Privacy Risks in Named Data Networking: What is the Cost of Performance?

Tobias Lauinger
Northeastern University,
Boston, USA
toby@ccs.neu.edu

Nikolaos Laoutaris
Telefónica Research,
Barcelona, Spain
nikos@tid.es

Pablo Rodriguez
Telefónica Research,
Barcelona, Spain
pablorr@tid.es

Thorsten Strufe
Technische Universität
Darmstadt, Germany
strufe@cs.tu-darmstadt.de

Ernst Biersack
Eurécom,
Sophia-Antipolis, France
erbi@eurecom.fr

Engin Kirda
Northeastern University,
Boston, USA
ek@iseclab.org

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
The authors take full responsibility for this article's technical content. Comments can be posted through CCR Online.

ABSTRACT

Named Data Networking architectures have been proposed to improve various shortcomings of the current Internet architecture. A key part of these proposals is the capability of caching arbitrary content in arbitrary network locations. While caching has the potential to improve network performance, the data stored in caches can be seen as transient traces of past communication that attackers can exploit to compromise the users' privacy. With this editorial note, we aim to raise awareness of privacy attacks as an intrinsic and relevant issue in Named Data Networking architectures. Countermeasures against privacy attacks are subject to a trade-off between performance and privacy. We discuss several approaches to countermeasures representing different incarnations of this tradeoff, along with open issues to be looked at by the research community.

Categories and Subject Descriptors

C.2.1 [**Computer Systems Organization**]: Computer-Communication Networks—*Network Architecture and Design*; K.6.5 [**Computing Milieux**]: Management of Computing and Information Systems—*Security and Protection*

General Terms

Design, Human Factors, Security

Keywords

Attacks, Caching, Named Data Networking, Next-Generation Internet Architecture, Privacy

1. INTRODUCTION

Caches are a useful tool for improving performance. When deployed in a network, they can provide their users with cached local copies instead of fetching data from the original source, which decreases response times and upstream network traffic. However, as a necessary part of their normal operation, caches store data from past communication. By observing and comparing response times, a user of a cache can distinguish a cache hit from a cache miss and thereby gather evidence of the past communication that has transited through the cache. In this way, information about communication by one user may leak to another user and ultimately lead to a breach of privacy.

As an example for such a privacy leak, consider the static images used in an online shop: The existence of certain combinations of such images in a cache could reveal what products a user has looked at or ordered. Similarly, cached voice data, even if encrypted, might indicate that a phone call is going on, and its addressing metadata can leak who is communicating with whom.

There is a fundamental trade-off between privacy and performance: Countermeasures against privacy leaks necessarily lead to reduced performance. In order to prevent attackers from detecting cache hits, for instance, the response time for cache hits and cache misses could be made the same, but this would imply artificially increased response times for cache hits. Alternatively, cache hits could be prevented altogether by not caching certain types of traffic. However, this traffic could not benefit from the performance gains of caching, and there would be at least a small performance penalty for traffic classification as well as a potential for misclassification.

The challenge in designing countermeasures against cache-based privacy attacks is to find an operating point in this trade-off that provides users with privacy while maximising performance. For instance, countermeasures against privacy attacks should have only minimal impact on traffic that is not privacy-sensitive. Named Data Networking (NDN) architectures with ubiquitous and protocol-agnostic caching are an interesting scenario to study this problem because they provide potentially far-reaching opportunities for cache-based privacy attacks and require mechanisms to protect their users' privacy.

2. PRIVACY IN NAMED DATA NETWORKS

Named data networking architectures [3] are an attempt at solving various shortcomings of the current Internet architecture: Instead of data packets that are forwarded between two end hosts addressed by their location, the new lingua franca of such networks are uniquely named, location-independent and freely duplicatable data objects. These data objects can be cached in arbitrary locations of the network, and their authenticity and integrity are guaranteed through digital signatures. Confidentiality and access control can optionally be ensured using encryption. Furthermore, the communication paradigm of the network is typically based on publish/subscribe or pull-based schemes. These changes are designed to provide a high level of content security, allow more flexible routing, and reduce latencies and redundancy in network links by locally caching popular objects.

However, any change in the architecture and paradigms of a network potentially introduces new opportunities for attacks. With respect to privacy, we identify the following two main categories of attacks in NDN architectures:

- *Information leakage through caches:* Communication leaves behind transient traces in caches, and this information might be available to any user of a cache. For instance, Krishnan and Monrose [7] exploited DNS responses cached in local resolvers to detect users performing web searches for predefined keywords when the web browser's prefetching feature was enabled. This attack was limited to information leaked through DNS queries, and the often relatively high number of users per DNS resolver made it more ambiguous which individual requested the information.

  In an NDN deployment, all network protocols that are subject to caching can become targets of cache-based attacks. Furthermore, the number of users per cache can be much lower if caches are located at lower aggregation levels, such as in DSLAMs, for instance. When fewer users share a cache, less prior (external) information is required to link an object found in the cache to the user who requested it. Because any type of network traffic could be cached in any location, the potential for cache-based privacy attacks is much higher in NDN architectures than in the current Internet.

- *Censorship and surveillance:* By giving unique names to data objects, NDN architectures might leak more fine-grained information about the contents or meaning of the data that is being exchanged than the current Internet with its data exchange based on location. In particular, named content makes it rather convenient for "privileged" stakeholders such as government authorities, ISPs or employers to block direct (i.e., not tunnelled) access to certain content names. Similarly, access to names can be monitored, even though tracing back data to the origin or destination might be more challenging because of the possible absence of location identifiers in the packets.

These potential attack vectors illustrate that network-level privacy is an important (and intrinsic) concern in NDN architectures. Censorship and surveillance attacks have already received some attention in the form of steganography [1] and tunnelling [5] countermeasures. The category of cache-based attacks is much less thoroughly explored. The authors of this editorial discussed the possibility of privacy attacks based on caches [8, 9] in Content-Centric Networking [6], but the question of countermeasures is still open. While tunnelling approaches can also protect users against cache-based privacy attacks by unprivileged attackers, the generalised use of tunnelling would undo several advantages of NDN architectures and might negatively impact performance by precluding the possibility of caching before the endpoint of the tunnel. However, if a user's ISP (and government) can be trusted, much more lightweight approaches are conceivable to prevent everyman from spying on their neighbours.

3. INFORMATION LEAKS

The privacy attacks that we consider in this editorial refer to information leaked from content objects being transmitted and cached in the network. Note that we allow content objects to be encrypted. In that case, the relevant information leaks through the associated unencrypted metadata, such as the name/identity or the size[1] of the object, and through network-level observations such as the time of the request. In other words, the privacy leak does not necessarily refer to the information contained in the object, but may simply be the fact that the object is being requested by someone.

In the following, we focus on privacy attacks based on network caches that can be carried out by any user of a cache. Such caches can be located in arbitrary network devices (such as routers), and they can potentially cache any type of data transiting through the network. On a high level, a privacy leak with respect to a cache arises when an attacker can (1) detect that a privacy-sensitive object is cached, and (2) can infer which user requested the object.

What exactly constitutes a privacy-sensitive object depends a lot on the context, including the time and location of the associated request for that object. Certain political or religious beliefs, medical conditions, sexual practices, or even music tastes, for instance, might be encouraged or tolerated in some locations but could lead to blackmail, persecution or prosecution in others. Consequently, an object is not privacy-sensitive per se; it is in a privacy-sensitive state only with respect to the circumstances under which it is requested.

Most objects can become privacy-sensitive only if they can be linked to an individual, potentially using prior or external information. For instance, an object might be so specific or might have been requested at a certain point in time so that an attacker can infer which user requested the object. While this step may seem difficult to automate, a determined attacker (who might be anyone from a curious neighbour to an envious colleague, identity thief or industrial spy) must be expected to have sufficient prior knowledge and time to identify and attribute privacy-sensitive objects.

4. PRIVACY ATTACKS

Since an attacker in our attack model has no access to the network traffic transmitted over the ISP's links, the only way to detect requests is to check whether the corresponding data is present in a shared cache. To do so, the attacker first needs to compile a list of the names of privacy-sensitive and attributable objects. The attacker then (periodically) checks whether any of the corresponding objects is cached, which would mean that someone from the cache's user population has requested the object.

In a technical report [9], we described how this attack could be carried out in a deployment of CCNx (PARC's prototype implementation of Content-Centric Networking [6]), addressing issues such as how often an attacker needs to request an

[1]Prior work demonstrated that phrases spoken in encrypted voice conversations can be guessed [10] if objects are not padded to a constant size; a similar attack targeted interactive web browsing sessions protected with SSL or WPA [2].

object, and how a cache hit can be detected. For the purpose of this editorial note, we discuss this issue at a very high level that is likely to apply to other NDN architectures, too.

The main assumption of the attack is that the attacker can detect cache hits. An NDN protocol might provide functionality to request an object only if it is cached (e.g., by allowing users to set a hop counter in request messages); this would directly leak the required information to the attacker. If this option is not available, an attacker can still exploit the fact that a cached content object is delivered much faster than an object fetched from a remote location and can *guess* whether an object was cached based on the response time.

The feasibility and reliability of such guesses depend on many (and as of today unknown) deployment details such as the network infrastructure and the routing protocol, which make it difficult to address this issue in detail at this time. However, it is important to understand that the underlying issue is system-inherent and that, therefore, there will be (at least some) scenarios where the attack can be carried out successfully. Furthermore, an attack algorithm does not need to be efficient to be a threat: Consider, for instance, an attacker who is connected to the same DSLAM as the victim, where the DSLAM is the first cache and all content requests are routed through this cache. Such an attacker can afford to send hundreds of requests per second without exceeding the capacity (and cost) of a standard DSL subscription.

5. "NAÏVE" COUNTERMEASURES

Countermeasures against the attack can be classified into *detection* and *prevention* approaches. The former category of countermeasures aims at detecting attack instances, punishing attackers and thereby dissuading future attackers. The latter category targets the assumptions needed for the attack, which makes it more difficult (or impossible) for an attacker to be successful in the first place.

5.1 Detecting an Attack

Cache-based privacy attacks could be detected using techniques similar to prior work that has addressed cache pollution attacks [4]. Cache pollution attacks encompass, for instance, creating a fake popularity for an otherwise unpopular object, which is similar to what happens when an attacker repeatedly requests an object to check whether it is cached. However, there are three significant challenges to the detection approach: Firstly, attackers could potentially vary in their behaviour to evade detection. Secondly, such an approach requires state and processing power in routers, which can be expected to be a scarce and expensive resource in content routers. Thirdly, some NDN architectures make it technically infeasible to identify the origin or destination location of a message within the network, meaning that attack attribution (and dissuasion) can work only in access routers where users are directly connected.

5.2 Preventing an Attack

To render attacks unsuccessful, the functionality used and the assumptions made by the attacker could be targeted:

- The protocol functions used by the attacker (such as the hop count field) could be disabled,
- attempts could be made to close the timing side channel (by delivering cached objects with the same delay as when they were fetched initially), and
- the network deployment could be chosen such that caches are shared by larger numbers of users.

However, the problem with this approach is that it removes some of the advantages of an NDN architecture and may result in less functionality and potentially worse performance.

6. "SELECTIVE" COUNTERMEASURES

The countermeasures described above affect even traffic that is not critical with respect to privacy. However, most traffic (in terms of bytes) is probably not privacy-sensitive. Therefore, countermeasures against privacy attacks may achieve better overall performance if they apply only to the privacy-sensitive part of the network traffic. An example for such a strategy could be to prevent privacy-sensitive traffic from being cached, so that potential attacks yield less useful information and the users' privacy is preserved. The privacy/performance tradeoff thus becomes one of accurately classifying traffic into sensitive and non-sensitive communication with a reasonable performance penalty.

Privacy-sensitive content has specific properties: It has *low local and instantaneous popularity.* Note that we consider popularity in terms of the individuals who request the content, and as a function of time and location.

- *Low content popularity* is characteristic for our definition of privacy-sensitivity because a content object that becomes more popular also becomes more difficult to link to an individual user (since there are more users who might potentially have requested it). Furthermore, a request for a more popular content object has a lower entropy and is therefore less revealing to an attacker. Popularity needs to be defined in terms of individuals (and not requests) because 100 requests for the same object made by 100 different individuals, for instance, might not be as telling as 100 requests for the same object made by one single individual.

- When considering an object's popularity, it is important to factor in *context* (i.e., time and location), and to do this at the right granularity: An object that is popular city-wide might be unpopular in some of the city's neighbourhoods, and the popularity of an object can be very different during daytime and nighttime, for instance. A request for an otherwise popular object in a place or at a time when it is locally and instantaneously unpopular can still leak information. For instance, a transaction at a popular web shop might be attributable to an individual during nighttime if that person is the only one in the neighbourhood with their lights on.

These properties could be leveraged to classify traffic and to design more selective and less obtrusive countermeasures against privacy attacks.

6.1 Selective Caching

The low popularity of privacy-sensitive objects can help to tie together performance and privacy objectives instead of opposing them: In order to maximise performance (in terms of cache hits), low-popularity objects should not be cached. Therefore, optimising for performance could also improve privacy. However, while these two objectives are similar, they are not the same: Protecting privacy requires a much more sophisticated definition of popularity, and one would also expect that certain privacy guarantees hold at *any* time as opposed to performance considerations that are typically average or long-term objectives.

Entirely network-based traffic classification for selective caching can be expected to be challenging to implement in practice. For instance, in order to guarantee that a privacy-sensitive object is *never* cached, an object may be inserted into a cache only when it has already reached high popularity, and it must be evicted as soon as its popularity begins to vanish. Thus, one of the main research challenges is how to assess popularity correctly without consuming too many resources. Furthermore, new attacks might attempt to create a fake popularity for objects so that they can be used in subsequent privacy attacks.

An in-network approach has the advantage of being of benefit to all users, and it can leverage the fact that the users' interest in privacy and the network operators' interest in making caching most effective are similar and can potentially be aligned. However, if the cost of assessing the extended popularity and of giving privacy guarantees is too high, an end-to-end approach could be more promising.

6.2 Selective Tunnelling

Instead of classifying traffic in the network, this task could be carried out on the end hosts. Sensitive content can be tunnelled to a trusted endpoint (using a system such as Andana [5], for instance) while the non-sensitive content is transferred over the network as usual. If users trust their ISP and make the content classification available to the network, an intermediate solution is possible: The end hosts classify and flag sensitive requests or data, and the ISP isolates them from non-sensitive content, e.g. by not caching flagged content. This measure could be restricted to the access network; flags can be ignored in the core network where requests are sufficiently aggregated and attacks by end users become too expensive.

Ideally, the classification process should be automated because users who are unaware of the problem risk intrusion into their privacy, while users who are aware of the problem but find the classification process too tedious might simply flag everything as sensitive, which might negatively affect the overall network performance. The usual considerations of end-host vs. in-network classification apply, that is, end hosts have more domain knowledge and can rely more on user input or customisation, but they have no global view and it is more difficult to enforce correct usage of the mechanism.

7. CONCLUSION

Caching of data can provide better performance, but it comes at the cost of introducing a potential for privacy breaches. This risk is aggravated if caching is made ubiquitous and general-purpose, as it is possible (and suggested) for NDN architectures.

The tradeoff between performance and privacy can be tackled at various layers of abstraction by deciding whether certain protocol features should be allowed, at what aggregation level caches should be placed, and what content may be cached. Given a (hypothetical) means of classifying objects according to their sensitivity, the most fine-grained approach is to leave the majoritarian non-sensitive traffic unaffected and to prevent privacy-sensitive content from being cached. Content objects involved in privacy-sensitive communication are very likely to have low popularity, thus precluding them from caching not only improves privacy, but might increase the overall network efficiency as well. However, assessing the sensitivity of content is a difficult problem; the heuristic outlined in this short discussion certainly needs refinement.

Because the privacy/caching issue is intrinsic in NDN architectures, there is an urgent need for at least a coarse privacy concept, one that goes beyond the "tunnel your communication when you believe you have something to hide" approach. Our daily communication traces can be misused and exploited in so many (not necessarily foreseeable) ways that it is not realistic to expect users to correctly appreciate when they need to have protection mechanisms switched on. Network planners and operators have good reasons to take into consideration users wishing to protect their privacy—if every user started tunnelling their traffic, on the path from the end host to the tunnel endpoint there would be not much left from the great advances of an NDN architecture.

8. REFERENCES

[1] S. Arianfar, T. Koponen, B. Raghavan, and S. Shenker. On preserving privacy in content-oriented networks. In *ICN '11*. ACM, Aug. 2011.

[2] S. Chen, R. Wang, X. Wang, and K. Zhang. Side-channel leaks in web applications: A reality today, a challenge tomorrow. In *Symposium on Security and Privacy*. IEEE Computer Society, May 2010.

[3] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi. A survey on content-oriented networking for efficient content delivery. *IEEE Communications Magazine*, 49(3):121–127, 2011.

[4] L. Deng, Y. Gao, Y. Chen, and A. Kuzmanovic. Pollution attacks and defenses for Internet caching systems. *Computer Networks*, 52(5):935–956, 2008.

[5] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Unzun. ANDaNA: Anonymous named data networking application. In *NDSS '12*, Feb. 2012.

[6] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. In *CoNEXT '09*. ACM, Dec. 2009.

[7] S. Krishnan and F. Monrose. DNS prefetching and its privacy implications: When good things go bad. In *LEET '10*. Usenix, Apr. 2010.

[8] T. Lauinger. Security & Scalability of Content-Centric Networking. Master's thesis, Eurécom, Sophia-Antipolis, France and Technische Universität Darmstadt, Germany, Sept. 2010.

[9] T. Lauinger, N. Laoutaris, P. Rodriguez, T. Strufe, E. Biersack, and E. Kirda. Privacy Implications of Ubiquitous Caching in Named Data Networking Architectures. Technical Report TR-iSecLab-0812-001, iSecLab, Aug. 2012.

[10] C. V. Wright, L. Ballard, S. E. Coull, F. Monrose, and G. M. Masson. Spot me if you can: Uncovering spoken phrases in encrypted VoIP conversations. In *Symposium on Security and Privacy*. IEEE Computer Society, 2008.