and trust commands or requests from the owner. In an example, owner **102** may sign certificate **140** using owner private key **142**, and the certificate may grant or authorize user **104** request operations to be performed in endpoint **108**. The trust of owner **102** may be referred to as 'root' trust, which is known in the art and will not be described further except as needed for discussion of this disclosure.

In an example, certificates may be generally long-lived and may be distributed over different components of multiple node system **100**. In certain examples, long lived certificates may be difficult to control, which may cause security problems with multiple node system **100**. These security problems may be solved or corrected by any suitable means including, but not limited to, specifying expiration rules for the certificates, and allowing explicit revocation of the certificates.

In certain examples, one or more suitable declarative managers including, but not limited to, systemd, and Linux 'init' and 'cron' utilities, may ensure that a service or operation of a declarative request is started each time the computer starts, to run it periodically, or the like. In an example, a basic declarative request may be like an imperative one, such that the request may be evaluated, and action is taken when the request is received. However, if certificate-based permissions were used to attest the validity and authorize an operation to be performed, the certificate management practices may change this validity over time. For example, the first-time endpoint **108** installs a service and is told it to execute the service, the certificate must have been valid. But upon a subsequent reboot, if the certificate has expired, then the service would not restart upon reboot. An information handling system, such as endpoint node **108**, may be improved by controlling how certificates are utilized for declarative requests as will be described herein.

In an example, a certificate solution might include verifying a certificate for a declarative request only when the request is initially made. The declarative request may be for a service to be run every time an information handling

system, such as endpoint node **108**, is booted. Based on this solution, if the certificate later becomes invalid, the service may restart at a later boot of the information handling system because the certificate would not be verified again. In an example, the certificate solution of only verifying certificate **140** when declarative request **150** is received at endpoint node **108** might not always be appropriate. For example, there may be many reasons why certificate **140** may not work or not be valid later.

If the service initiated by declarative request **150** is an important service in endpoint node **108**, this service should not suddenly stop working just because certificate **140**, or an upstream dependency, has expired. In this situation, the service initiated by declarative request **150** maybe should continue to operate even if user **104** forgets to renew long-forgotten certificates, such as certificate **140**. For example, essential services should not stop running fail to restart for simple expirations of certificates, such as certificate **140**. In an example, owner **102** may explicitly revoke certificate **140**. In this example, owner **104** may want to stop or end the service or services initiated by declarative request **150**. In certain examples, certificate **140** may be revoked for any suitable number of reasons including, but not limited to, user **104** retiring and the user being fired. In the case of user **104** retiring, all certificates including certificate **140** may be retired, but owner **102** may want any long-standing services started by declarative request **150** to continue to operation. In the situation where user **104** is fired, all certificates including certificate **140** may be retired, and owner **102** may want all services started by declarative request **150** to end or be terminated.

In an example, processor **120** of endpoint node **108** may perform one or more suitable operations to determine whether declarative request **150** is authorized. For example, processor **120** may utilize public key **130** to verify key signature **144** in certificate **140** of declarative request **150**. In response to certificate **140** being verified, processor **120** may allow the operation or service identified in declarative request **150** to run or be executed. After the operation has begun, processor **120** may continually monitor the operation or service to ensure that the specified conditions are maintained.

In certain examples, processor **120** may implement a declarative request solution to continually determine whether a service in declarative request **150** may be executed. For example, processor **120** may recheck certificate **140** at any suitable point including, but not limited to, before restarting the operation of declarative request **150**, when any state change of the operation is made, when known actions to impact the certificate occur, and periodically. In an example, if endpoint node **108** is rebooted, processor **120** may recheck certificate **140** of declarative request **150** before restarting an operation or service in the declarative request. In this example, certificate **150** may be revoked by owner **102** after the initial authorization of declarative request **150**. Thus, upon endpoint node **108** rebooted, processor **120** may recheck certificate **150** and determine that the certificate is revoked and as a result prevent the restart of the service in declarative request **150**.

In an example, during the initial authorization of declarative request **150**, processor **120** may determine that certificate **140** was only valid for a period of time. In certain examples, the period of time may be any suitable length of time including, but not limited to, a number of days, weeks, months, and year. Based on the detection of an expiration time for certificate **140**, processor **120** may set a revocation timer that is substantially equal to the period of the certifi-

cate. In response to the end of the timer, processor **120** may invalidate declarative request **150** and stop the execution of the service. In an example, if processor **120** receives an update indication or updated certificate **140** with a new expiration period, the processor may cancel, reset, or restart the timer.

In certain examples, certificates, such as certificate **140**, may be revoked via Certificate Revocation Lists (CRLs) **160**, an Online Certificate Status Protocol (OSCP), or the like. In an example, CRLs **160** may be published and stored in any suitable location, such as control plane node **110**, of multiple node system **100**. Processor **120** may receive CRLs **160** via any suitable manner, such as control plane node **110** pushing the CRLs to endpoint node **108**, processor pulling the CRLs, the processor reading data from the CRLs, or the like. In an example, processor **120** may recheck certificate **140** based on a new CRL **160**. Endpoint node **108** may include one or more policies **170** to control how and when processor **120** rechecks certificate **140**.

In an example, multiple node system **100** may include security policy **170**, which may be provided to endpoint **108** and stored in storage **122** for use by processor **120**. In certain examples, security policy **170** may specify a granular basis of how processor **120** should handle rechecking the authentication of certificate **140** in declarative request **150**. Security policy **170** may be provided to endpoint node **108** via any suitable manner, such as being sent by control plane node **110**. In an example, policy **170** may be specified as part of declarative request **150** of the work being done. Policy **170** may be indicated in any suitable manner including, but not limited to, a single field or value to specify what to do, and a set of individual flags defining what to do in various instances.

In certain examples, a flags or security policies may be any suitable policies including, but not limited to, a policy to validate certificate **140** only during initial ingestion of declarative request **150**, a policy that only priority revocations may invalidate a declarative task, a policy that an expiration of the certificate will invalidate the task, and a policy that any invalidation, revocation or expiration will invalidate the task. In an example, if certificate **140** is validated during the ingestion only policy, the job or request may continue forever even if security of the certificate becomes invalidated.

In an example, multiple node system **100** may allow assessment policies to be specified or controlled by separate entities, via a different permissions, privileges, or enforcement schema. The assessment policies may be system or implementation dependent, and may include, but are not limited to, a hard-coded policy, a single centrally controller global policy, an administrator only permission, a permissions-based model, a per-system basis, and a per job basis. In an example, a single centrally controlled global policy may be a policy that is applied to all systems or endpoint nodes, such as endpoint node **108** under control of central plane **110**. In certain examples, an administrator only permission, such as "root" user or the like, may mean only such an administratively privileged user could specify Security-Policy. A permissions-based model may enable only a user with specific permissions may set Security Policies.

In certain examples, certificate **140** for user **104** may be invalidated for any reason. Based on certificate **140** being invalidated, processor **120** may determine whether to stop the job or service associated with declarative request **150**. In an example, processor **120** may perform or execute any suitable mechanism, such as auditing, eventing, or the like,

to determine whether the job associated with declarative request **150** should be stop based on certificate **140** being invalidated.

When processor **120** is executing in an auditing mode, the processor may poll or interrogate control plane node **110** for changes to certificates, such as certificate **140**, declarative requests, such as declarative request **150**, or the like. In this example, processor **120** may detect declarative request **140** and may check or audit this request to determine whether it is still valid. In an example, determining whether declarative request **140** may include determining whether certificate **140** associated with the declarative request has expired or has been revoked. As described above, certificate **140** may be checked by pulling or reading receive CRL **160** to determine whether the certificate is in the CRL. If certificate **140** is in CRL **140**, the certificate is expired or revoked. Otherwise, certificate **140** is still active and the job or service for declarative request **150** may continue. During the auditing mode, a user of endpoint node **108** may cause processor **120** to actively perform one or more operations to determine whether certificate **140** for declarative request **150** is valid.

While processor **120** is in the eventing mode, events or changes in certificate may be generated by any component with multiple node system **100** and reported to the processor of endpoint node **108**. In certain examples, the reporting of events may be provided to endpoint node **108** via owner **102**, control plane **110**, or the like. In an example, control plane node **110** or processor **120** may periodically check CRLs **160** and determine whether certificate **140** has been invalidated, has expired, or the like. In certain examples, processor **120** may evaluate or retrieve data from CRL **160** at a particular interval. However, if endpoint node **108** is unplugged from multiple node system **100**, processor **120** may not be able to access CRL **160**. In this situation, processor **130** may no longer be able to determine whether certificate **140** has been revoked. Processor **120** may have a maximum check interval with respect to CRL **160**. If processor **120** cannot access CRL **160** before the expiration of the maximum check interval, the processor may generate an event based on the expiration of the maximum check interval.

In the eventing mode, if certificate **140** is in CRL **160**, an invalidation event or an expiration event may be generated for the certificate and declarative request **150**. Based on the invalidation or expiration event, processor **120** may provide a notification to any user, such as a user of endpoint node **108**, user **104**, owner **102**, or the like, of multiple node system **100**. In certain examples, the events may be generated, and the audits may be performed in any suitable manner know to one skilled in the art.

As discussed above, revocation of certificate **140** may happen for many reasons, and the handling of the revocation may be different in different scenarios but may be the same in of different scenarios. In certain examples, owner **102** may utilize priority revocation to revoke certificate **140**. In an example, priority revocation refers to a process where the revocation may include additional information over information identifying certificate **140** as the certificate to revoke. This additional information may be whether a high priority flag is set, a numerical level for the revocation, or the like. In these examples, the priority revocation may allow the implementation to put different data with the revocation, which may be used by subsequent rules to make more informed decisions with respect to the revocation of certificate **140**.

In an example, when a priority flag is set in a revocation, the revocation of certificate **140** may be identified as a

priority revocation. In this example, security policy **170** may specify that only priority revocations, such as those revocations with the priority flag set, should be immediately invalidated. This security policy **170** may enable owner **102** to cancel all jobs for declarative request **150** when user **104** loses permissions based on severe actions but leaving jobs for declarative request when the user loses permission based on less severe actions, such as user **104** retiring from a company.

In certain examples, the priority revocation may include different revocation levels, such as three levels, four levels, five levels, six levels, or the like, from a lowest priority to a highest priority. In an example, the revocation levels may be utilized to identify different reason or level for certificate **140** to be revoked. For example, if owner **102** is a company and user **104** is an employee in the company the revocation levels may refer to reason why the user is no longer at the company. In an example, a first or lowest priority level may identify that user **104** has transferred to a different area of the company, a second priority level may identify that the user has retired, a third priority level may identify that the user quit, a fourth priority level may identify that the user was fired, and a fifth priority level may identify that the user was a corporate spy. Based on the different priority levels, an administer may make rules in policy **170** as to how processor **120** may handle the different priority or security risks associated with priority revocation levels for certificate **140**. For example, policy **170** may indicate that no action is performed when the priority level is the lowest priority. Policy **170** may also indicate that the jobs associated with declarative request **150** be immediately stopped if the priority level of the revocation is the highest priority. In certain examples, policy **170** may further indicate that each of the intermediate priority levels have a different one of a group of actions including, but not limited to, create an event for the revocation and provide a notification to a user of endpoint **108**. Based on the priority level and policy **170**, processor **120** may perform one of the indicated actions.

As described above, security within endpoint node **108** may be improved by controlling how processor **120** operates based on revocations of certificate **140** associated with declarative request **150**. In an example, security within endpoint node **108** may be improved via policy-based rules by which work, jobs, or services for declarative request **150** may be invalidated under different circumstances. Another improvement may be that owner **102** or other authorities above user **104** may establish and enforce policies **170**. In certain examples, a user associated with endpoint **108** may cause processor **120** to audit any lesser policy issues or violations rather than proactively invalidating certificate **140** under these issues.

Endpoint node **108** may be further improved by including a priority revocation levels to differentiate revocations which may be deemed more important or critical than other revocations. In certain examples, endpoint node **108** may also be improved via a flexible or policy-driven/specified re-assessment interval. In an example, processor **120** may implement a mechanism to automatically cancel or re-assess policies upon expirations of certificates, such as certificate **140**. Processor **120** may further implement auditing of CRL **160** to trigger security revocations to be checked, and utilizing a certificate recheck interval to determine certificate validity.

FIG. **2** illustrates a flow of a method **200** for handling revocations of certificates associated with declarative requests according to at least one embodiment of the present disclosure, starting at block **202**. It will be readily appreci-

ated that not every method step set forth in this flow diagram is always necessary, and that certain steps of the methods may be combined, performed simultaneously, in a different order, or perhaps omitted, without varying from the scope of the disclosure.

At block **204**, a declarative request is received by a processor of an endpoint node in a multiple node system. In an example, the declarative request may be received from a user node via a control plane of the multiple node system. The declarative request may include a job, workload, service, or the like to be continually executed within the endpoint node. The declarative request may also include a certificate authorizing the user to request that the job, workload, or service be performed in the endpoint node.

At block **206**, a determination is made whether the declarative request is authorized. In an example, a processor of the endpoint may perform the authorization determination via any suitable operation or operations. For example, the processor may determine whether a signature in the certificate of the declarative request is valid. In certain examples, the signature may be valid based on processor utilizing a public owner key to verify that the signature was created from a corresponding private owner key. In an example, the public owner key may be stored in a memory of the endpoint node.

In response to the declarative request being authorized, one or more operations associated with the declarative request are performed in the endpoint node at block **208**. At block **210**, a policy associated with the certificate is determined. At block **212**, the certificate for the declarative request is re-checked based on the policy. In an example, the certificate may be re-checked based on a trigger event. In certain examples, the policy may indicate that the trigger event for the certificate being re-checked may be any suitable event, such as at periodic intervals, response to a generated event, or the like.

At block **214**, a determination is made whether the certificate has been revoked. In an example, a processor of the endpoint node may access a certificate revocation list (CRL) and determine whether the certificate is located within the CRL. If the certificate is in the CRL, the processor may determine that the certificate has been revoked. If the certificate has been revoked, the flow continues at block **302** of FIG. **3**. If the certificate has not been revoked the flow ends at block **216**.

FIG. **3** illustrates a flow of a method **300** for determining a revocation priority level for a certificate of a declarative request according to at least one embodiment of the present disclosure, starting at block **302**. It will be readily appreciated that not every method step set forth in this flow diagram is always necessary, and that certain steps of the methods may be combined, performed simultaneously, in a different order, or perhaps omitted, without varying from the scope of the disclosure.

At block **304**, a determination is made whether a revocation priority indication field is included within the revocation of the certificate for the declarative request. If a revocation priority indication field is not set the flow ends at block **306**. If a revocation priority indication field is located within the revocation, a type of revocation indication is determined at block **308**. In an example, the type revocation priority indication may be a priority flag, a priority level indicator, or the like.

At block **310**, a priority level is determined for the revocation. In an example, if the priority indication is a priority flag, the priority level may be high if the flag is in a first state, such as a high or low state, and the priority level

may be low if the flag is in a second state, such as a low or high state. If the priority indication is a priority level, the priority level may be one of multiple priority levels. In an example, the priority levels may increase from a low priority, to a next higher priority, to any suitable number of additional next higher priorities, and to a highest priority.

At block **312**, a policy associated with the type of revocation priority indication. At block **314**, an action for the revocation of the certificate is performed and the flow ends at block **306**. In an example, the action for the revocation may be based on the policy and the priority level in the priority indication field. For example, if the security policy specifies that only priority revocations, such as those revocations with the priority flag set, should be immediately invalidated, all jobs are canceled that are associated with a declarative request having this revoke certificate. If the revocation type is different priority levels, the policy may indicate that no action is performed when the priority level is the lowest priority. The policy may also indicate that the jobs associated with declarative request be immediately stopped if the priority level of the revocation is the highest priority. In certain examples, the policy may further indicate that each of the intermediate priority levels have a different one of a group of actions including, but not limited to, create an event for the revocation and provide a notification to a user of endpoint. Based on the priority level and the policy, a processor may perform one of the indicated actions.

FIG. **4** shows a generalized embodiment of an information handling system **400** according to an embodiment of the present disclosure. For purpose of this disclosure an information handling system can include any instrumentality or aggregate of instrumentalities operable to compute, classify, process, transmit, receive, retrieve, originate, switch, store, display, manifest, detect, record, reproduce, handle, or utilize any form of information, intelligence, or data for business, scientific, control, entertainment, or other purposes. For example, information handling system **400** can be a personal computer, a laptop computer, a smart phone, a tablet device or other consumer electronic device, a network server, a network storage device, a switch router or other network communication device, or any other suitable device and may vary in size, shape, performance, functionality, and price. Further, information handling system **400** can include processing resources for executing machine-executable code, such as a central processing unit (CPU), a programmable logic array (PLA), an embedded device such as a System-on-a-Chip (SoC), or other control logic hardware. Information handling system **400** can also include one or more computer-readable medium for storing machine-executable code, such as software or data. Additional components of information handling system **400** can include one or more storage devices that can store machine-executable code, one or more communications ports for communicating with external devices, and various input and output (I/O) devices, such as a keyboard, a mouse, and a video display. Information handling system **400** can also include one or more buses operable to transmit information between the various hardware components.

Information handling system **400** can include devices or modules that embody one or more of the devices or modules described below and operates to perform one or more of the methods described below. Information handling system **400** includes a processors **402** and **404**, an input/output (I/O) interface **410**, memories **420** and **425**, a graphics interface **430**, a basic input and output system/universal extensible firmware interface (BIOS/UEFI) module **440**, a disk controller **450**, a hard disk drive (HDD) **454**, an optical disk

drive (ODD) 456, a disk emulator 460 connected to an external solid state drive (SSD) 462, an I/O bridge 470, one or more add-on resources 474, a trusted platform module (TPM) 476, a network interface 480, a management device 490, and a power supply 495. Processors 402 and 404, I/O interface 410, memory 420, graphics interface 430, BIOS/UEFI module 440, disk controller 450, HDD 454, ODD 456, disk emulator 460, SSD 462, I/O bridge 470, add-on resources 474, TPM 476, and network interface 480 operate together to provide a host environment of information handling system 400 that operates to provide the data processing functionality of the information handling system. The host environment operates to execute machine-executable code, including platform BIOS/UEFI code, device firmware, operating system code, applications, programs, and the like, to perform the data processing tasks associated with information handling system 400.

In the host environment, processor 402 is connected to I/O interface 410 via processor interface 406, and processor 404 is connected to the I/O interface via processor interface 408. Memory 420 is connected to processor 402 via a memory interface 422. Memory 425 is connected to processor 404 via a memory interface 427. Graphics interface 430 is connected to I/O interface 410 via a graphics interface 432 and provides a video display output 436 to a video display 434. In a particular embodiment, information handling system 400 includes separate memories that are dedicated to each of processors 402 and 404 via separate memory interfaces. An example of memories 420 and 430 include random access memory (RAM) such as static RAM (SRAM), dynamic RAM (DRAM), non-volatile RAM (NV-RAM), or the like, read only memory (ROM), another type of memory, or a combination thereof.

BIOS/UEFI module 440, disk controller 450, and I/O bridge 470 are connected to I/O interface 410 via an I/O channel 412. An example of I/O channel 412 includes a Peripheral Component Interconnect (PCI) interface, a PCI-Extended (PCI-X) interface, a high-speed PCI-Express (PCIe) interface, another industry standard or proprietary communication interface, or a combination thereof. I/O interface 410 can also include one or more other I/O interfaces, including an Industry Standard Architecture (ISA) interface, a Small Computer Serial Interface (SCSI) interface, an Inter-Integrated Circuit (I2C) interface, a System Packet Interface (SPI), a Universal Serial Bus (USB), another interface, or a combination thereof. BIOS/UEFI module 440 includes BIOS/UEFI code operable to detect resources within information handling system 400, to provide drivers for the resources, initialize the resources, and access the resources. BIOS/UEFI module 440 includes code that operates to detect resources within information handling system 400, to provide drivers for the resources, to initialize the resources, and to access the resources.

Disk controller 450 includes a disk interface 452 that connects the disk controller to HDD 454, to ODD 456, and to disk emulator 460. An example of disk interface 452 includes an Integrated Drive Electronics (IDE) interface, an Advanced Technology Attachment (ATA) such as a parallel ATA (PATA) interface or a serial ATA (SATA) interface, a SCSI interface, a USB interface, a proprietary interface, or a combination thereof. Disk emulator 460 permits SSD 464 to be connected to information handling system 400 via an external interface 462. An example of external interface 462 includes a USB interface, an IEEE 4394 (Firewire) interface, a proprietary interface, or a combination thereof. Alternatively, solid-state drive 464 can be disposed within information handling system 400.

I/O bridge 470 includes a peripheral interface 472 that connects the I/O bridge to add-on resource 474, to TPM 476, and to network interface 480. Peripheral interface 472 can be the same type of interface as I/O channel 412 or can be a different type of interface. As such, I/O bridge 470 extends the capacity of I/O channel 412 when peripheral interface 472 and the I/O channel are of the same type, and the I/O bridge translates information from a format suitable to the I/O channel to a format suitable to the peripheral channel 472 when they are of a different type. Add-on resource 474 can include a data storage system, an additional graphics interface, a network interface card (NIC), a sound/video processing card, another add-on resource, or a combination thereof. Add-on resource 474 can be on a main circuit board, on separate circuit board or add-in card disposed within information handling system 400, a device that is external to the information handling system, or a combination thereof.

Network interface 480 represents a NIC disposed within information handling system 400, on a main circuit board of the information handling system, integrated onto another component such as I/O interface 410, in another suitable location, or a combination thereof. Network interface device 480 includes network channels 482 and 484 that provide interfaces to devices that are external to information handling system 400. In a particular embodiment, network channels 482 and 484 are of a different type than peripheral channel 472 and network interface 480 translates information from a format suitable to the peripheral channel to a format suitable to external devices. An example of network channels 482 and 484 includes InfiniBand channels, Fibre Channel channels, Gigabit Ethernet channels, proprietary channel architectures, or a combination thereof. Network channels 482 and 484 can be connected to external network resources (not illustrated). The network resource can include another information handling system, a data storage system, another network, a grid management system, another suitable resource, or a combination thereof.

Management device 490 represents one or more processing devices, such as a dedicated baseboard management controller (BMC) System-on-a-Chip (SoC) device, one or more associated memory devices, one or more network interface devices, a complex programmable logic device (CPLD), and the like, which operate together to provide the management environment for information handling system 400. In particular, management device 490 is connected to various components of the host environment via various internal communication interfaces, such as a Low Pin Count (LPC) interface, an Inter-Integrated-Circuit (I2C) interface, a PCIe interface, or the like, to provide an out-of-band (OOB) mechanism to retrieve information related to the operation of the host environment, to provide BIOS/UEFI or system firmware updates, to manage non-processing components of information handling system 400, such as system cooling fans and power supplies. Management device 490 can include a network connection to an external management system, and the management device can communicate with the management system to report status information for information handling system 400, to receive BIOS/UEFI or system firmware updates, or to perform other task for managing and controlling the operation of information handling system 400.

Management device 490 can operate off of a separate power plane from the components of the host environment so that the management device receives power to manage information handling system 400 when the information handling system is otherwise shut down. An example of management device 490 include a commercially available

BMC product or other device that operates in accordance with an Intelligent Platform Management Initiative (IPMI) specification, a Web Services Management (WSMan) interface, a Redfish Application Programming Interface (API), another Distributed Management Task Force (DMTF), or other management standard, and can include an Integrated Dell Remote Access Controller (iDRAC), an Embedded Controller (EC), or the like. Management device **490** may further include associated memory devices, logic devices, security devices, or the like, as needed or desired.

FIG. **5** illustrates a flow of a method **500** for handling revocations of certificates associated with declarative requests according to at least one embodiment of the present disclosure, starting at block **502**. It will be readily appreciated that not every method step set forth in this flow diagram is always necessary, and that certain steps of the methods may be combined, performed simultaneously, in a different order, or perhaps omitted, without varying from the scope of the disclosure.

At block **504**, a determination is made whether a revocation for a certificate has been received. In an example, the certificate may be for any type of request, such as a declarative request. A declarative request or command may involve an action to be performed for an extended amount of time. In certain examples, there may be certain situations or circumstances that a declarative request should continue even if the associated certificate has been revoked. For example, if a service initiated by a declarative request is an important service in an endpoint node, this service should not suddenly stop working just because the associated certificate has been revoked. In certain examples, an owner or administrator of an endpoint may explicitly revoke certificate for any suitable number of reasons including, but not limited to, a user retiring, and the user being fired. The administrator may utilize a priority field or other feature of the revocation to indicate whether the service associated with the request should be immediately stopped or may continue.

At block **506**, a determination is made whether a revocation priority indication field is included within the revocation of the certificate for the declarative request. In an example, a revocation priority indication field may be any suitable field in the revocation to indicate whether a special action should be taken based on the revocation. A type of revocation priority indication in the revocation field may be a priority flag, a priority level indicator, or the like.

If a revocation priority indication field is located within the revocation, a determination is made whether a priority level is above a threshold at block **508**. In an example, if the priority indication is a priority flag, the priority level may be high if the flag is in a first state, such as a high state, and the priority level may be low if the flag is in a second state, such as a low state. In this example, the priority level may be above the threshold if the priority flag is in a high state. If the priority indication is a priority level, the priority level may be one of multiple priority levels. In an example, the priority levels may increase from a low priority to a next higher priority, to any suitable number of additional next higher priorities, and to a highest priority. For example, the priority level may be a particular number and the threshold may be a number associated with the service, the endpoint, or the like. In certain examples, the threshold may be different for different services or endpoints based on different threshold numbers being assigned to the different services or endpoints.

If the priority level is above the threshold, the certificate may be revoked or terminated immediately at block **510** and

the flow may end at block **512**. In certain examples, the revocation may be above the threshold for any suitable reason including, but not limited to, a user associated with the certificate being fired. In an example, the user may be fired for malicious activity, such as being a corporate spy. Based on the user being fired for malicious activity, the administrator may want the services associated with a declarative request from the user to be revoked or terminated immediately. In this situation, the administrator may include a priority level that is above the threshold in the revocation priority indication field of the revocation for the associated certificate.

If the priority level is below the threshold, a warning or flag may be set for the certificate at block **514** and the flow may end at block **512**. In certain examples, an administrator may set the priority to a level below the threshold for any suitable reason including, but not limited to, a user being transferred to a different area of the company, user having retired, and the user quitting. In an example, the warning or flag may be stored in a memory of the endpoint, and the warning or endpoint may be utilized during later operations. For example, if an administrator performs an audit of certificates associated with different services or endpoint node, the flag or warning set in block **514** may be provided to the administrator for further analysis.

Referring back to block **506**, if the revocation priority indication field is not set, the revocation for the declarative request is scanned for one or more keywords at block **516**. In an example, a keyword may be utilized to identify an action or actions to take with respect to a certificate identified in the revocation. In certain examples, any suitable keyword may be utilized including, but not limited to, kill, cancel, fired, retired, transferred, quit, flag, and set warning.

If one or more of the keywords are not found in the revocation, the flow may continue as described above with respect to block **514**. In one or more of the keywords are found in the revocation, an action or operation associated with the keyword is performed at block **520** and the flow ends at block **512**. Based on the keyword or keywords indicating that the user was fired for malicious activity, the associated action or operation may be to immediately revoke or terminate services for a declarative request in a certificate associated with the revocation. Based on the keyword or keywords indicating that the user was transferred to a different area of the company, retired, quit, or the like, the associated action or operation may be to set a warning or flag for the certificate. In an example, the warning or flag may be stored in a memory of the endpoint, and the warning or endpoint may be utilized during later operations. For example, if an administrator performs an audit of certificates associated with different services or endpoint nodes, the flag or warning set in block **520** may be provided to the administrator for further analysis.

Although only a few exemplary embodiments have been described in detail herein, those skilled in the art will readily appreciate that many modifications are possible in the exemplary embodiments without materially departing from the novel teachings and advantages of the embodiments of the present disclosure. Accordingly, all such modifications are intended to be included within the scope of the embodiments of the present disclosure as defined in the following claims. In the claims, means-plus-function clauses are intended to cover the structures described herein as performing the recited function and not only structural equivalents, but also equivalent structures.

What is claimed is:

1. An endpoint node of a multiple node environment, the endpoint node comprising:

a storage configured to store data associated with a certificate of the endpoint node; and

a processor to communicate with the storage, the processor to:

receive a revocation associated with the certificate;

determine a priority level for the revocation;

determine whether the revocation includes a revocation priority indication field; and

in response to the revocation including the revocation priority indication field, determine whether a priority level is above a threshold;

in response to the revocation not including the revocation priority indication field, scan the revocation for a plurality of keywords; and

based on the priority level, perform an action associated with the certificate, wherein the action corresponds to one of a plurality of priority levels.

2. The endpoint node of claim 1, in response to the priority level being above the threshold, the processor further to immediately revoke the certificate.

3. The endpoint node of claim 2, wherein a service associated with the certificate is immediately terminated based on the certificate being revoked.

4. The endpoint node of claim 1, in response to the priority level being below the threshold, the processor further to: set a flag as the data for the certificate in the storage.

5. The endpoint node of claim 1, wherein in response to the keywords not being found in the revocation, the processor to: set a flag as the data for the certificate in the storage.

6. The endpoint node of claim 1, wherein in response to one of the keywords being found in the revocation, the processor to: perform an action associated with the one of the keywords, wherein the action is for the certificate.

7. A method comprising:

receiving, by a processor of an endpoint, a revocation associated with a certificate;

determining a priority level for the revocation;

determining whether the revocation includes a revocation priority indication field;

in response to the revocation including the revocation priority indication field, determining whether the priority level is above a threshold;

in response to the revocation not including the revocation priority indication field, scanning the revocation for a plurality of keywords; and

based on the priority level, performing an action associated with the certificate, wherein the action corresponds to one of a plurality of priority levels.

8. The method of claim 7, in response to the priority level being above the threshold, the method further comprises: immediately revoking the certificate.

9. The method of claim 8, wherein a service associated with the certificate is immediately terminated based on the certificate being revoked.

10. The method of claim 7, in response to the priority level being below the threshold, the method further comprises: setting a flag as the data for the certificate in the storage.

11. The method of claim 7, wherein in response to the keywords not being found in the revocation, the method further comprises: setting a flag as the data for the certificate in the storage.

12. The method of claim 7, wherein in response to one of the keywords being found in the revocation, the method further comprises: performing an action associated with the one of the keywords, wherein the action is for the certificate.

13. A method comprising:

receiving, by a processor of an endpoint, a revocation associated with a certificate;

determining whether the revocation includes a revocation priority indication field;

in response to the revocation including the revocation priority indication field, determining whether a priority level is above a threshold;

in response to the priority level being above the threshold, immediately revoking the certificate;

in response to the priority level being below the threshold, setting a flag as the data for the certificate in the storage; and

in response to the revocation not including the revocation priority indication field, the method further comprises: scanning the revocation for a plurality of keywords.

14. The method of claim 13, wherein in response to the keywords not being found in the revocation, the method further comprises: setting a flag as the data for the certificate in the storage.

15. The method of claim 13, wherein in response to one of the keywords being found in the revocation, the method further comprises: performing an action associated with the one of the keywords, wherein the action is for the certificate.

* * * * *