On the Interplay among Naming, Content Validity and Caching in Information Centric Networks

Andrea Detti, Alberto Caponi, Giuseppe Tropea, Giuseppe Bianchi, Nicola Blefari-Melazzi
CNIT - Department of Electronic Engineering, University of Rome "Tor Vergata"
Via del Politecnico 1, Rome (Italy)
andrea.detti@uniroma2.it

***Abstract*—Information Centric Networking (ICN) is paradigm in which the network layer provides users with content addressed "by name". In-network caching is one of the key functionality to be provided by ICN nodes. To avoid network nodes caching fake contents, it is necessary to verify the validity of data items. A content is deemed to be valid if it verifies three properties: i) integrity: it has not be modified; ii) provenance: it comes from the intended source; iii) relevance: it is indeed the content requested from the user (by using the name of that content). In this paper, we discuss the interplay among three pivotal ICN aspects: caching, validity and naming. Specifically, we will investigate different naming and digital signature schemes, evaluating their speed, overhead and their impact on caching performance. Perhaps counter-intuitively, we find that the relatively slow verification time of today's signatures, which bottlenecks the rate of storing new data items in the network caches, does not come as a critical shortcoming, but may actually even improve the cache hit probability when the LRU caching policy is employed.**

Keywords: information centric networking; caching; digital signature; naming schemes; performance evaluation

I. INTRODUCTION

Many researchers are working on a possible paradigm shift: from the traditional reliance on point-to-point communication, to the emergence of *Information Centric Networking* (ICN) primitives [1][2][3][4], releasing the ties which have historically bound content retrieval operations to the actual location in which content is stored. The characteristic of ICN is the ability to deliver a *named* block of data, irrespective of where (i.e., on which specific server) such data block is actually stored. ICN ideas are not new, dating back to at least year 2000 [5]; however, research on ICN has lately seen a significantly renewed interest since the specification of the CCN (Content Centric Networking) architecture [1]. Since then, several ICN proposals, including, but not nearly limiting to SAIL [6], CONVERGENCE [7], GreenICN [8], OFELIA [10] have specified technical approaches and solutions for addressing content by using names that do not include references to their location, routing a user request for a content name towards the "closest" copy of the content, delivering the content back to the requesting host, etc.

As discussed in [9], even if differing in notable technical details, most ICN proposals share significant similarities. Accordingly, we believe that the discussion on the interplay between naming, content validity and caching carried out in this paper may apply to different ICN proposals, besides their specifics. Nevertheless, for the purpose of concreteness, we base our quantitative investigation on the ICN model detailed in [1], where:

- content is identified through unique names (strings);
- content is segmented into chunks;
- each chunk has a unique identifier, which includes the content-name as prefix;
- to fetch content, a client sequentially downloads all the component chunks [12];
- network nodes *route-by-name* requests for chunks on the shortest-path and by using a name-based routing table [11] [13];
- network nodes provide an *en-route caching* service [14];
- the latter two processes take place *at line rate* [26].

The specific contribution of this paper is twofold. First, we extend the analyses, carried out in [15][16], on the impact of different content naming schemes on the security properties of the information-centric network, by more closely discussing pros and cons of possible combination of human-readable or self-certifying names with traditional or identity-based signature schemes. Second, and perhaps more significantly, we evaluate the impact of signature processing on the ICN caching performance. Our major finding, which at a first glance may appear counter-intuitive, is that *the limited speed of practical signature verification algorithms does not seem to be a critical problem for an ICN* even if it is far from permitting content caching at line rate (a substantial fraction of content cannot be verified in time for caching). On the contrary, we show that caching performance, in terms of cache hit probability, may even *increase* when the LRU caching policy is employed.

II. NAMING AND CONTENT VALIDITY

A. Possible combinations of naming schemes and signature approaches

It is well known [15][16] that the choice of the naming scheme in an ICN has implications on security. Tab. 1 classifies naming schemes along two main independent axes: impact on routing plane and impact on security properties. Along the first dimension we have names that can be hierarchical or flat; along the second dimension we have names that can be self-certifying or human-readable.

The work for this paper was performed in the context of the FP7/NICT EU-JAPAN GreenICN project.

Tab. 1 –Naming schemes

Routing / *Security*	**Flat**	**Hierarchical**
Human-readable	Foo.com.video1.mp3	Foo.com/video1.mp3
Self-certifying	0x3fb889fffa	0x65de3/video1.mp3

Tab. 2 –Data packaging model

Data ***U***nit	***C***ontent	***N***ame = P/L
		Data
	Verification block	***S***ignature
		Additional ***INFO***

The decision whether to use flat versus hierarchical names mainly impacts the scalability of the ICN routing plane. This issue is one of the most important challenges in the ICN community [9], and deserves a dedicated paper. However, it is noteworthy that we are considering an ICN that routes requests at line-rate using name-based routing-table. In this *specific* scenario, using flat-names implies a routing table with an entry per object. Considering that the current number of (Google) indexed Web objects is in the order of $5 \cdot 10^{10}$, the resulting routing table would be too large to be accommodated within a current fast memory chip, e.g. TCAM or SRAM [11], thus preventing the practical deployment of the ICN. For this reason, we assume to use hierarchical names, since they "help scalability by reducing the size and update-rate of the routing tables" [15]. We consider a rather general hierarchical naming scheme, whose form is Principal/Label (*P*/*L*) [4]. The name is formed by a sequence of component strings, separated by a reserved character, e.g. "/". The first component *P* is an identifier of the Principal (term with same meaning as used in [15], or as Publisher in [16]) of the resource (e.g. Foo.com). Contents (published) by the same principal have the common prefix *P* and are differentiated from each other by the sequence of components that follows, which form the so-called Label (*L*). Furthermore, at the chunk level, the label *L* also includes protocol components used to identify the specific chunks (e.g. Foo.com/video1.mp3/chunk1).

Since content chunks are delivered to users not only from trusted origin servers but also from distributed caches, users should be able to verify the validity of received data by means of "security" information included in the data item [1]. Moreover, as pointed out in [9] and [29], not only end users but also caching nodes shall verify in-transit content chunks, so as to avoid the cache to be *poisoned* with fake contents. A content chunk is considered valid if it verifies the following criteria:

- **Integrity:** received content has not been modified, i.e. it is the originally published one
- **Provenance:** source of the content is authentic, i.e. the data is provided by the principal *P*
- **Relevance:** received content is really the content requested by the user

Assuming that each publisher of data, who is the principal of corresponding content, has a real-world identity and a public/private key pair she uses to sign published contents, the above requirements can be satisfied using digital signatures[1]. Different digital signature schemes exist and their viability depends both on the properties of the name structure itself (e.g., human readability) and on performance aspects (e.g. speed of verification, bandwidth overhead, etc.).

To compare the different possibilities, Tab. 2 sketches a general data unit format, including signature-related information. Every chunk of data *D* is packaged together with a header field *N*, which is the chunk name in the form P/L. The principal, by means of her private key, digitally signs the resulting package $C=\{N, D\}$. The signature *S*, plus any other information (*INFO*) needed by the network nodes to perform *on-line* verification of *C* against *S*, is included at the tail, in verification block *V*. Each network data unit *U* is then $\{C, V\}$, where $C=\{N, D\}$ and $V=\{S, INFO\}$.

Since the name *N* is part of the digitally signed material *C*, *relevance* of the data unit is always automatically guaranteed, because it is not possible to change the name of the data item without failing the digital verification. We now explore trade-offs coming from three different combinations of name structure and signature algorithms, when verifying *integrity* and *provenance*.

1) Human-readable names & traditional signature - the principal identifier *P* is a human-readable string, e.g. "Foo.com", and the signature is a traditional one, such as the topmost employed RSA [20] or ECDSA [19] algorithms. The Additional *INFO* field (Tab. 2) contains the Digital Certificate of the principal, to enable nodes to verify chunks on-the-fly, i.e. without requiring the temporary download of the certificate from a remote storing node. Digital certificate includes the public key of the principal, properly signed by a Certification Authority (CA). Verification of the signature *S* and of the certificate ensures integrity. To verify provenance, we want to be sure that the principal identified by *P* in the name *N*, really is who has generated the content. This is done i) by reading through the certificate, which connects *P* with its public key ii) and by the verification of the signature. It is worth observing that, in this case, secure operation of in-network caching requires the presence of a CA infrastructure, whose public keys shall be preloaded on network nodes.

2) Human-readable names & ID-based signature - IDentity-based signature uses public/private key pairs, too. However, in this case, the public key can directly be any string, e.g. the principal identifier *P* itself [18]. Private key is released to the user by a Private Key Generator (PKG) infrastructure. The signature algorithm is usually based on the discrete logarithm problem and elliptic curves [17]. The Additional *INFO* field (see Tab. 2) includes an identifier of the PKG, by means of which the verifier looks up configuration parameters, specific of the chosen PKG, needed to verify the signature. Those parameters are fetched from a preloaded set on network

[1] Here we don't deal with Confidentiality (content encryption) that remains an end-to-end service between publisher and end-user.

nodes. Verification of the signature *S* using PKG parameters ensures integrity. In this case, to verify provenance, we do not need a certificate connecting *P* with its public key, because the public key itself is the principal identifier *P*. It is worth observing that secure operation of in-network caching requires the presence of a PKG infrastructure that, however, could also be distributed [27].

3) Self-certifying names & traditional signature – a self-generated public key is used as the principal identifier *P*. Signature algorithms may be RSA or ECDSA, so the principal identifier *P* (being equal to the key) is not human-readable. The Additional *INFO* field (see Tab. 2) is void. Integrity is assured by verifying the signature *S*. Just like the case of identity-based signature, provenance is verified by simply checking that the public key used for the signature is the principal identifier *P*. Differently from the other cases, secure operation of in-network caching stands up without the need of any external (to the network) CA or PKG infrastructure. As a drawback, in case of self-certifying names, users likely need trusted translation services (DNS, Google-like, personal address books, etc.), to go from keywords/descriptors to a principal identifier *P*. Instead, in case of human-readable names, users may exploit mnemonic properties of the principal identifier *P* by directly giving it to the network layer to access its data.

B. Overhead and verification time of the possible combinations

In this section we compare the discussed combinations of naming schemes and signature algorithms with respect to the protocol overhead and with respect to the signature verification speed. Configuration parameters that provide the same level of security are used for the comparison. Specifically, for RSA signature we use keys of 1024 bits; for ECDSA signature, we use an elliptic curve over a 160bit prime field (NIST secp160k1 [22]); for ID-based signature we use the same elliptic curve and the approach proposed in [17], where the (slow) pairing computation is not required.

We evaluated verification speed using OpenSSL API and using an Intel I7 processor @1.8Ghz. We point out that, in case of combinations 1a and 1b, it is needed to verify the digital certificate, too. Accordingly, we assume that the verification of the certificate lasts a time equal to the verification of the signature *S*. We point out that this evaluation of speed has the focussed goal of qualitatively comparing the different combinations with respect to the verification performance, rather than to exactly evaluate speed performance of a signature scheme. Indeed, ICN routers will likely implement hardware-based integrity check, which makes the verification process faster. Such an improvement may be obtained by all of the schemes, so the qualitative comparison carried out in this section remains valid.

Tab. 3 reports sizes of verification block *V* in the different cases. Techniques based on elliptic curves (ECDSA and ID-based) provide lower bit overhead compared to RSA, but have quite slower verification performance. We observe that the overhead introduced by the combinations 1a, 1b and 3a is rather high, considering that an ICN chunk has a relatively small size, e.g. 4kB. This large overhead may thwart the expected ICN benefits of traffic reduction achieved by in-network caching, for instance.

Tab. 3 – Sizes of the verification block fields and verification time in case of Human Readable (H.R.) and Self-Certifying (S.C.) names, and RSA, ECDSA and Identity-Based (I.B.) signature schemes

Combination	Add. INFO	Sign. (bits)	Add. INFO (bits)	Verification time (ms)
1a- H.R. with RSA	Certificate	1024	2048	0.12
1b - H.R. with ECDSA	Certificate	320	408	0.58
2 - H.R. with I.B.	PKG id	506	8	0.33
3a - S.C. with RSA	None	1024	0	0.06
3b - S.C. with ECDSA	None	320	0	0.29

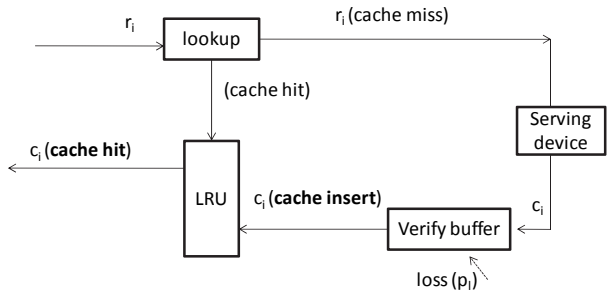

Fig. 1 – Caching functional model

Consequently, architectural choices seem to favour human-readable names with identity-based signature (2) or self-certifying names with ECDSA (3b). However, in the IDentity-based scheme, the network cannot operate as a self-standing entity, but requires the external, human-managed PKG infrastructure, and this may complicate network deployment.

III. CONTENT VALIDITY AND CACHING PERFORMANCE

To avoid denial of service due to cache poisoning [29] (i.e. injection of fake contents), ICN nodes shall check validity of each cached chunk [9].

At the same time, *asymmetric* key algorithms (e.g. RSA, ECDSA, ID-based), as of today, are deemed to be too complex to support line rate operations of backbone nodes, which usually are in the order of tens of Gbit/s. For instance, in case of 4kB chunks, using the hardware and software of test reported in Tab. 3, in cases (2) and (3b), we support signature checking at a rate of "only" 90 Mbps and 120 Mbps, respectively. As a consequence, the verification process may skip, due to the overload, many of the incoming chunks, especially in backbone ICN nodes. Non verified chunks do not enter the cache and this is why in this section we analyze the variation of cache hit probability, in case the stream of data items entering the cache is a random sampled version of the ingress stream at the node. We term such scenario as *lossy caching*, since we consider as *lost* an item which is not inserted in the cache.

To carry out the analysis, we use the caching functional model depicted in Fig. 1. When a request r_i for the i-th chunk reaches a node, the node looks-up for it in the cache. In case of a cache hit, the chunk c_i is retrieved from the cache and sent back to the user. In case of cache miss, the node forwards the request r_i in the upstream direction. When the chunk c_i comes back from an upstream serving device (the original server, an intermediate cache, etc.), the chunk is immediately forwarded to the user (figure does not show this process), meanwhile a copy of the chunk is inserted in the FIFO buffer of the signature verification process. If verification succeeds, c_i is inserted in the cache. Since the verify buffer is limited and the arrival rate could be greater than the verification rate, then the buffer performs as a lossy queue. Therefore, a request that suffers for a cache miss may, or may not, result in later insertion of the corresponding chunk in cache.

We consider a Least Recently Used (LRU) replacement strategy, e.g. implemented with the traditional *stack of references*. The stack contains references to the data items currently stored in the cache memory. Each time an item is retrieved from or inserted in the cache, its reference is moved on top of the LRU stack. In case of insertion events (and cache full), the insertion of a new item implies removal from the cache of the last item referenced by the stack, that is, exactly, the Least Recently Used item.

Concerning the storing space of cache, we assume that storage capacity *SC* is expressed as the number of data items that may be stored therein. This assumption is practical in case of in-network caching, where the cache capacity may be bounded by the size of the fast memory (e.g. SRAM) used to implement the lookup table, which indexes actual data stored in a larger slow memory (e.g. DRAM).

In what follows we first carry out the analysis on a single cache then we extend the analysis to a network of caches.

A. Single cache analysis

In order to evaluate the effect of the loss on a single cache, we implemented the model of Fig. 1 through a simple MATLAB simulator, in which we replace the verify buffer with a random (Bernoulli) dropper. We find out that i) considering a stream of requests that follows the Independent Reference Model (IRM) [23][24] and ii) considering the sequence of lost items as the result of a probabilistic sampling, the presence of a lossy verify buffer does not decrease the average cache hit probability of a LRU cache; conversely, losses may even lead to an *increase* of the average cache hit probability[2].

We remind that the independent reference model assumption provides that requests for data items occur in an infinite sequence, where the probability that the requested item has index i is proportional to the popularity law $q(i)$, e.g. the well-known Zipf. This simple model clearly does not account for the so-called *temporal locality*, i.e. the fact that requests of the same item may be grouped in time. However, as discussed in [28], in a hierarchy of caches, the leaf caches absorb a valuable part of temporal locality present in the stream of requests. So we argue that the temporal locality experienced in core nodes of ICN is rather limited, and these nodes are the ones where the loss mainly shows up.

Regarding the assumption of modeling the sequence of lost items as the result of a Bernoulli sampling process, we observe that it is well known from traditional (e.g., NetFlow) trace sampling, that under mild assumptions (large amount of flows, packets suitably interleaved, sampling rate small, see a rigorous analysis in [30]) deterministic sampling of data items is practically equivalent to probabilistic sampling of data items. So, from a performance point of view, a system in which deterministically only one item every T milliseconds is accepted by the verification process is close to a system where accepted items are randomly chosen, meanwhile maintaining the same acceptance probability.

Fig. 2 reports the average cache hit probability h_t as a function of the loss (p_l) and in case of a cache with a capacity equal to 1% of the data set size. We consider a data set formed by 10^5 items, whose popularity follows a Zipf distribution with slope α=0.75 [3]. Items are ranked according to their popularity, so that the lower the rank, the higher the popularity. We observe that by increasing the loss probability, the cache hit probability increases as well.

The reason behind this behaviour is that the loss yields a reduction of the number of requests that produce updates of cache status, since some of them are lost by the verify buffer. However, *the reduction is unequal among data items*. Un-popular items have a low cache hit probability, so a request for an unpopular item *often* faces the lossy verify buffer. Conversely, popular items have a high cache hit probability, so a request of a popular item *rarely* faces the lossy verify buffer. As a consequence, although all items experience reduction of their possibility of updating the cache, this reduction is most severe for un-popular items in this case, when contrasted to a lossless verify buffer case. With lossy verification, popular items remain in the cache for a longer time and they obtain a greater cache hit probability, leading to a *global* increase of the cache-hit probability.

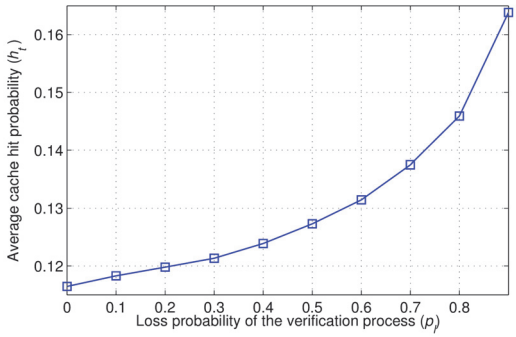

Fig. 2 – Average cache hit probability versus loss

[2] We note that other works in literature (e.g. [14]) use controlled probabilistic operations to improve caching performance. But in our case these operations are non controllable and are forced by exogenous aspects, i.e. the overflow of the verification process.

[3] We also considered cache capacities of 0.1% and 10%, and Zipf slopes of 0.6 and 1. In these cases too we obtained a behavior similar to Fig. 2

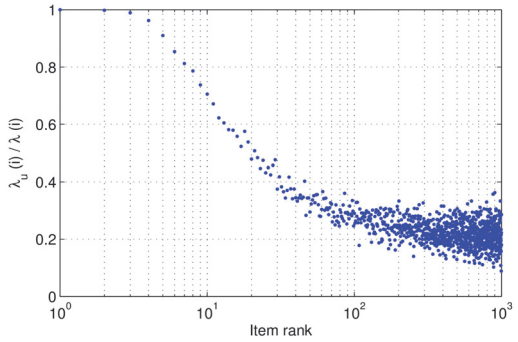

Fig. 3 – Ratio between cache update rate λ_u and the request rate versus the item rank in case of loss probability equal to 0.8

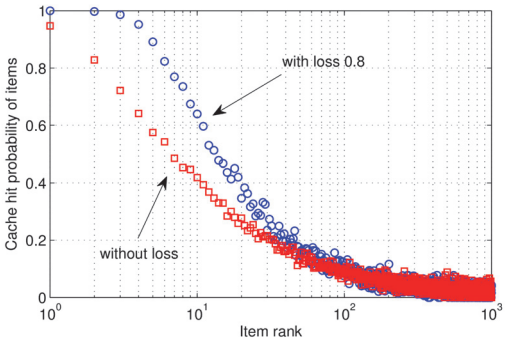

Fig. 4 – Cache hit probability of items $h(i)$ versus the item rank

To support this conclusion we observe that the events actually updating the status of the LRU cache are: *cache hit* and *cache insert* events (Fig. 1). In case of loss, update events are triggered only by a *subset* of requests. Conversely, without loss, every request leads to a cache update. Therefore the loss leads to a *reduction* of the cache update rate. Accordingly, let us define:

- $\lambda(i)$ as the *request rate* of the *i*-th item (chunk)
- $\lambda_u(i)$ as the *caching update events rate* of the *i*-th item, i.e. the aggregate rate of the events of cache hit and cache insert of the *i*-th item
- $q(i)$ as the *popularity* of the *i*-th item, i.e $q(i) = \lambda(i)/\sum_j \lambda(j)$
- $h(i)$ the cache hit probability of the *i*-th item
- h_t the average cache hit probability, i.e. $h_t = \sum_i q(i)\lambda(i)$

Using the model of Fig. 1, the relationship between the update rate $\lambda_u(i)$ and the request rate $\lambda(i)$ can be expressed as:

$$\lambda_u(i) = \lambda(i)(h(i) + (1 - h(i))(1 - p_l))$$

We observe that in presence of a loss probability p_l, the update rate $\lambda_u(i)$ is equal to the request rate $\lambda(i)$, multiplied by the *reducing* factor $(1-p_l)(1-h(i))$. As a consequence, an increase of p_l leads to reduction $\lambda_u(i)$ by a factor that is not equal among items, but increases with the index *i* of the item.

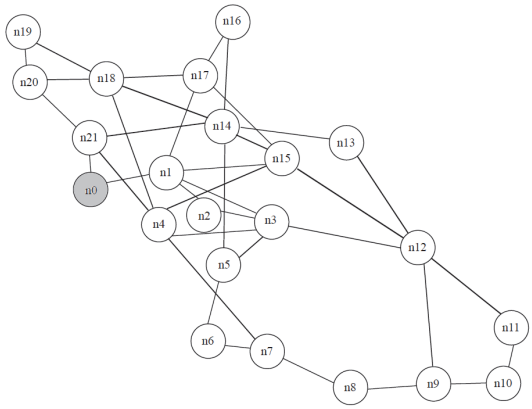

Fig. 5 – GEANT topology

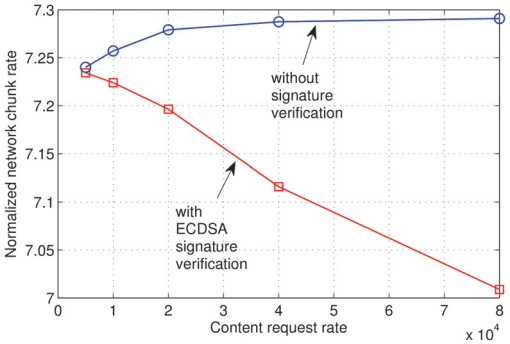

Fig. 6 – Normalized network chunk rate versus content request rate

In case of two items *i* and *j*, with *i*>*j*, the more popular item *j* has a cache hit probability $h(j)$ greater than $h(i)$. Thus, the multiplier factor $(1- p_l)(1-h(i))$ is smaller (i.e. worse) for *i* than for the more popular *j*. As proof of concept of the unequal reduction of the update rate, Fig. 3 reports the ratio $\lambda_u(i)/\lambda(i)$ for the different items, in case of $p_l = 0.8$. We see that the ratio $\lambda_u(i)/\lambda(i)$ decreases while reducing the popularity of the item, i.e. increasing the rank.

Fig. 4 shows the cache-hit probability of items $h(i)$ as a function of the item rank both in case of loss equal to 0.8 and in case of absence of loss. We note that the presence of loss promotes the cache-hit probability of popular items (low rank), and this is a consequence of the unequal update rate reduction shown in Fig. 3. We remind that, under the IRM assumption, the sum of the cache hit probability $h(i)$ is equal to the cache storage capacity *SC* [23][24]. Therefore, the loss shifts the "energy" *SC* of distribution $h(i)$ towards lower indexes, where the values of $q(i)$ are greater. In turn, the average cache hit probability $h_t = \sum_i q(i)h(i)$ increases.

B. Cache network analysis

In this section, we evaluate the impact of this lossy-caching within a network, by using the ccnSim simulator [21]. In both cases, the real popularity of items follows a Zipf distribution with slope α=0.75 but, differently from the single cache analysis, here we model a realistic signature verify buffer of

length 100; the signature verification time is 0.29 ms, i.e. the "self-certifying with ECDSA combination" (3b) of Tab. 3.

We consider the GEANT network shown in Fig. 5, formed by 22 nodes (see [25] for more detail). Node 0 is the repository of all contents. On average, content is segmented into 3 chunks (4kB per chunk). Each node has a LRU cache with a capacity equal to 1% of the whole set of available chunks. We generate arrival of content requests with an exponential negative distribution, and uniformly distribute the generated requests among nodes.

Fig. 6 shows the aggregate rate of chunks exchanged on all links of the network, normalized to the content request frequency, considering both the presence and the absence of signature check. We observe that, with signature check, the increase of the request rate leads to a small reduction of traffic; indeed, the increase of the request rate, increases the loss on the verify buffer, hence caching is more effective.

IV. CONCLUSIONS

In this paper, we have discussed different combinations of naming and signature schemes, and analysed the impact of signature check on cache-hit probability. Our most notable finding relates to the impact of the inability to have caching at line rate of each and every in transit item, due to the computation time of digital signatures verification. Our results show that, in case of cache loss due to the overload of the security processor of a node, cache hit performance of a LRU cache does not decrease, and could even increase. This shows that the speed of signature verification is not a significant criticality of an ICN.

It is worth to highlight that we derive this conclusion considering request streams without temporal locality. Instead, in [31] we derive that in presence of strong temporal locality, cache loss may reduce the cache-hit probability. As we discussed, the loss reduces the update rate of the cache, therefore the promptness of a LRU cache in adapting to the temporary change of popularity. However, valuable cache loss *practically* occurs in the network core, where the rate of forwarded traffic abundantly exceeds the rate of digital signature verification. And in a cache network temporal locality at the ingress of core caches is strongly reduced [28], due to the filtering effect of edge caches.

REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. Briggs, R. Braynard, "Networking named content", ACM CoNEXT 2009.

[2] A. Detti, N. Blefari Melazzi, S. Salsano, M. Pomposini, "CONET: A Content Centric Inter-Networking Architecture", ACM SIGCOMM Workshop on Information-Centric Networking, ICN 2011.

[3] D. Trossen, M. Sarela, K. Sollins: "Arguments for an information-centric internetworking architecture", SIGCOMM Computer Communication Review, vol. 40, pp. 26-33, 2010.

[4] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, I. Stoica, "A data-oriented (and beyond) network architecture", ACM SIGCOMM 2007.

[5] M. Gritter, D. R. Cheriton, "TRIAD: A New Next-Generation Internet Architecture", www-dsg.stanford.edu/triad/, July 2000.

[6] SAIL project website: www.sail-project.eu

[7] CONVERGENCE project website: www.ict-convergence.eu

[8] GreenICN project website: www.greenicn.org

[9] A. Ghodsi, T. Koponen, B. Raghavan, S. Shenker, A. Singla, J. Wilcox, "Information-Centric Networking: Seeing the Forest for the Trees", 10th ACM Workshop on Hot Topics in Networks, HotNets 2011.

[10] N. Blefari Melazzi, A. Detti, G. Morabito, S. Salsano, L. Veltri: "Supporting Information-Centric Functionality in Software Defined Networks", IEEE ICC 2012, Software Defined Networks Workshop.

[11] A. Detti, M. Pomposini, N. Blefari Melazzi, S. Salsano: "Supporting the Web with an Information Centric Network that Routes by Name", Elsevier Computer Networks, August 2012.

[12] S. Salsano, A. Detti, M. Cancellieri, M. Pomposini, N. Blefari-Melazzi, "Transport-layer issues in Information Centric Networks", ACM SIGCOMM Workshop on Information-Centric Networking, ICN 2012.

[13] N. Blefari Melazzi, A. Detti, M. Pomposini: "Scalability Measurements in an Information-Centric Network", in "Measurement-based experimental research: methodology, experiments and tools", Springer Lecture Notes in Computer Science (LNCS), vol. 7586, 2012.

[14] I. Psaras, W. K. Chai, G. Pavlou, "Probabilistic in-network caching for information-centric networks", ACM SIGCOMM Workshop on Information-Centric Networking, ICN 2012.

[15] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures", ACM SIGCOMM workshop on Information-centric networking, ICN 2011.

[16] D. K. Smetters, V. Jacobson, "Securing network content", PARC TR-2009-1, October 2009.

[17] D. Galindo, F. D. Garcia, "A Schnorr-Like Lightweight Identity-Based Signature Scheme", in International Conference on Cryptology in Africa: Progress in Cryptology (AFRICACRYPT '09), Bart Preneel (Ed.). Springer-Verlag, Berlin, Heidelberg, 135-148.

[18] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes", Advances in Cryptology: Proceedings of CRYPTO 84, Lecture Notes in Computer Science, 7:47--53, 1984.

[19] D. Johnson and A. Menezes, "The elliptic curve digital signature algorithm (ECDSA)", Technical Report CORR 99-34, University of Waterloo, Canada, February 24 2000.

[20] R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM 21 (2): 120–126, 1978.

[21] ccnSim: scalable chunk-level simulator of Content Centric Networks, http://perso.telecom-paristech.fr/~drossi/index.php?n=Software.ccnSim

[22] Standards for efficent cripthography, "SEC 2: Recommended Elliptic Curve Domain Parameters", Certicom Research available at www.secg.org/collateral/sec2_final.pdf

[23] C. Fricker, P. Robert, J. Roberts, "A Versatile and Accurate Approximation for LRU Cache Performance", International Teletraffic Congress (ITC 24), September 4-7, 2012, Krakow, Poland.

[24] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: modeling, design and experimental results", IEEE JSAC, 20(7), 2002.

[25] D. Rossi, G. Rossini, "Caching performance of content centric networksunder multi-path routing (and more)", Tech. Report, Telecom ParisTech 2011, www.enst.fr/~drossi/paper/rossi11ccn-techrep1.pdf

[26] S. Arianfar, P. Nikander, and J. Ott. "On content-centric router design and implications", in ReArch Workshop, volume 9, page 5. ACM, 2010.

[27] A. Kate and I. Goldberg, "Distributed private-key generators for identity-based cryptography", International conference on Security and Cryptography for Networks (SCN'10), Springer-Verlag, Berlin, Heidelberg, 2010.

[28] R. Fonseca, V. Almeida, M. Crovella, and B. Abrahao, "On the intrinsic locality properties of web reference streams", IEEE INFOCOM, 2003.

[29] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in Named-Data Networking", ArXiv, abs/1208.0952, 2012.

[30] Y. Chabchoub, C. Fricker, F. Guillemin, and P. Robert, "Deterministic versus probabilistic packet sampling in the internet", in 20th international teletraffic conference, ITC20, 2007.

[31] G. Bianchi, A. Detti, A. Caponi, and N. Blefari Melazzi, "Check before storing: what is the performance price of content integrity verification in LRU caching?", SIGCOMM Comput. Commun. Rev. 43, 3 (July 2013).