A Security Certification Scheme for Information Centric Networks

Marco Anisetti, Senior Member, IEEE, Claudio A. Ardagna, Senior Member, IEEE, Filippo Berto, Ernesto Damiani, Senior Member, IEEE

Abstract—**Information-Centric Networking is an emerging alternative to host-centric networking designed for large-scale content distribution and stricter privacy requirements. Recent research on Information-Centric Networking focused on the protection of the network from attacks targeting the content delivery protocols, while assuming genuine content can always be retrieved from trustworthy nodes. This assumption opens the door to poisoning attacks that can substantially impair the operation of the network. In addition, current security techniques are often focused on specific attacks and countermeasures, missing the big picture. In this paper, we depart from the assumption on the trustworthiness of network nodes and propose a novel certification methodology for Information-Centric Networks that supports continuous security verification of non-functional properties. Our methodology provides a complete and detailed view of the network security status, increasing the trustworthiness of the network and its services. The proposed approach builds on an enhanced certification model capturing the evolution of the system over time. It also defines certification services that fully integrate with existing networks to collect evidence on the target of certification and carry out the certification process. It finally propose two certification processes, a standard centralized version based on a certification authority managing the whole process, and a decentralized and collaborative version to exploit the caching capabilities of Information-Centric Networks and improve the system performance. Efficency, performance, and soundness of our approach have been experimentally evaluated in a simulated Named Data Networking (NDN) network targeting property availability.**

Index Terms—**Content-centric networking; named data networking; security; certification;**

I. Introduction

Information-Centric Networking (ICN) is a network paradigm that addresses contents in the network using unique URI-like names. ICN is increasingly adopted as a substitute for the common TCP/IP network stack when in-protocol content distribution and privacy features are paramount [1]–[6]. The shift from the traditional host-based paradigm of the TCP/IP stack removes the requirement of uniquely identifying the network nodes involved in the communication through network addresses and compresses the network, transport, and application layers into a single hybrid layer. ICN networks are also agnostic over the transmission mean, allowing the same stack to be adapted to different physical network layers, such as Ethernet, WiFi, Bluetooth or other network protocols. The main advantage of ICN-based networks is their in-protocol distributed caching solution, that is, while TCP/IP based media sharing solutions are not scalable and require Content Distribution Networks (CDNs) to satisfy large client bases, in ICNs each node can cache contents, immediately satisfying multiple clients requests and reducing the stress on the rest of the network. ICN also reduces privacy concerns since its packets do not carry user identifying information.

In the last decade, the research and development community has made huge steps forward in the implementation of high quality, high performance, and functionally robust ICN networks [6], [7]. Also security of ICN has been deeply analyzed targeting specific attacks [8], [9] and countermeasures [10]–[13]. Monitoring solutions have been proposed to monitor the status of a network in deep, consisting of software and hardware tools [14]–[18] that measure the state of its nodes and their communication links (e.g., load, traffic utilization, exposed services and uptime). Several protocols have been applied to monitor networks, like SNMP and ICMP, supporting easy detection and configuration of network nodes and aggregation of monitoring measurements.

Monitoring is however not always enough to measure the security status of a network. Security assurance has then been widely adopted to improve the security status of a target system, providing justifiable confidence that it behaves as expected despite failures and attacks. In this context, certification stands out as a preferred assurance technique, collecting evidence about a system to prove a specific property on it. The collected evidence is evaluated by a trusted Certification Authority (CA) that awards a certificate to the system proving a specific (set of) property. Certification schemes have been applied beyond traditional software (Common Criteria [19]) and targeted web and cloud services [20]–[23] and, more recently, complex service compositions, where the collected evidence is based on monitoring, testing, or formal proofs. The peculiarities of recent certification schemes [20], [21], being dynamic, continuous, lightweight, make them an opportunity even for verifying properties of complex network protocols. However, to the best of our knowledge, security assurance and certification of ICN are still in their infancy. Transparency and trustworthiness of information-centric networks become then a major hurdle against their widespread adoption and can open the door to persistent threats that affect the network behavior to its foundation. In addition, weaknesses to poisoning attacks and system malfunctioning can impair the entire network operation [14], [17], [24].

Marco Anisetti, Claudio A. Ardagna, Filippo Berto, and Ernesto Damiani are with the Department of Computer Science, Università degli Studi di Milano, Milan, Italy. Ernesto Damiani is also with Center for Cyber-Physical Systems, EECS Department, Khalifa University of Science and Technology, Abu Dhabi, UAE.
E-mail: {firstname.lastname}@unimi.it,ernesto.damiani@ku.ac.ae

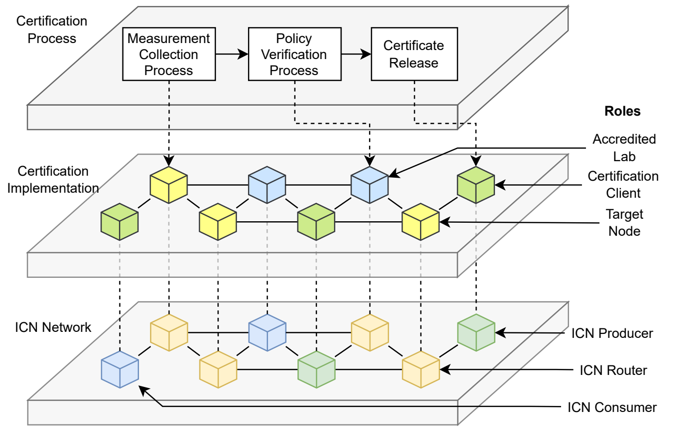


Figure 1: A layer-based view of our System model.

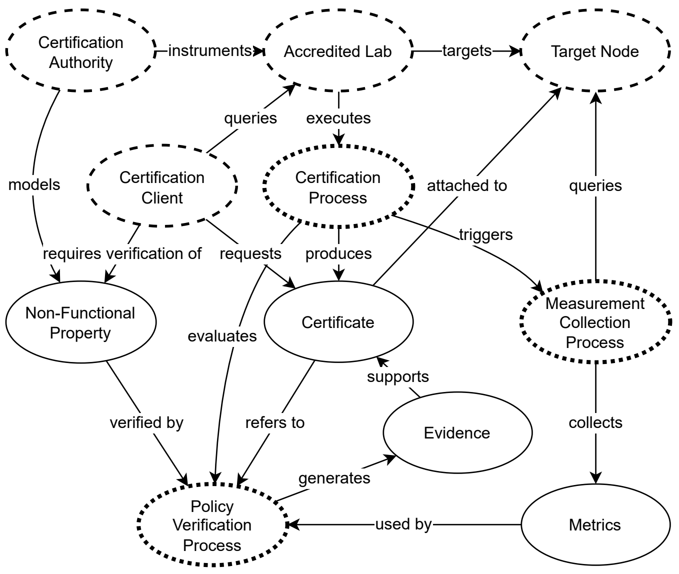


Figure 2: Certification methodology. Dashed lines refer to certification roles, dotted lines refer to certification process, and black lines refer to components of the certification methodology.

In this paper, we present a certification methodology for information-centric networks continuously certifying non-functional properties of network nodes in operation. Our methodology is based on an abstract certification model that provides all building blocks for network certification, from evidence collection based on metrics, to certification policies modeling the expected behavior to be certified on a specific set of network nodes and non-functional properties modeling the expected behavior of the whole system. Our certification methodology is continuous meaning that non-functional properties are verified over time to capture any changes in the security status of the system, and corresponding nodes, and take corrective actions. It can support an effective Quality of Service (QoS) approach, where network functioning is adapted to evolving conditions, increasing network trustworthiness and quality. It complements modern composite applications based on microservices, paving the way to a new generation of certified compositions tightly intertwined with networking technologies [4], [5], [25], [26].

This paper extends our network-level certification approach in [27] and its contribution is threefold. It first provides an enhanced certification model capturing the evolution of the system over time (Section III), and new services providing certification functionalities (Section IV), which are fully integrated with the original protocols reducing the performance impact on the overall network. It then defines two deployment models (Section V and Section VI), centralized and decentralized, which fully integrate with ICNs improving their trustworthiness. It finally proposes an implementation of the proposed approach that is fully tested in an ICN network (Section VII), providing a discussion on application scenarios of interest for ISPs or cloud providers offering certified services (Section VIII).

II. Certification Methodology and System Model

Our system model is a standard ICN network extended with the certification methodology presented in this paper. In ICN, content consumers (ICN Consumer) request contents by sending *interest packets* to neighbour nodes only including the content name and optional request configuration parameters. Content producers (ICN producer) register a series of prefixes for which they can respond with *data packets*, containing the content itself and a signature that guarantees the integrity and non-repudiability of the data. *Data packets* shared through the network can be cached by any nodes in the packet path: each router (ICN Router) that receives an *interest packet* first checks for a matching *data packet* in its Content Store (CS) and alternatively forwards the request to its neighbours. Figure 1 graphically shows our complete system model as a traditional ICN network extended with the certification methodology in Figure 2. Any nodes in the ICN network can play a role in our certification methodology as follows: i) the *Certification Client* asking for a certification process execution, ii) the *Certification Authority (CAs)*, responsible for the entire certification process, and iii) the *Target Node*, as the target of the certification process. The *Certification Process* orchestrates two sub-processes, the *Measurements Collection Process* and the *Policy Verification Process* to release and distribute certificates. The certification client queries the certification authority to execute a certification process on a set of target nodes to prove a given non-functional property. The target nodes are empowered with our Measurements Collection Process that produces metrics to be evaluated by the Policy Verification Process. The Policy Verification Process generates evidence based on metrics to support the award of the certificates that are attached to the target nodes and retrieved by the certification clients.

More in detail, the certification roles have the following responsibilities.

- Certification Authority is a (set of) network node orchestrating the entire certification process and implementing the Policy Verification Service. It is responsible for the verification of policies, and the production of certificates assessing the collected evidence. CAs listens for certification requests from certification clients. For each valid certification request, CA starts a certification process, as described in Sections V and VI, and once terminated returns a list of the produced certificates names, each of which is associated with one of the target nodes. Depending on the network and configuration scale, CAs has a different view on the status of the whole network; an in depth analysis of the possible solutions is presented in Sections V-A and VI-A.

- Certification Client is a network client having interest in the certification process. Any devices in the network, including those that are not acting as routers, can request a certification to one or multiple CAs to verify properties over a set of target nodes. The obtained information helps the client in its internal processes by identifying nodes suitable for the deployment of a service, suggesting a more efficient or secure routing path, improving privacy by excluding untrusted nodes, to name but a few.
- Target node is a network node whose status can be measured by CAs and can be targeted by certification requests from certification clients. It runs the Measurements Collection Service that measures its internal state and exposes it to trusted CAs.[1]

The three roles are independent from each other and a ICN node can play multiple roles; for instance, a certification client can request a certification to a CA for its own status or be a CA itself. A node can act as a CA for its own status (self-certification), although the produced certificates cannot be considered reliable from other clients. This decoupling property is particularly interesting in the case of a totally decentralized certification model as we discuss in Section VI-A.

Our certification methodology is built on three main building blocks, an abstract certification model (Section III), the certification services (Section IV), the certification process (Sections V and VI), which completes our system model. In the following, we consider NDN, the most common and studied ICN protocol, as the reference protocol.

III. Abstract Certification Model

Similarly to what happen for complex service-based systems, an abstract certification model is defined to cope with the network complexity, where each network implementation differs from the others and needs a specific certification process. Figure 3 shows the certification model at the basis of our certification process, where: *i) Certificate* is the result of a *Policy* verification aimed to prove a behaviour supporting a given *non-functional property* to be certified; *ii)* each *Policy* is composed of set of *Rules* that are expressed based on *Metrics* captured on the system under certification; *iii)* each metric produces the *Evidence* stored in the certificate to support the given non-functional property.

A. Metrics

A metric, as retrieved by the Measurement Collection Process, is a function defined i) on a single node focused

[1]Encryption can be adopted to preserve confidentiality over the shared measurements.

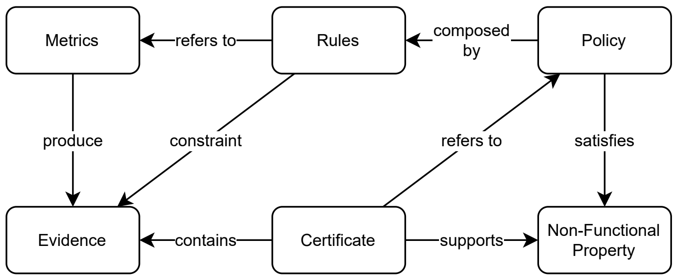


Figure 3: Abstract certification model.

on measuring its attributes (e.g., the amount of available memory), or ii) on multiple nodes to measure their interaction (e.g., minimum network bandwidth between any points). A metric function should be compliant with key requirements that influence its accuracy and correctness:

- each metric must focus on a singular aspect of the system, preventing unnecessary duplication;
- the computational effort necessary must be negligible, compared to the descriptive value of the output;
- metrics must be calculated in parallel, minimizing the total waiting time;
- metrics must show the temporal evolution of the system;
- metrics must not interfere with the system processes, including the network protocol;
- metrics should be as much as possible scenario and user independent.

Given the fact that the measurements used to compute metrics spans over a time frame, metrics are often significantly affected by the interval of time considered during their evaluation. For instance, a metric that calculates the average load in a network node in a large time span may hide significant spikes that a stricter evaluation could identify. Given a specific input time interval, a metric function produces a measurement of a specific aspect of the system for that time interval. Formally, we define a metric as follows.

Definition III.1. Given $\mathbf{N}$ a set of network nodes, $\mathbf{T}$ a time interval and $\mathbb{V}$ the space of possible values that a metric can assume, a metric $\mathbb{M}$ can be defined as:

$$\mathbb{M} : \mathbf{N} \times \mathbf{T} \rightarrow \mathbb{V}$$

We note that a metric $m \in \mathbb{M}$ can be evaluated for any sets of nodes $\mathbf{n} \subseteq \mathbf{N}$ in an interval of time $\mathbf{t} \in \mathbf{T}$, denoted as $m(\mathbf{n}, \mathbf{t})$. Each metric evaluates the *state* function on the target nodes in several time points inside the chosen interval. We also note that, metric values $\mathbb{V}$ are bounded to specific data types and values ensuring a finite and concrete representation. Metrics evaluated on a target system produce a simplified vision of its internal state, hiding unnecessary complexity while extracting significant information.

We note that, evaluating the metric against a chosen subset of nodes in $\mathbf{N}$ in a given instant of time we can map each node to a partial order. This approach helps in efficiently exploring network nodes during the evaluation of a set of policies by defining appropriate heuristics.

$$
\begin{aligned}
D\_Value &::= \langle constant \rangle \,|\, \langle metric\ evaluation \rangle \\
D\_Acc &::= \langle D\_Value \rangle [\langle constant \rangle] \,| \\
&\quad\ \langle D\_Value \rangle [\langle constant \rangle : \langle constant \rangle] \\
D\_Expr &::= \langle D\_Value \rangle \,|\, \langle D\_Acc \rangle \,| \\
&\quad\ \langle D\_Expr \rangle \langle D\_Op \rangle \langle D\_Expr \rangle \,| \\
&\quad\ \langle D\_Transf \rangle (\langle D\_Value \rangle) \\
D\_Op &::= + \,|\, - \,|\, \times \,|\, / \\
D\_Transf &::= \sum \,|\, \prod \,|\, min \,|\, max \,|\, abs \,|\, len \,|\, \ldots \\
D\_Cmp\_Op &::= < \,|\, \leq \,|\, = \,|\, \neq \,|\, \geq \,|\, > \\
B\_Value &::= true \,|\, false \,|\, \langle rule\ evaluation \rangle \,| \\
&\quad\ \langle D\_Expr \rangle \langle D\_Cmp\_Op \rangle \langle D\_Expr \rangle \\
B\_Op &::= \wedge \,|\, \vee \,|\, \equiv \,|\, \oplus \\
B\_Expr &::= \langle B\_Value \rangle \,|\, !(\langle B\_Expr \rangle) \,| \\
&\quad\ \langle B\_Expr \rangle \langle B\_Op \rangle \langle B\_Expr \rangle \\
Rule &::= \langle rule\ name \rangle (t_1, t_2) = \langle B\_Expr \rangle
\end{aligned}
$$

Figure 4: Rules expressed in a BNF notation. For the ke of simplicity, we skipp some trivial non-terminals.

B. Rules

Rules are Boolean functions based on one or more metrics and an interval of time. We formally define a rule as follows.

Definition III.2. Given $\mathbf{N}$ a set of network nodes, $\mathbf{T}$ a time interval, and $\mathbb{B} = \{\mathtt{true}, \mathtt{false}\}$, rule $\mathbb{R}$ is defined as:

$$\mathbb{R} : \mathbf{N} \times \mathbf{T} \to \mathbb{B}$$

Each rule is a Boolean expression grounded on the simplified BNF described in Figure 4.

In this notation, $\langle constant \rangle$ is a value in V, $\langle metric\ evaluation \rangle$ is in the form $m_i(t', n')$, and $\langle rule\ name \rangle$ is an unique rule identifier in the form $r_i(t', n')$. Following the abstraction notation, $t, t' \in \mathbf{T}$ and $n, n' \in \mathbf{N}$. Since several rules can share parts of definition, we included $\langle rule\ evaluation \rangle$ in the BNF, allowing a rule evaluation to be called from within another rule, even on a different time interval. Metric and rule evaluation can only be applied within the time interval and nodes of the original rule, such that $t' \subseteq t$ and $n' \subseteq n$, forbidding recursively defined rules over shifting time intervals.

For each rule a partial order $(\mathbb{R}, \preceq_r)$ that indicates the strictness of the rule is defined as follows.

$$
\begin{aligned}
r_a \preceq_r r_b \iff & \forall \mathbf{n} \in \mathbf{N}, \mathbf{t} \in \mathbf{T} \\
& r_b(\mathbf{n}, \mathbf{t}) \Rightarrow r_a(\mathbf{n}, \mathbf{t})
\end{aligned}
$$

Example III.1. Consider the two rules $r_i(\mathbf{n}, \mathbf{t}) = m(\mathbf{n}, \mathbf{t}) \leq 10$ and $r_j(\mathbf{n}, \mathbf{t}) = m(\mathbf{n}, \mathbf{t}) \leq 5$ where m indicates the maximum Round Trip Time (RTT) in *ms* allowed between the nodes in $\mathbf{n}$ in the time interval $\mathbf{t}$. We can see how r_j is stricter than r_i, since all nodes in which r_j is valid will also be valid for r_i while the opposite is not true, therefore $r_i \preceq_r r_j$.

C. Policy

A policy describes the expected behavior of a group of nodes by indicating a set of rules that should be positively evaluated. We define a policy as a subset in the powerset of the space of all possible rules. More formally we define the set of all policies as

$$\mathbf{P} = \wp(\mathbf{R})$$

Policy are evaluated by combining the output of each of their rules: a policy is verified for a given set of nodes $\mathbf{n} \in \mathbf{N}$ in an interval of time $\mathbf{t} \in \mathbf{T}$ if and only if all of its rules evaluated in n and t produce a positive output. We note that a policy including pairs of conflicting rules, for instance $\{m(\mathbf{t}, \mathbf{n}) = 1, m(\mathbf{t}, \mathbf{n}) = 2\}$, produces a negative output. By contrast, the policy corresponding to the empty set of rules produces a positive output by default, regardless of which set of nodes and time intervals are given as input.

Due to the inherent compositional nature of rules in our model, we can define a partial order $(\mathbf{P}, \preceq_p)$ as follows:

$$
\begin{aligned}
a \preceq_p b \iff & \forall r_a(\mathbf{n_a}, \mathbf{t_a}) \in a \, \exists r_b, \mathbf{t_b}, \mathbf{n_b} \\
& \text{where } r_a \preceq_r r_b, \mathbf{t_a} \subseteq \mathbf{t_b}, \mathbf{n_a} \subseteq \mathbf{n_b} \,| \\
& r_b(\mathbf{n_b}, \mathbf{t_b}) \in b
\end{aligned}
$$

$a \preceq_p b$ if and only if b contains at least the same or stricter rules than a and each rule in b is evaluated on a superset of the time intervals and on a superset of the set of nodes of its counterpart in a.

Policies are defined as sets of rules, therefore we can combine multiple policies together in a single set. Exploiting the partial order $(\mathbf{P}, \preceq_p)$, we can define a policy $\mathbf{P}$ as the Least Upper Bound (LUB) of several policies, producing the equivalent of concatenating the policy rules with the logical *and* operator. This more effective than a simple union as we can shrink multiple version of the same rules to a single stricter one.

Policies can also be used as a selection mechanism for identifying a subset of nodes within the network with peculiar characteristics. Given a target policy $\mathbf{p}$ that we want to validate, we can verify which subsets of network nodes in a set $\mathbf{N}$ verify the policy. A policy-based filter can be generated by combination of multiple policies using the Greatest Lower Bound (GLB) operator such that $\mathbf{p} = glb(\mathbf{P})$, where $\mathbf{P}$ is the set of target policies. This is equivalent to concatenate the policy rules with the logical *or* operator. A typical use case for such an approach is the deployment of a service over a set of nodes all ensuring a policy such as data replication grade or channel encryption.

D. Certificate

Certificate is the outcome of the certification process and are composed of:

- the policies a certificate proves;
- the validation parameters, such as the target set of nodes and the time interval;
- the evidence supporting the certificate and verified by the involved policies.

Definition III.3. We define a certificate as a tuple $\langle \mathbf{t}, \mathbf{n}, \mathbf{P}, \mathbf{m} \rangle$ where $\mathbf{t}$ is a time interval in $\mathbf{T}$, $\mathbf{n}$ is a set of nodes in $\mathbf{N}$, $\mathbf{p}$ is a policy that has been verified in the interval $\mathbf{t}$ for all nodes in $\mathbf{n}$, and $\mathbf{m}$ is the set of evidence related to nodes in $\mathbf{n}$ evaluated during the verification of policy $\mathbf{p}$.

$\mathbf{m}$ is optional to protect against the release of sensitive information. A certificate is awarded by a CA if and only if its policy $\mathbf{P}$ has been successfully verified.

E. Non-Functional Property

A Non-functional property is an abstract concept that identify the expected status of a system. Our model treats property as a generalization of one or multiple verified policies, meaning that a group of nodes $\mathbf{n}$ has a property p in the interval $\mathbf{t}$ if a selected set of policies has been verified. Property is verified according to different and non-intersecting policies. For instance, property *confidentiality* is verified if all policies ensuring an encrypted network traffic are verified. Properties that describe the same concept can have several degrees of satisfaction depending on which of the associated policies have been verified. In the previous example, a policy that ensures all traffic is encrypted using an RSA key of length 2048 bits is weaker than a policy requiring a key length of 4096 bits.

We note that different clients can define properties using different sets of policies, depending on their requirements and use cases.

IV. Certification Services

The abstract certification model in Section III is instantiated in the certification processes in Sections V and VI using the the measurement collection and policy verification services described in the remaining of this section.

A. Measurement Collection Service

The measurement collection service allows CAs to query the internal state of any target nodes through their metrics. There are three different ways to implement our measurement collection service: *i)* pull, *ii)* push, and *iii)* hybrid. The pull solution works as follows:

1) each target node executes a service that binds to a known prefix listening for measurement requests in the form `/.../`$\langle$*node*$\rangle$`/measure/`$\langle$*metric*$\rangle$`/`$\langle$*params*$\rangle$, where $\langle$*metric*$\rangle$ uniquely identifies the metric chosen by the CA, while $\langle$*parameters*$\rangle$ indicates the metric parameters such as the interval of time used to measure;
2) a CA can repeatedly send interest requests to a node with the necessary fields to query its state;
3) when a target node receives a valid measurement request, it replies with a data packet containing the result of the metric evaluation with the given parameters;
4) data packets can be cached, supporting the efficient distribution of the measurements to the CAs that sent a matching request;
5) the contents of the data packets can be encrypted to preserve confidentiality.

This approach has the disadvantage of requiring the CAs to know the prefix of a possibly large number of nodes, but leaves total control on the CAs over which metrics need to be queried and when.

The push solution works as follows:

1) each CA executes a service that binds to a known prefix listening for measurement updates request in the form `/.../`$\langle$*node*$\rangle$`/update/`$\langle$*measurements*$\rangle$, where $\langle$*measurements*$\rangle$ is an encoded list of measurements;
2) each target node hosts a service that periodically evaluates all metrics using a fixed set of parameters;
3) after each iteration, the node sends the CA an update over the newly obtained measurements.

This solution moves the responsibility of maintaining synchronization from the CA to the nodes and reduces synchronization delays. Unfortunately, it also increases the total amount of data sent, as even unnecessary measurements can be contained in the packets. Moreover the ICN caching mechanism cannot be used for requests, therefore the total network traffic would increase significantly in the case of multiple CAs. While this method is possible, it can experience efficiency and scalability issues.

The hybrid solution combines the pull and push implementations. Depending on the amount of updated data to share and the size of the network, it can improve the synchronization with a limited increase in traffic. It works as follows:

1) when an update is ready a node sends a small notification request to the CAs;
2) then, the CAs can request the status of the node as in the pull solution.

This addition helps in synchronizing the two parties, reducing the idle time from when the information is ready and when it is collected by the CAs, while maintaining the advantages of ICN caching mechanism. However, it also introduces complexity and additional traffic compared to the push solution.

Our model implements the pull solution and supports the hybrid one, providing the best tradeoff in complexity and network usage. Each node in the network exposes a predefined prefix in the form `/.../`$\langle$*node*$\rangle$`/measure/list`, which returns the list of available metrics and a prefix in the form `/.../`$\langle$*node*$\rangle$`/measure/`$\langle$*metric*$\rangle$`/[to]/[from]` allowing other nodes to query its metrics. Depending on the type of metric, the two parameters `to` and `from` can be optional.

B. Policy Verification Service

Policy Verification Service formalizes how clients can request verifications to CAs and the certification response. The implementation of such service with the pull approach can be summarized as follows:

1) each CA executes a service that binds to a known prefix listening for certification requests in the form `/.../`$\langle$*node*$\rangle$`/verify/`$\langle$*parameters*$\rangle$, where $\langle$*parameters*$\rangle$ indicates the verification parameters, including which policy, time interval, and nodes subset to use in the evaluation;

2) for each valid certification request, the CA service executes a certification process, as described in Sections V and VI, which produces a list of content names, each pointing to a certificate;
3) once the certification process is terminated, the service responds to the client request with the list of certificates produced in the form of content names.

These responses can be cached, allowing other clients with matching requests to be immediately satisfied.

V. Centralized Certification Process

We instantiate the abstract certification model in Section III using the certification services in Section IV and the centralized certification process in Figure 5(a), where the certification authority mediates all certification activities.

A. Network Model

Figure 5(a) presents a centralized network model at the basis of a centralized certification process, where a single CA is responsible for all certification activities and any nodes can be both certification client and target. While this approach introduces a single point of failure on the CA, which also becomes a significant bottleneck in larger networks, it introduces some major advantages as follows.

- *Service discovery.* The CA knows the prefixes exposed by the target nodes to query their metrics. With a centralized network a common approach to service discoverability is to use a registration approach so that *i)* the CA prefix is known to any nodes in the network and *ii)* each node that connects to the network notifies its prefix to the CA through a registration request. This solution is simple to implement and does not rely on protocol-specific service discovery features.
- *Simpler certificate distribution.* The CA distributes the certificates it produces as contents of a self-owned predefined prefix. The nodes that are awarded with a certificate are notified by the CA. This solution allows any clients in the network to query for a certificate knowing only the CA's base prefix, while exploiting the caching capabilities of the network for an efficient distribution of common data requested by multiple clients.
- *Results caching.* The CA is the only actor receiving certification requests and producing corresponding certificates. Caching of previously verified policies is effective, reducing the number of network requests necessary to evaluate new requests.

B. Certification Process

Algorithm 1 presents the centralized certification process and corresponding policy verification, where a is a certification client, c a CA, $\mathbf{b}$ a subset of nodes, and $\mathbf{t}$ a time interval. Figure 6 visually represents the steps of the algorithm in an example network.

A policy verification request sent by a certification client (line 22) is handled by a CA. The certification process starts by checking whether the locally cached certificates already verify the target policy; an initially empty policy is expanded by applying the GLB operator (line 2-6). If the target policy is smaller in $\preceq_p$ than the obtained set, the target policy is verified and the list of cached certificates returned as output (lines 7-8). If the cached certificates are insufficient, the certification process proceeds by evaluating each rule that is not verified yet (lines 9-13). The certification process finally checks if all the rules have been verified; if yes, it generates and returns a certificate to the client, otherwise, it returns an empty list (lines 14-19). Finally, the client receives the list of certificates (line 23).

Algorithm 1 Centralized certification process

```
1:  function HANDLE REQUEST(policy: $\mathbf{p}(\mathbf{b},\mathbf{t})$)
2:      $\mathbf{V} = \emptyset$
3:      for all $\mathbf{v} \in$ cached_certificates() do
4:          for all rule $r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})$ do
5:              if $r(\mathbf{b},\mathbf{t}) \preceq_p \mathbf{v}$ then
6:                  $\mathbf{V} = GLB(\mathbf{V},\mathbf{v})$
7:      if $\mathbf{p}(\mathbf{b},\mathbf{t}) \preceq_p \mathbf{V}$ then
8:          return $\mathbf{V}$
9:      for all rule $r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})$ do
10:         if $r(\mathbf{b},\mathbf{t}) \not\preceq_p \mathbf{V}$ then
11:             for all metric evaluation $m(\mathbf{b}',\mathbf{t}') \in r(\mathbf{b},\mathbf{t})$ do
12:                 m_res$[m] = m(\mathbf{b}',\mathbf{t}')$
13:     p_valid $= \bigwedge_{r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})} r(\mathbf{b},\mathbf{t})$
14:     if p_valid then
15:         cert = new_certificate($\mathbf{p}(\mathbf{b},\mathbf{t})$)
16:         $\mathbf{V} = \{$cert$\}$
17:     else
18:         $\mathbf{V} = \emptyset$
19:     return $\mathbf{V}$
20:
21: function REQUEST VERIFICATION(policy : $\mathbf{p}(\mathbf{b},\mathbf{t})$)
22:     res = send_request(policy)
23:     certs = collect_certs(res)
```

VI. Decentralized Certification Process

We the abstract certification model in Section III using the certification services in Section IV and the decentralized certification process in Figure 5(b), where multiple certification authorities manage the certification activities and their output can be independently combined.

A. Network Model

Figure 5(b) presents a decentralized network model at the basis of a decentralized certification process, where a every node in the network can act as a CA making the certification process completely decentralized. This approach eliminates the single point of failure of the centralized network model and allows clients to request certifications to several nodes in the network. It also enables clients to selectively specify the CA node on the basis of its trust level, possibly requiring the CA to filter out those certificates produced by untrusted sources.

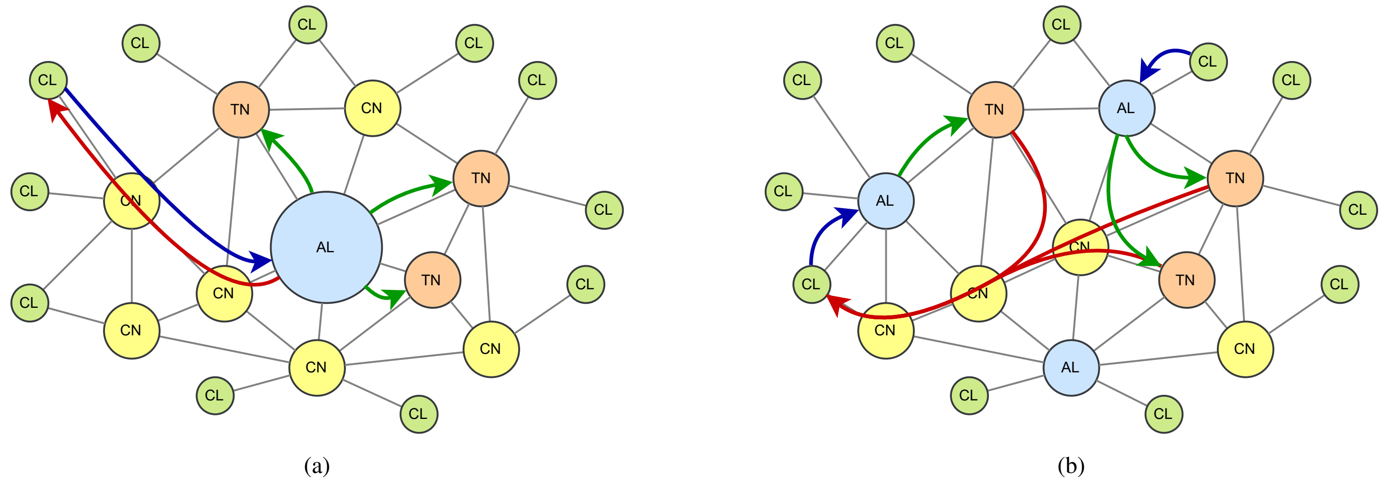


Figure 5: Abstract certification model instantiation: (a) centralized certification process, (b) decentralized certification process. Clients (CL) request a policy verification on a set of target nodes (TNs) to CAs.

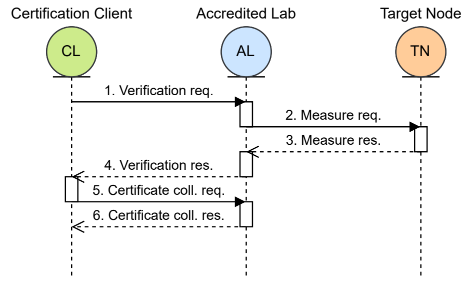
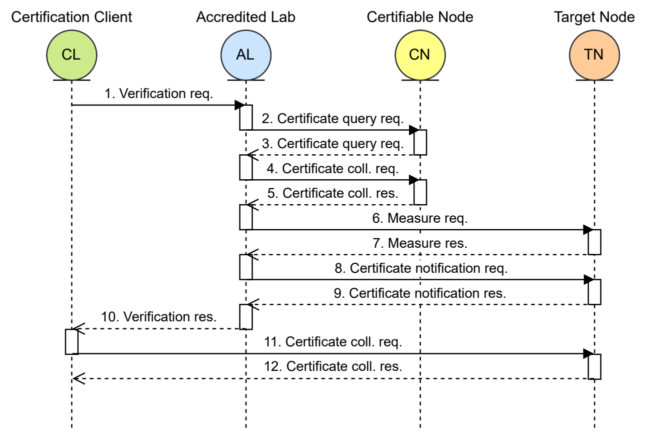


Figure 6: Centralized certification process: Communication flow.

The decentralized approach provides additional advantages as follows.

- *Service discovery.* The distributed network model extends the previous one by including an automatic service discovery mechanism, which allows each node to search for CAs nodes in their proximity. As discussed in Section IV, CAs, as well as all nodes in the network, exposes services on predefined prefixes. Depending on the capabilities of the underlying ICN protocol, the clients can either use in-protocol service discovery features to identify nodes with such prefixes, like Named Data Link State Routing (NLSR) in NDN, or send discovery requests to each network interface in a multicast fashion with an increasing maximum hops limit. This approach allows clients to operate independently from a central authority and to self-organize in spatially localized sub-networks.
- *Decentralized certificate distribution.* The tasks of storing and distributing the awarded certificates are outsourced from the CA to the target nodes. A CA that successfully certified a node sends a registration request submitting the signed certificate as a parameter of a predefined prefix of the target node in the form `/.../`$\langle$*node*$\rangle$`/register/`$\langle$*certificate*$\rangle$. A node that receives such a request stores the certificate in a local storage using a unique identifier, exposes the certificate as a content on a predefined prefix in the form `/.../`$\langle$*node*$\rangle$`/certificate/`$\langle$*id*$\rangle$, and responds to the registration request with the complete content name. The CA can then collect the list of certificate names, one for each target node, and answer to its certification client. The target nodes also expose their list of awarded certificates including the policy and parameters used on a predefined prefix in the form `/.../`$\langle$*node*$\rangle$`/certificates/[filter]`, allowing CAs to easily query their storage for previous certificates that can used as a baseline for further verification. The certificates distribution is then decoupled from the CAs that produced them and rather controlled by the target nodes, while maintaining the effectiveness of the caching capabilities of ICN. The decentralized approach increases the total number of requests necessary to produce a certificate in small networks but strongly reduces traffic originated by packets being forwarded in large networks, with respect to what expected in the centralized solution. In other words, it prevents long paths from the periphery of the network to the central CA and vice versa.
- *Result caching.* Caching of previous results is more effective than the one in the centralized model. Each CA can store the certificates produced by itself and query others CAs's certificates directly to the target nodes. This approach enables a distributed and cooperative certification service, where each verification can exploit previously verified policies to produce new knowledge.
- *Policy query service.* The policy verification process in our distributed network model employs an additional network service to allow CAs to query target nodes for stored certificates matching a minimum policy. These targets listen for query requests on a predefined prefix in the form `/.../`$\langle$*node*$\rangle$`/certificates/`$\langle$*filter*$\rangle$, where filter is an encoded policy definition. When a request is received the node iterates over its certificates, checks which ones pass the filter, collects their content name in a list and returns it to the requester. This



Certification Client
CL
Accredited Lab
AL
Certifiable Node
CN
Target Node
TN
1. Verification req.
2. Certificate query req.
3. Certificate query req.
4. Certificate coll. req.
5. Certificate coll. res.
6. Measure req.
7. Measure res.
8. Certificate notification req.
9. Certificate notification res.
10. Verification res.
11. Certificate coll. req.
12. Certificate coll. res.

Figure 7: Decentralized certification process: Communication Flow.

solution allows CAs to rapidly collect information about previously verified policies over their target nodes without requiring a network-wide level of synchronization over the status of certificates.

B. Certification Process

Algorithm 2 presents the decentralized certification process and corresponding policy verification, where a is a certification client, c a CA, $\mathbf{b}$ a subset of nodes, and $\mathbf{t}$ a time interval.

A policy verification request sent by a certification client (line 32) is handled by a CA. The certification process starts by checking whether the locally cached certificates already verify the target policy: an initially empty policy is expanded by applying the GLB operator (line 2-6). If the target policy is smaller in $\preceq_p$ than the obtained set, the target policy is verified and the list of cached certificates returned as output (lines 7-8). If the cached certificates are insufficient, the certification process queries the neighbour nodes for certificates that satisfy the inner rule evaluations and merges them to the previous partial solution (lines 9-13). If the obtained solution is sufficient, it returns the list of certificates (lines 14-15); otherwise, the certification process proceeds by evaluating each inner rule that is not verified yet (lines 16-21). The certification process then checks if all the inner rule evaluations have been verified; if yes, it generates and returns a certificate to the client, otherwise, it returns an empty list (lines 22-29). Finally, the client receives the list of certificates (line 33).

The evaluation of certificates stored locally or on neighbour nodes (lines 9-13) could be insufficient to verify the target property. This means that at least one among the set of rules, the set of nodes, or the time interval causes a failure in the evaluation. Automatic and timely identification of the cause of the failure permits to rapidly identify which inner rule evaluations are missing to meet the policy requirements and, in turn, ask them to other nodes or manually verify them. We present two examples where the evaluation of certificates stored locally or on neighbour nodes prevent the CA to re-evaluate already verified policies or to reuse partial results.

Algorithm 2 Decentralized certification process

```
1:  function HANDLE REQUEST(policy: $\mathbf{p}(\mathbf{b},\mathbf{t})$)
2:      $\mathbf{V} = \emptyset$
3:      for all $\mathbf{v} \in$ cached_certificates() do
4:          for all rule $r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})$ do
5:              if $r(\mathbf{b},\mathbf{t}) \preceq_p \mathbf{v}$ then
6:                  $\mathbf{V} = GLB(\mathbf{V},\mathbf{v})$
7:      if $\mathbf{p}(\mathbf{b},\mathbf{t}) \preceq_p \mathbf{V}$ then
8:          return $\mathbf{V}$
9:      for all rule $r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})$ do
10:         cert_names = query_certs(r($\mathbf{b},\mathbf{t}$))
11:         certs = collect_certs(cert_names)
12:         for all $\mathbf{v} \in$ certs do
13:             $\mathbf{V} = GLB(\mathbf{V},\mathbf{v})$
14:     if $\mathbf{p}(\mathbf{b},\mathbf{t}) \preceq_p \mathbf{V}$ then
15:         return $\mathbf{V}$
16:     for all rule $r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})$ do
17:         if $r(\mathbf{b},\mathbf{t}) \not\preceq_p \mathbf{V}$ then
18:             for all rule $r \in \mathbf{p}'$ do
19:                 for all metric evaluation $m(\mathbf{b}',\mathbf{t}') \in r(\mathbf{b},\mathbf{t})$
    do
20:                     m_res[$m$] = $m(\mathbf{b}',\mathbf{t}')$
21:     p_valid = $\bigwedge_{r(\mathbf{b},\mathbf{t}) \in \mathbf{p}(\mathbf{b},\mathbf{t})} r(\mathbf{b},\mathbf{t})$
22:     if p_valid then
23:         cert = new_certificate($\mathbf{p}(\mathbf{b},\mathbf{t})$)
24:         $\mathbf{V} = \{$cert$\}$
25:         for all $b \in \mathbf{b}$ do
26:             notify_new_certificate($b$, cert)
27:     else
28:         $\mathbf{V} = \emptyset$
29:     return $\mathbf{V}$
30:
31: function REQUEST VERIFICATION(policy : $\mathbf{p}(\mathbf{b},\mathbf{t})$)
32:     res = send_request(policy)
33:     certs = collect_certs(res)
```

Example VI.1. Consider the case of a policy that checks whether a rule r is valid for the interval $\mathbf{t}$ for the nodes $\mathbf{n}$. The CA queries the nodes in $\mathbf{n}$ and retrieves a certificate that validates r for all nodes in n' where $\mathbf{n} \subseteq \mathbf{n}'$ for the interval $\mathbf{t}'$ with $t \subseteq t'$. It follows that, if CA trusts the certificate results, the rule r has already been verified and thus also the policy.

Example VI.2. Consider the case of a policy that checks a rule r is valid for the interval $\mathbf{t} = \langle t_1, t_2 \rangle$ for the nodes $\mathbf{n}$. The CA queries the nodes in $\mathbf{n}$ and retrieves a certificate that validates r for all nodes in $\mathbf{n}$ for the interval $\mathbf{t}' = \langle t'_1, t'_2 \rangle$ with $t'_1 \in [t_1, t_2]$. It follows that, if CA trusts the certificate results, it needs to verify r only for the interval $[t'_1, t_2]$. A similar scenario considers a rule already verified on a subset of the nodes; in this case the CA could trust the certificate and verify only the unchecked nodes.

We note that the decentralized certification process proposes a collaborative approach designed to exploit the caching capabilities of Information-Centric Networks and improve the overall system performance. The decentralized approach outperforms the performance of the standard centralized process based on a certification authority managing the entire certification activities, also improving its security and addressing the problem of a single point of failure. The centralized approach has the main benefit of being inline with current certification frameworks. For this reason, our experiments in Section VII focus on the decentralized certification process only.

VII. Experimental Evaluation

We experimentally evaluated our certification methodology in a simulated ICN for non-functional properties CS availability, host availability, and network availability. To evaluate the performance of our certification process, we first defined the certification policies for the target properties (Section VII-A); we then evaluated the performance of the policy verification process (Section VII-B) and the network bandwidth consumed by the entire certification process execution (Section VII-C). We executed our experiments using the *Criterion* framework for the *Rust* programming language, repeating all tests at least 100 times and until the confidence on the measure is higher than 95%. All tests have been run on Linux with kernel 5.10.78 using an AMD 5900x processor and 32GB of RAM.

A. Certification Policies

We defined a set of policies modeling non-functional properties CS availability, host availability, and network availability in a ICN network node as follows.

CS optimality. It verifies the correct configuration of the CS and its operational status. It contains the following rules:

- *CSMemoryusageUB* checks whether the CS is lower than 60% to prevent memory starvation;
- *CSPolicy* checks whether the CS selected policy is Least Recently Used (LRU);
- *CSUsageLB* checks whether the CS has at at least 1% utilization to verify that it is enabled and operational.

Execution optimality. It verifies whether the network node has enough resources to operate correctly and avoid starvation. It contains the following rules:

- *FreeMemory* verifies whether the node has at least 200 MB of memory as a minimal system requirement;
- *NodeLoadUB* checks whether the node CPU delayed load is higher than 90% as an indicator of over-utilization.

Regular network traffic. It analyzes the recent network traffic looking for anomalies regarding the packet size and name components. It contains the following rules:

- *PITDataMinSizeLB* defines a lower bound of 10B to the forwarded data packets to detect possible pollution attacks attempts;
- *PITInterestMinComponentsLB* and *PITInterestMinComponentsUB* identify a range of valid values between 3 and 12 for the average number of name components in the forwarded interest packets;
- *PITDataAvgComponentsLB* and *PITDataAvgComponentsUB* define a range of valid values between 3 and 12 for the average number of name components in the forwarded data packets;
- *PITPendingInterestUB* sets an upper bound of 100 pending interest packets stored in the Pending Interest Table (PIT);
- *PITInterestMinSizeLB* defines a lower bound of 5B to the minimum size of the forwarded interest packets.

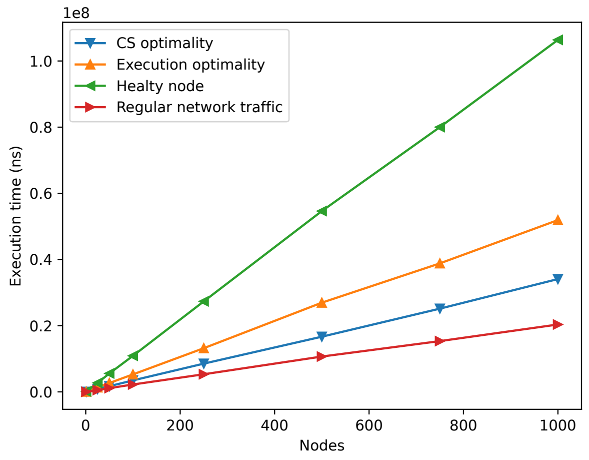


Figure 8: Execution time for the 4 policies varying the number of verified nodes.

Healthy node. It combines using a Boolean AND the three previous policies.

B. Policy Verification Process Performance

Given our implementation of the policy verification process, its asymptotic complexity is estimated as $\mathcal{O}(n \cdot r)$ with n being the number of nodes and r the number of rules included in the policy. We empirically verified this behavior by measuring its execution time varying the complexity of the target policy and the number of target network nodes. To exclude any delay or interference due to network interactions, we simulated local executions only, providing pre-cached measurements for each network node.

Figure 8 shows the execution time varying the number of target network nodes from one to 1000 in the 4 policies . The execution time grew linearly with the number of nodes for all the policies. Policy healthy node has lowest performance being a combination of the other three policies. Difference between policies CS optimality, execution optimality, regular network traffic depend on the performance of the specific rules composing them.

We then measured the impact of the policy complexity on the execution time in terms of rules by comparing the results obtained with a fixed number of nodes and a given time interval. We repeated the tests comparing the execution time of the 4 policies using 500 target nodes. The average execution time for the 4 policies are 16 ms for *CS optimality*, 27 ms for *Execution optimality*, 11 ms for *Regular network traffic*. and 54 ms for *Healthy node*. Our results show how policy Healthy node has an evaluation time close to the sum of the execution time of the single policies. We observe that the growth is linear with the number of evaluated rules, as expected.

Although the asymptotic complexity seems to be reflected in our experiments, it refers to the worst case, where *i)* the measurements cannot be shared between one or multiple rules, *ii)* no caching of the previously evaluated policies is allowed. If we consider sharing and caching, the asymptotic complexity is reduced to a logarithmic growth. Considering the caching abilities of both the CA and the network, a more fair asymptotic complexity estimation is $\mathcal{O}(log(n) \cdot log(r))$.[2]

C. Network Usage

We evaluated the network bandwidth used by the services during the certification process. The maximum size of each network packet is generally limited by the ICN protocol, 4KB packets in NDN. We then used an upper bound to the number of network requests to estimate the total traffic. We expect the total number of requests to complete a single certification to be lower than $\mathcal{O}(n \cdot r \cdot m)$, where n is the number of network nodes, r is the number of rules in the policy, and m is the number of metrics. Assuming the total number of metrics is limited and lower than the number of rules, with rules reusing the same measurements, we can simplify the previous asymptotic complexity to $\mathcal{O}(n \cdot m)$.

Figure 9 shows the relationship between the number of evaluations and the number of metrics evaluation requests sent during a certification process in the 4 policies in Section III-C. Our results show a linear correlation between evaluations and requests. The evaluation to requests ratio is specific to the policy definition and varies from 2:1 ratio, with policy *Execution optimality*, up to 10:1 ratio, with policy *Healty node*.

The policy verification service in the decentralized certification process produces additional traffic in the form of policy verification requests, policy query requests, and certificates retrieval. While in the first two cases their number is constant, the number of certificates retrieval requests grows linearly with the number of nodes and rules. A single certificate contains information about multiple nodes and multiple rules, therefore the actual ratio follows a logarithmic growth.

Again, this experiment had been carried out in the worst case scenario, where none of the requests are locally cached in the CA. When caching at both CA and ICN network nodes are considered, asymptotic complexity is reduced to $\mathcal{O}(log(n) \cdot log(m))$. In a more efficient implementation, measurements and rule evaluations can be retrieved once and shared by multiple rules, resulting in a drastic decrease in the total number of requests. We note that thanks to the ICN network caching capabilities, these request are more likely to be resolved by the caches of one of the network nodes in the request path before reaching their target node.

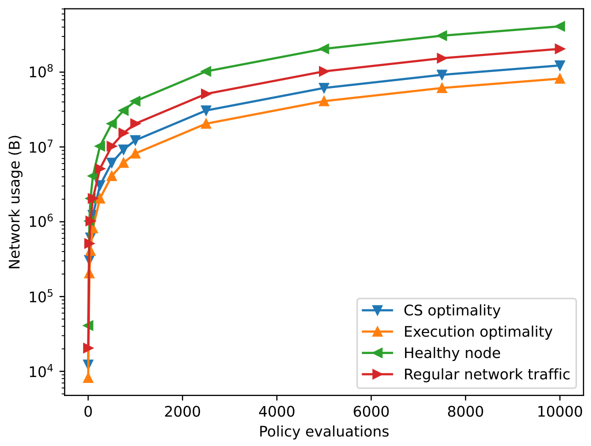


Figure 9: Number of metrics requests per policy evaluation.

VIII. Discussion

The certification methodology in this paper is fully compatible with ICN and does not require changes at protocol level. It can also substantially improve ICN functionalities in different scenarios that are summarized in the following of this section.

A. Network Adaptation

Modern networks have strong flexibility requirements, especially in mobile contexts, with devices entering and exiting the network, or even moving inside it, large spikes of traffic, and an ever increasing variety of network services. Effective adaptation is a hard problem, especially in large scale networks like national Internet Service Provider (ISP) networks, where the number of connected devices easily exceeds millions. The decentralized certification process in Section VI allows clusters of devices to self-regulate based on inferred network properties, while maintaining high levels of trust and privacy.

As an example, let us consider a scenario in which the network is capable of detecting a malfunctioning or compromised node, requiring that all the traffic is routed to an alternative path. This approach is viable for large monitored network nodes, like ISPs, cloud centers, and large firms ingress points, where teams of experts are available and the computational power is not a limiting factor. On the other hand, small scale networks, like offices, districts switches, hospitals, and small companies likely cannot afford a dedicated team. A network adaptation solution based on our distributed certification process permits the definition of automatic security measures to respond to the emergency, while requiring far less resources and expertise.

B. Secure Service Deployment

The deployment of a service in a set of network of nodes (e.g., in a public cloud, a multi-tenant environment, a cluster of servers) raises security concerns. Which properties can the host guarantee? What level of security can we expect from the nodes? Would the nodes have the necessary resources to run the service?

A certification process permits to verify policies and identify whether a certain set of nodes is suitable for the deployment of the chosen services. Our certification process can both evaluate

[2]Additional improvements can be obtained by parallelizing the execution of metric and rule evaluation, first inspecting their dependencies.

if a given set of nodes is suitable for the deployment of the chosen services and filter from an arbitrary large set of nodes the most suitable ones. This can be achieved by first using a policy-based filtering over the whole network and then a metric-based ordering on the remaining nodes. We note that this approach can be integrated in service deployment schedulers, to improve their effectiveness and enforcing resource or security constraints.

C. Attack Detection

Our certification process can be used to continuously monitor a target network with the goal of identifying misbehavior instead of certifying specific non-functional properties. This can be easily obtained with ad hoc policies focused on anomaly detection. An example of policy that can be used for anomaly detection policies is the one focused on detection of cache pollution attacks, a Denial Of Service (DOS) technique used to incapacitate the cache of an ICN network nodes by forcing them to store useless or unpopular data. Rules that analyse the network traffic statistics can rapidly identify changes in the type of contents stored across the network and alert of a possible attack event.

IX. Related Work

Certification methodologies have been successfully applied in many context including software and services. Anisetti et al. [20] proposed a formal certification scheme to validate non-functional properties of cloud-based services. Ardagna et al. [28] described a lightweight certification methodology for cloud environments, supported by continuous monitoring of infrastructures, platforms and services. Stephanow at al. [22] described a test-based certificate solution to identify whether a cloud service provider assured quality levels match the real measurements to prevent fraudulent and opportunistic behaviours. Felici at al. [23] proposed a multi-layer security certification scheme based on testing and monitoring probes. The notion of certification have been rarely applied in the past to verify networking protocols and nodes. Wu at al. [29] and Bossert at al. [30] applied certification to generic network security evaluation. They based their paper on Common Criteria certification model which has severe limits in dynamic environment. Network monitoring, is one of the prominent way to keep control of the networking traffic and behaviour and can be used for obtaining evidence for Certification. Monitoring in ICN networks has been extensively covered in literature, with particular emphasis on security of the network. In [14], [31] the authors proposed a monitoring plane for NDN with the goal of identifying network traffic anomalies and prevent content poisoning attacks. Another interesting solution is the one proposed by Van Adrichem at al. [32], which presented an implementation of an SDN layer for monitoring and traffic shaping in NDN. More recently, research has focused evaluating both networking nodes and protocols [33], [34]. Zhou et al. [33] presented a network-behavior monitoring schema aimed to identify congestions. Bialas et al. [34] presented a monitoring technique focused on anomaly detection.

Even if monitoring of ICN and trust in general is receiving an increasing attention by researchers, the certification in ICN networks still a quite unexplored topic. In this paper we extend our previous work in [27] that from the best of our knowledge constitute the first attempt to apply certification framework for ICN nodes using a rule-based schema.

X. Conclusions

While current literature has already demonstrated the effectiveness of ICN networks in large and complex scenarios, it does not include any unified solutions for monitoring and certification of such networks. In this paper we presented a certification methodology for such networks, capable of efficiently verifying complex policies to ensure the expected levels of QoS. We defined an abstract certification model that can be easily adapted for a large variety of applications, from Service Level Agreements monitoring to attack detection systems. We experimentally evaluated the performance of an implementation of our certification service confirming the feasibility of our methodology. We believe this paper is an important step in the evolution and diffusion of ICN based services in the field of edge and cloud computing, as a more efficient and trustworthy solution.

References

[1] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, and J. Cao, "Named data networking: A survey," *Computer Science Review*, vol. 19, pp. 15–55, 2016.

[2] A. Afanasyev, J. Burke, T. Refaei, L. Wang, B. Zhang, and L. Zhang, "A brief introduction to named data networking," in *Proc. of MILCOM*. Los Angeles, CA: IEEE, 2018, pp. 1–6.

[3] H. Khelifi, S. Luo, B. Nour, H. Moungla, Y. Faheem, R. Hussain, and A. Ksentini, "Named data networking in vehicular ad hoc networks: State-of-the-art and challenges," *IEEE Communications Surveys Tutorials*, vol. 22, no. 1, pp. 320–351, 2020.

[4] Z. Li, Y. Liu, Y. Chen, Y. Xu, and K. Liu, "Performance analysis of a novel 5g architecture via content-centric networking," *Physical Communication*, vol. 25, pp. 328–331, 2017.

[5] W. Shang, A. Bannis, T. Liang, Z. Wang, Y. Yu, A. Afanasyev, J. Thompson, J. Burke, B. Zhang, and L. Zhang, "Named data networking of things (invited paper)," in *Proc. of IEEE IoTDI*, Berlin, Germany, 2016, pp. 117–128.

[6] S. Lederer, C. Mueller, C. Timmerer, and H. Hellwagner, "Adaptive multimedia streaming in information-centric networks," *IEEE Network*, vol. 28, no. 6, pp. 91–96, 2014.

[7] C. Tsilopoulos and G. Xylomenos, "Supporting diverse traffic types in information centric networks," in *Proc. of ACM SIGCOMM workshop on Information-centric networking*, ser. ICN '11. New York, NY, USA: ACM, 2011, pp. 13–18.

[8] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang, "Interest flooding attack and countermeasures in named data networking," in *Proc. of IFIP Networking Conference*, Brooklyn, NY, USA, 2013, pp. 1–9.

[9] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in named data networking," in *Proc. of 22nd ICCCN*, Nassau, Bahamas, 2013, pp. 1–7.

[10] L. Yao, Y. Zeng, X. Wang, A. Chen, and G. Wu, "Detection and defense of cache pollution based on popularity prediction in named data networking," *IEEE TDSC*, pp. 1–1, 2020.

[11] H. Salah, M. Alfatafta, S. SayedAhmed, and T. Strufe, "CoMon++: Preventing cache pollution in NDN efficiently and effectively," in *Proc. of 42nd IEEE LCN*. Singapore: IEEE, 2017, pp. 43–51.

[12] Amin Karami and Manel Guerrero-Zapata, "An ANFIS-based cache replacement method for mitigating cache pollution attacks in named data networking," *Computer Networks*, vol. 80, pp. 51–65, 2015.

[13] M. Conti, P. Gasti, and M. Teoli, "A lightweight mechanism for detection of cache pollution attacks in named data networking," *Computer Networks*, vol. 57, no. 16, pp. 3178–3191, 2013.

[14] T. Nguyen, H. Mai, G. Doyen, R. Cogranne, W. Mallouli, E. M. d. Oca, and O. Festor, "A security monitoring plane for named data networking deployment," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 88–94, 2018.

[15] P. Tammana, R. Agarwal, and M. Lee, "Distributed network monitoring and debugging with SwitchPointer," in *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*. Renton, WA: USENIX Association, Apr. 2018, pp. 453–456.

[16] P.-W. Tsai, C.-W. Tsai, C.-W. Hsu, and C.-S. Yang, "Network Monitoring in Software-Defined Networking: A Review," *IEEE Systems Journal*, vol. 12, no. 4, pp. 3958–3969, Dec. 2018.

[17] T. Nguyen, X. Marchal, G. Doyen, T. Cholez, and R. Cogranne, "Content poisoning in named data networking: Comprehensive characterization of real deployment," in *Proc. of IFIP/IEEE IM*, Lisbon, Portugal, 2017, pp. 72–80.

[18] S. Lee, K. Levanti, and H. S. Kim, "Network monitoring: Present and future," *Computer Networks*, vol. 65, pp. 84–98, Jun. 2014.

[19] H. C. A. van Tilborg and S. Jajodia, Eds., *ISO 15408 CC – Common Criteria*. Boston, MA: Springer US, 2011, pp. 648–648.

[20] M. Anisetti, C. A. Ardagna, E. Damiani, and F. Gaudenzi, "A semi-automatic and trustworthy scheme for continuous cloud service certification," *IEEE TSC*, vol. 13, no. 1, pp. 30–43, 2020.

[21] M. Anisetti, C. Ardagna, E. Damiani, and G. Polegri, "Test-Based Security Certification of Composite Services," *ACM Transactions on the Web*, vol. 13, no. 1, pp. 3:1–3:43, Dec. 2018.

[22] P. Stephanow, G. Srivastava, and J. Schutte, "Test-Based Cloud Service Certification of Opportunistic Providers," in *Proc. of 9th IEEE CLOUD*. San Francisco, CA, USA: IEEE, Jun. 2016, pp. 843–848.

[23] M. Egea, K. Mahbub, G. Spanoudakis, and M. R. Vieira, "A Certification Framework for Cloud Security Properties: The Monitoring Path," in *Accountability and Security in the Cloud*, M. Felici and C. Fernández-Gago, Eds. Cham: Springer International Publishing, 2015, vol. 8937, pp. 63–77, series Title: Lecture Notes in Computer Science.

[24] E. G. AbdAllah, H. S. Hassanein, and M. Zulkernine, "A Survey of Security Attacks in Information-Centric Networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1441–1454, 2015.

[25] T. Liang, J. Pan, M. A. Rahman, J. Shi, D. Pesavento, A. Afanasyev, and B. Zhang, "Enabling named data networking forwarder to work out-of-the-box at edge networks," in *Proc. of IEEE ICC Workshops*, Dublin, Ireland, 2020, pp. 1–6.

[26] M. Hussaini, S. A. Nor, H. Bello-Salau, H. J. Hadi, A. A. Gumel, and K. A. Jahun, "Mobility support challenges for the integration of 5g and IoT in named data networking," in *Proc. of 2nd IEEE NigeriaComputConf*, Zaria, Nigeria, 2019, pp. 1–7.

[27] Marco Anisetti, Claudio A. Ardagna, Filippo Berto, and Ernesto Damiani, "Security Certification Scheme for Content-Centric Networks," in *Proc. of the 17th IEEE SCC*, Chicago, IL, USA, September 2021, p. 10.

[28] C. A. Ardagna, R. Asal, E. Damiani, T. Dimitrakos, N. El Ioini, and C. Pahl, "Certification-Based Cloud Adaptation," *IEEE TSC*, pp. 1–1, 2018.

[29] X.-H. Wu, J.-P. Li, and W. Yao, "A network security evaluation model based on common criteria," in *Proc. of IEEE ICACIA*. IEEE, 2008, pp. 416–420.

[30] G. Bossert and F. Guihery, "Security evaluation of communication protocols in common criteria," in *Proc. of IEEE ICC*, 2012.

[31] H. L. Mai, T. Nguyen, G. Doyen, R. Cogranne, W. Mallouli, E. M. de Oca, and O. Festor, "Towards a security monitoring plane for named data networking and its application against content poisoning attack," in *Proc. of IEEE/IFIP NOMS*, Taipei, Taiwan, 2018, pp. 1–9.

[32] N. L. M. van Adrichem and F. A. Kuipers, "NDNFlow: Software-defined Named Data Networking," in *Proc. of the 1st IEEE NetSoft*, Apr. 2015, pp. 1–5.

[33] Y. Zhou, J. Bi, T. Yang, K. Gao, J. Cao, D. Zhang, Y. Wang, and C. Zhang, "HyperSight: Towards scalable, high-coverage, and dynamic network monitoring queries," *IEEE J-SAC*, vol. 38, no. 6, pp. 1147–1160, 2020.

[34] A. Bialas, M. Michalak, and B. Flisiuk, "Anomaly detection in network traffic security assurance," in *Proc. of DepCoS-RELCOMEX*, ser. Advances in Intelligent Systems and Computing, W. Zamojski, J. Mazurkiewicz, J. Sugier, T. Walkowiak, and J. Kacprzyk, Eds. Cham: Springer International Publishing, 2020, pp. 46–56.

Marco Anisetti is an Associate Professor at the Università degli Studi di Milano. His research interests are in the area of computational intelligence, and its application to the design and evaluation of complex systems. He has been investigating innovative solutions in the area of cloud security assurance evaluation. In this area he defined a new scheme for continuous and incremental cloud security certification, based on distributed assurance evaluation architecture.

Claudio A. Ardagna is Full Professor at the Università degli Studi di Milano, the Director of the CINI National Lab on Big Data, and co-founder of Moon Cloud srl. His research interests are in the area of cloud-edge security and assurance, and data science. He has published more than 140 contributions in international journals, conference/workshop proceedings, and chapters in international books. He has been visiting researcher at BUTP, Khalifa University, GMU.

Filippo Berto is a Ph.D. student at the Università degli Studi di Milano. His research interests are in the area of cybersecurity, edge coputing and distributed systems. His current research field are security assurance, 5G and cloud-edge networks, and Named Data Networking, focusing on networks and services certification techniques.

Ernesto Damiani is a Full Professor at Università degli Studi di Milano , Italy, Senior Director of the Robotics and Intelligent Systems Institute, and Director of Center for Cyber Physical Systems (C2PS) within Khalifa University (UAE). He is President of the Consortium of Italian Computer Science Universities (CINI). Ernesto has been a recipient of the Research and Innovation Award from the IEEE Technical Committee on Homeland Security, of the Stephen Yau Award from the Service Society, of the Outstanding contributions Award from IFIP TC2.