Covert Ephemeral Communication in Named Data Networking

Moreno Ambrosin, Mauro Conti
University of Padua, Italy
{ambrosin,conti}@math.unipd.it

Paolo Gasti
New York Inst. of Technology
pgasti@nyit.edu

Gene Tsudik
University of California, Irvine
gts@ics.uci.edu

ABSTRACT

In the last decade, there has been a growing realization that the current Internet Protocol is reaching the limits of its senescence. This has prompted several research efforts that aim to design potential next-generation Internet architectures. Named Data Networking (NDN), an instantiation of the content-centric approach to networking, is one such effort. In contrast with IP, NDN routers maintain a significant amount of user-driven state. In this paper we investigate how to use this state for covert ephemeral communication (CEC). CEC allows two or more parties to covertly exchange ephemeral messages, i.e., messages that become unavailable after a certain amount of time. Our techniques rely only on network-layer, rather than application-layer, services. This makes our protocols robust, and communication difficult to uncover. We show that users can build high-bandwidth CECs exploiting features unique to NDN: in-network caches, routers' forwarding state and name matching rules. We assess feasibility and performance of proposed cover channels using a local setup and the official NDN testbed.

1. INTRODUCTION

The current IP-based Internet architecture represents an unprecedented success story, wildly exceeding its designers' expectations in terms of adoption, size of deployment and scalability. Part of IP's success is due to its light-weight design: virtually all state used for communication is maintained at the endpoints, rather than within the network. For this reason, IP-based networks are – arguably, by design – extremely robust against random failures. However, lack of in-network state is the reason for some of IP's shortcomings, including poor support for efficient large-scale content distribution.

Content distribution currently accounts for most Internet traffic [26]. Therefore, most major services [30, 17, 11, 15] have been – for performance, cost and reliability reasons [3] – relying on Content Distribution Networks (CDNs): large, complex, geographically distributed infrastructures implemented at various layers of the networking stack that efficiently deliver content to end users. This state of affairs motivated research into new networking architectures that can better serve today's Internet traffic. Named Data Networking (NDN) [19] is one of these architectures.

NDN is an example of Content-Centric Networking (CCN). In NDN, location-agnostic content is directly addressable by name, regardless of who publishes it. This allows routers to store a copy of forwarded data in a local cache, which can be used to satisfy subsequent requests. Content is requested using a special kind of packets, called *interests*. Interests are routed similarly to IP packets; however, content is forwarded along the reverse path traversed by the corresponding interest. Data forwarding information is stored by routers, for a short amount of time, in a data structure called Pending Interest Table (PIT).

User-driven soft-state on routers facilitates efficient content distribution at the network layer. However, availability of this state within the network creates a new set of problems. In particular, NDN prompts new security [12, 2, 8, 27, 9, 29] and privacy [1, 10] issues. In this paper, we investigate whether router state can be used for covert and ephemeral communication. We show how two parties can secretly communicate, without directly exchanging any packets, and without injecting new content into the network (i.e., without publishing new data). This is a significant departure from what can be done with IP, where lack of user-driven state within the network forces users to rely on the application layer for implementing covert channels.

We believe that this work is both timely and important. The former – because of a recent surge of interest in content-centric networking, and in NDN in particular. The latter, because, to the best of our knowledge, it represents the first attempt to identify and address covert ephemeral communication (CEC) in a content-centric architecture. CEC is, in fact, relevant in many realistic scenario, e.g.:

1. In tightly-controlled environments, where mandatory access control is in place (e.g., in the military), CEC can be used to exfiltrate sensitive information, possibly collected by malware. Ephemeral nature of published data makes subsequent forensic analysis difficult.
2. In countries with oppressive governments, civil rights activists can covertly communicate to coordinate and exchange information. CEC offers plausible deniability.

Studying whether CEC in NDN is possible – and how to implement it – is an important step towards fully understanding this means of communication, regardless of whether

NDN sees limited deployment (e.g., as an overlay on top of IP) or widespread adoption (i.e., as a replacement for IP).

With this motivation, we design several protocols for exchanging covert ephemeral messages (CEMs) between a single sender and one or more receivers. We perform extensive evaluation of our techniques on a local network and on a geographically distributed NDN testbed. Our experiments confirm that CEC is indeed possible, and show that our techniques provide high bandwidth and low error rate.

Organization. We present an overview of NDN in Section 2. Section 3 introduces the system model. We present the delay-based CEC mechanisms in Section 4 and common-prefix-based CEC techniques in Section 5. Section 6 discusses sources of error and error handling. Experimental results are described in Section 7. Security analysis is discussed in Section 8. Section 9 reviews related work. We conclude in Section 10.

2. NDN OVERVIEW

In this section we present an overview of NDN. Readers familiar with NDN may skip this section without loss of continuity.

NDN is a networking architecture based on named data. Data is requested via *interests*, and delivered in *data packets* [7]. Data packets include a name, a payload and a digital signature computed by the content producer.[1] A name is composed of one or more components, which have a hierarchical structure. In NDN notation, "/" separates name components, e.g., `/cnn/politics/frontpage`. Content is delivered to consumers only upon explicit request, which can include the full name of a particular data packet or a prefix of such a name – e.g., `/cnn/politics` is a prefix of `/cnn/politics/frontpage`. In case of multiple data packets under a given name (or prefix), optional control information can be carried within the interest to restrict desired content. If no additional information is provided, producers and routers return arbitrary data packets matching the request (preferably, from a local cache).

If no local copy of a data packet is available, NDN routers forward interests towards content producers responsible for the requested name, using name prefixes (instead of today's IP address prefixes) for routing. Each NDN router maintains a Pending Interest Table (PIT) – a lookup table containing outstanding [*interest*, *arrival-interfaces*] entries. When an NDN router receives an interest, it first looks up its PIT to determine whether another interest for the same name is currently outstanding. There are three possible outcomes: (1) If the same name is already in the router's PIT and the arrival interface of the present interest is already in the set of *arrival-interfaces* of the corresponding PIT entry, the interest is discarded. (2) If a PIT entry for the same name exists, yet the arrival interface is new, the router updates the PIT entry by adding a new interface to the set. The interest is not forwarded further. (3) Otherwise, the router creates a new PIT entry and forwards the present interest. We refer to (1) and (2) as PIT hit, and to (3) as PIT miss.

Upon receipt of the interest, the producer injects a matching data packet into the network, thus *satisfying* the interest. The requested content is then forwarded towards the consumer, traversing – in reverse – the path of the corresponding interest. Each router on this path deletes the PIT entry corresponding to the satisfied interest. In addition, each router caches a copy of forwarded content in its local cache.

Unlike their IP counterparts, NDN routers can forward interests out on multiple interfaces simultaneously. This is done in order to maximize the chances of quickly retrieving requested content. A router that receives an interest for already-cached content does not forward the interest further; it simply returns cached content and retains no state about the interest.

Not all interests result in content being returned. If an interest encounters either a router that cannot forward it further, or a content producer that has no such content, no error packets are generated. PIT entries for unsatisfied interests in intervening routers are removed after a predefined *expiration* time. The consumer can choose whether to regenerate the same interest after a timeout.

3. SYSTEM MODEL

A CEC system involves a sender (Snd) and one or more receivers (Rcv). Snd wants to covertly publish a *time-bounded* (i.e., ephemeral) message M, while Rcv wants to retrieve it. A time-bounded message can only be read for a given period of time [4], after which it becomes unavailable, i.e., it *expires*. Depending on the scenario, the action of retrieving a CEM either makes it expire immediately, or "refreshes" it, hence deferring its expiration.

Snd and Rcv are not allowed to communicate directly. For example, the Internet provider of Snd and Rcv might monitor all activity between its users. Moreover, Snd and Rcv are not allowed to use services (such as email or on-line forums) to exchange data indirectly. Snd and Rcv have access to a producer (Pr), which is unaware of Snd and Rcv's intent to communicate, and only hosts content that cannot be modified by consumers.

All packets to and from Pr are routed through an NDN router (Rt), which caches all data packets it forwards. At first we will assume that Rt is Snd and Rcv's first-hop router. We will then relax this assumption, allowing Rt to be an arbitrary number of hops away from both. Figure 1 depicts our model.

We assume that Snd and Rcv have tightly synchronized clocks.[2] We believe that this assumption is realistic: two parties can use NTP servers or GPS devices to synchronize their clocks accurately, i.e., within 500 ns to a few milliseconds, depending on the synchronization protocol [25].

The adversary (Adv) has three goals: (1) detecting CEMs from Snd to Rcv; (2) preventing Snd and Rcv from communicating; and (3) accessing CEMs after they expire. Adv can monitor and modify traffic between users. Following the *retroactive privacy* definition of [4], we say that a CEC system is secure if any efficient Adv can win the following game with probability at most negligibly over 1/2:

1. Adv selects two same-length message M_0 and M_1, and sends them to Snd.
2. Snd selects a random bit a and publishes M_a.
3. After M_a is expired, Adv tries to retrieve M_a.
4. Adv outputs its guess a' for a; Adv wins if $a' = a$.

In all the proposed CECs, after Snd has sent a CEM, it

[1]Data packets also carry additional fields that are not relevant to this paper and are therefore ignored.

[2]However, this is not required in all our protocols.

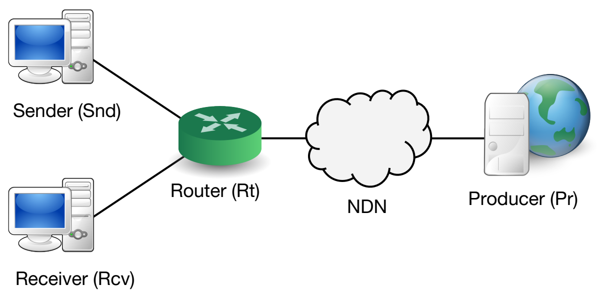


Figure 1: System model.

deletes locally. Similarly, Rcv deletes all CEMs soon after receiving them, i.e., before the messages expire. We assume that all parties can effectively delete data.

4. DELAY-BASED COVERT COMMUNICATION

Delay-based communication relies on the ability of Rcv to differentiate between a cache (or PIT) hit, and a cache (PIT) miss. Snd can exploit this by selecting a set of packets for which it issues interests, therefore causing cache/PIT hits for Rcv.

As a warm-up, we show how timing information can be used to covertly transmit a single-bit CEM from Snd to Rcv. Then, we describe how this can be efficiently extended to CEMs of arbitrary length. To simplify our notation, we refer to the RTT of a pair interest-data packet as the RTT of the data packet.

4.1 Single-Bit Transmission via Cache

We now show how Snd sends $b \in \{0,1\}$ to Rcv. If $b = 1$, Snd requests a data packet C. Otherwise, it does nothing. Rcv determines the value of b by requesting the same data packet C. If the RTT of C is below the expected RTT for non-cached data packets, Rcv sets b' to 1. Otherwise, b' is 0. This mechanism works reliably, i.e., $b' = b$ with overwhelming probability, if the following conditions are met:

1. Snd and Rcv agree ahead of time on a data packet that will be used for communicating, and when Snd will send b.
2. C must be non-popular, i.e., it should not be in Rt's cache prior to Snd's request.
3. There must be separation between the RTTs associated with cache hits and cache misses, and Rcv must have a good estimate for at least one of them with respect to C.
4. Rt should cache data packets for a non-negligible amount of time.

We believe that 1 and 2 can be easily satisfied in practice. With respect to 3, in order to distinguish a cache hit from a miss, Rcv must determine an appropriate threshold value t_{thresh}: iff the RTT of C is below t_{thresh}, then Rcv considers C as originating from a nearby cache. t_{thresh} can be estimated by requesting (more than once) a large number of non- popular data packets from the same producer that distributes C. The first interest for each data packet will be satisfied by the producer itself. All subsequent (closely spaced) requests for the same data packet will come from a nearby cache. Regardless of the network topology, there is usually a clear separation between cache hits and cache misses (see Section 7, figures 3a and 3b) and, therefore, also an appropriate value for t_{thresh}.

Rcv can determine if condition 4 holds by issuing multiple interests for data packets distributed by multiple producers, and measuring effects (if any) of content caching. If 4 does not hold, a different mechanism – such as the one based on PIT – is more appropriate.

We say that a CEM exchanged by Snd and Rcv is expired if C has been removed from all caches, or once it has been retrieved by Rcv.

Timing Constraints. In order to receive b reliably, Rcv must observe a set of timing constraints. In particular, Rcv's interest for C must be processed by Rt after C is cached (and made available to consumers), but before C expires from the same cache. (Without loss of generality, in the rest of the paper we assume that data packets in Rt's cache are available to consumers as soon as they are received by the router.) Let I indicate an interest for C, and $[I : A \rightarrow B]$, $[C : A \rightarrow B]$ the time required to I and C to travel from A to B. Let t_0 be the time at which Snd writes b, either by issuing I ($b = 1$) or by doing nothing ($b = 0$). Let $t_C = [I : \mathsf{Snd} \rightarrow \mathsf{Pr}] + [C : \mathsf{Pr} \rightarrow \mathsf{Rt}]$. C is available from Rt's cache at $t_0 + t_C$. Therefore, Rcv can "read" b starting at $t_b = t_0 + t_C - [I : \mathsf{Rcv} \rightarrow \mathsf{Rt}]$. When $[I : \mathsf{Snd} \rightarrow \mathsf{Rt}] \approx [I : \mathsf{Rcv} \rightarrow \mathsf{Rt}]$, $t_b \approx t_0 + \mathrm{RTT}_{\mathsf{Rt}\rightarrow\mathsf{Pr}}$ where $\mathrm{RTT}_{\mathsf{Rt}\rightarrow\mathsf{Pr}}$ represents the RTT for C between Rt and Pr. Rcv must retrieve b before $t_b + Exp_{\mathsf{Rt}}$, where Exp_{Rt} represents the freshness field of C, or the time after which C is evicted from Rt's cache, whichever comes first. Figure 2a summarizes these observations.

Time needed to read a single bit depends on the RTT associated with a cache hit, from Rcv's point of view. Let $\mathrm{RTT}_{\mathsf{hit}}$ and $\mathrm{RTT}_{\mathsf{miss}}$ indicate the average RTT for a cache hit and cache miss relative to C, as observed by Rcv. Rcv sets $b = 1$ iff the RTT of C is below $\mathrm{RTT}_{\mathsf{hit}} + \Delta < \mathrm{RTT}_{\mathsf{miss}}$, where Δ is a small constant used to account for variance in C's RTT. Rt can therefore determine b within $\mathrm{RTT}_{\mathsf{hit}} + \Delta$.

Covert messages distributed with this technique are ephemeral, i.e., they become unavailable after a certain amount of time without any further action from Snd or Rcv. Because Rt caches forwarded traffic, C will eventually be evicted from Rt's cache. In fact, we argue that C is always a good candidate for deletion: since C is not popular, both Least Frequently Used (LFU) and Least Recently Used (LRU) cache replacement policies will consider it for removal relatively early.

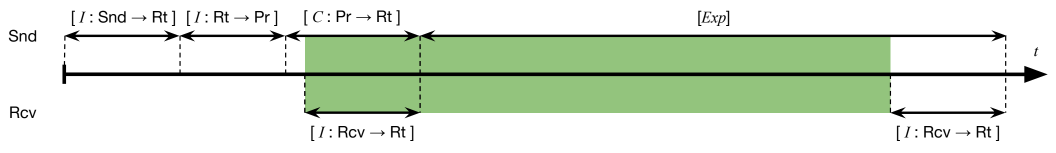
Once Rcv requests C, it will be stored in cache regardless of the original value of b. Therefore, after being retrieved, b will be set to 1 until C is evicted from Rt's cache.

Our experiments, reported in Section 7, show that this technique provides high bandwidth, with low error. Moreover, it is relatively easy to implement, since it does not require strict time synchronization.

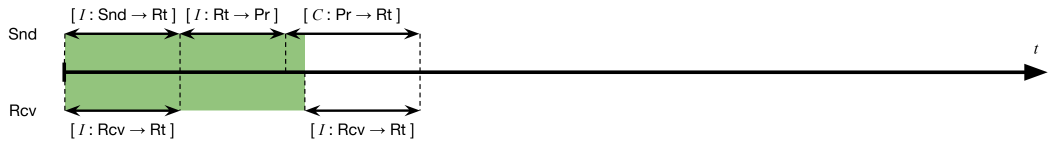
4.2 Single-Bit Transmission via PIT

In some circumstances, cache-based CEC is not applicable:

1. Rt might have no cache: small, low-cost, low-power embedded routers may not store forwarded data packets.
2. Rt's entire cache may be overwritten before Rcv issues I. This can happen if Rt's cache is very small, and the router forwards a large amount of traffic.



(a) Cache



(b) PIT

Figure 2: Time constrains for retrieving a CEM published using Rt's cache (top) and PIT (bottom). The colored area delimits the interval in which Rcv can retrieve b.

3. To prevent cache pollution attacks [9, 29], Rt may not add to its cache data packets that are forwarded only once. This behavior would force Snd to issue I multiple times before C is stored in Rt's cache, negatively affecting both bandwidth and detectability.
4. Rt may implement cache privacy techniques that involve delaying serving C when it is retrieved from the cache [1].

To overcome the above limitations, we design a technique based on PIT state. This technique requires strict time synchronization between parties. It is based on PIT hits (see Section 2): when Rt receives interest $I' = I$, while I is still in Rt's PIT, the two interests are "collapsed" within the same PIT entry. Rt adds the incoming interface of I' to the PIT entry of I, and does not propagate I' any further. Once C is received by Rt, it is forwarded to the interfaces on which I and I' were received.

This feature of NDN can be used by Snd and Rcv to covertly exchange a bit b as follows. If $b = 1$, Snd issues I, otherwise it does nothing. To receive b, Rcv issues $I' = I$ while a copy of I intents is still in Rt's PIT – if originally issued by Snd. If I is in Rt's PIT, then Snd's interest will be satisfied more quickly than if it was not, because either the original I would be already past Rt, or C would be already on its way back to Snd. If Rcv can correctly measure the corresponding difference in RTT, it can reliably receive b.

With this technique, we say that a CEM has expired if I has been removed from Rt's PIT *and* from all caches, or it has been retrieved by Rcv.

Timing Constraints. While the PIT-based CEC works regardless of Rt's cache behavior (or even cache availability), it imposes much stricter timing requirements on Rcv. In fact, I' must be received by Rt after I (if issued) is added to Rt's PIT. Moreover, I' must be received before C is forwarded to by Rt. This gives to Rcv a time window of $\text{RTT}_{\text{Rt}\to\text{Pr}}$.

As in the cache-based technique, messages exchanged via PIT are ephemeral: if I is not issued on time, the corresponding PIT entry will be removed once C is forwarded back to Snd. Also, after Rcv issues I, any attempt to retrieve b within the correct timing constraints will result in a collapsed interest (and therefore set $b = 1$), regardless of the original value of b. Figure 2b gives a graphical representation these constraints.

4.3 Tandem Data Packets

With geographically distributed deployments of NDN, and when Rt is far from Rcv, RTT associated with cache hits and misses may fluctuate significantly over time. In order to reduce the probability of erroneously detecting cache hits/misses, we introduce a technique – called Tandem Data Packets (TDP) – that uses two data packets to covertly receive a single bit. To transmit b, Snd and Rcv agree on data packets C_0 and C_1, which are assumed not to be in any cache. Then, Snd requests C_b. Rcv issue two consecutive interests, one for C_0 and one for C_1; if RTT of C_0 is lower than RTT of C_1, Rcv sets $b' = 0$, otherwise it sets $b' = 1$. The CEM is exchanged correctly if $b' = b$.

This technique does not require Rcv to make any *a priori* assumption on the exact RTT associated with cache hits and misses, besides the fact that the RTT of C_b is lower than the RTT of $C_{\neg b}$. As our experiments confirm, this reduces receiver error when, since RTT for both hits and misses is continuously updated according to network conditions.

With this technique, after it is obtained by Rcv b becomes inaccessible. In fact, both C_0 and C_1 will be stored in Rt's cache, due to Rcv's interests. Therefore, any difference in the RTT associated with C_0 and C_1 will not depend on b. Therefore, b expires if it has been removed from Rt's cache or it has been retrieved by Rcv.

Timing Constraints. Timing constraints are identical to those in Section 4.1.

4.4 Transmission of Multi-Bit Messages

Snd and Rcv may want to exchange messages composed of more than one bit. We discuss how to determine Snd's and Rcv's speed separately, since the two may send and receive at different rates.

Let $M = b_1, \ldots, b_n$ be an n-bit string. Suppose that Snd and Rcv agree on n different data packets $C_1, ..., C_n$. Instead of waiting for the full RTT of C, Snd can send new I_i for C_i before C_{i-1} has been received. Snd selects an interval t; two consecutive interests I_i, I_{i+1} are sent at t_i and t_{i+1}, where $t_{i+1} = t_i + t$. The minimum value for t is denoted as t_{min}, and corresponds to sending an uninterrupted burst of interests.

Similarly, Rcv selects a value t which is used to determine how subsequent interests are spaced. Snd and Rcv can select

t independently, as long as the timing constraints associated with the protocol are not violated.

We evaluate how this technique affects transmission error as a function of t and report our findings in Section 7.

Transmitting Multiple Bits with a Single Interest. For efficiency reasons, Snd can use a generalization of the TDP technique to send multiple bits using a single interest. Two parties agree a priori on a set of data packets, which we represent as a matrix:

$$Y = \begin{bmatrix} C_{(1,1)} & \cdots & C_{(1,2^m)} \\ \vdots & & \\ C_{(\ell,1)} & & C_{(\ell,2^m)} \end{bmatrix}$$

where m is the number of bits transmitted using one interest, and $\ell = \lceil n/m \rceil$. In order to publish M, Snd splits it in words $W_1, \ldots, W_\ell$ of m bits each (i.e., $W_1 = (b_1, \ldots, b_m)$, $W_2 = (b_{m+1}, \ldots, b_{2m})$, etc.). Rcv then issues interests for $C_{(1,W_1)}, C_{(2,W_2)}, \ldots, C_{(\ell,W_\ell)}$, where W_i is used as integer representation of the corresponding bit string. Thus, Snd can publish an n-bit message using $\lceil n/m \rceil$ interests.

To retrieve M, Rcv issues interests for all data packets in Y. Let $C_{i,j}$ be the data packet on the i-th row of Y such that the RTT of $C_{i,j}$ is the smallest across all $C_{i,1}, \ldots, C_{i,2^m}$. Rcv sets $W_i = j$, and $M = W_1 | \ldots | W_\ell$. The cost of retrieving M for Rcv is therefore exponential in m. (In practice, reasonable values for m are between 1 and 5). Note that when $m = 1$, this technique corresponds to TDP.

5. COMMON-PREFIX-BASED COVERT COMMUNICATION

Using previous techniques, a covert message can be retrieved only by a single receiver. Message is automatically "deleted" after it is "read" by Rcv. This is desirable when a CEM has only one intended recipient. However, when the CEM has multiple recipients, Snd must create a separate "instance" of the message for each. In this section, we propose a technique – called Common-Prefix-Based Covert Communication (CPC) – that allows Snd to publish a message once, and have multiple parties to retrieve it. Similarly to previous techniques, CEMs published using CPC are ephemeral.

CPC relies on NDN's longest prefix matching feature, instead of RTT measurements. This makes it robust against cache privacy techniques [1], which could defeat CEC techniques introduced in Section 4.

Communication via CPC works as follows. Snd and Rcv agree on two data packets C_0, C_1 which share a common name prefix, e.g., `/common/prefix/C0`, and `/common/prefix/C1`. [3] The common namespace is selected such that data packets published under it are not popular, i.e., not in Rt's cache. In order to transmit a single bit, Snd simply requests C_b. To receive b, Rcv issues an interest for `/common/prefix/`. Both C_0 and C_1 match Rcv's interest. Therefore, Rt will return one data packet among C_0 and C_1 that is still in its cache – or in its PIT, if Snd and Rcv's interests are closely spaced (see timing constraints below). This communicates b to Rcv.

This technique is very robust against changing network conditions. In particular, since timing is not used to either set or determine b, transient changes in RTT do not introduce communication errors: Rcv receives only C_b, regardless of how long it waits. Moreover, in contrast with previous techniques, when Rcv's interest is dropped (or, similarly C_b in response to Rcv's interest is dropped) Rcv can re-issue its interest, since this process does not affect C_b.

Common-prefix-based covert channels are suitable for distributing a single message to a (possibly large) set of receivers. Each interest for `/common/prefix/` issued by a recipient has the side-effect of "refreshing" C_b in Rt's cache, making b available longer. After recipients stop retrieving C_b, it "fades away" from all involved routers' caches, effectively erasing b. As an alternative, Snd or one of the recipients can request $C_{\neg b}$ which achieves a similar result.

A message exchanged using CPC expires when it is removed from all caches.

Timing Constraints. In order to successfully retrieve b, Rcv must issue an interest for `/common/prefix/` such that the interest is received by Rt after the interest for C_b from Snd. If the interest from Rcv is received before C_b is returned to Rcv, communication between Snd and Rcv is implemented through Rt's PIT. Otherwise, Rt's cache is used to exchange b. Snd's interest must also be received by Rt before C_b is removed from the cache.

5.1 Multiple-Bit Transmission

Since this technique is less susceptible to RTT fluctuations and packet loss, using it for sending and receiving multiple bits in bursts does not introduce significant errors. This is confirmed by our experiments, in Section 7.

Transmitting Multiple Bits with a Single Interest. Snd and Rcv can agree on data packets in matrix Y with the additional requirements that for $i \in [1, \ell]$, data packets in row i share the same common prefix $pref_i$. Snd splits M in $W_1, \ldots, W_\ell$, and – for each i – issues one interest for C_{i,W_i}.

Rcv needs to issue only *one* interest per word (i.e., per matrix row), requesting a data packet from $pref_i$. For this reason, Snd and Rcv can exchange an n-bit message using $\lceil n/m \rceil$ interests/data packets each.

In practice, m is limited only by availability of un-popular namespaces containing a sufficient number of data packets.

6. ERRORS AND ERROR HANDLING

Bit errors may be introduced by both Snd (write errors) and Rcv (read errors). Depending on the technique used to communicate, errors may be injected in M for different reasons and may be detected and dealt with in different ways. A write error occurs when a data packet requested by Snd is not added to Rt's cache or PIT. A read errors occurs as a result of an incorrect retrieval of a message bit after it has been correctly written, and before it is expired.

Delay-Based: Cache. We consider the following two issues as common causes for write errors:

1. Packet loss (either interests or data packets). Interests from Snd may be dropped along their way to Pr. Similarly, data packets from Pr may be dropped before they reach Rt. In both cases, no data packets added to Rt's cache, and therefore the send operation fails. This, however, can be detected by Snd, who simply re-issues interests for which it does not receive data packets.
2. Forwarded data packets not added to Rt's cache. This can be caused, for example, by meta-cache algorithms

[3] Common prefix can be followed by different children namespaces, e.g., `/common/prefix/foo/C0` and `/common/prefix/yet /another/prefix/C1`.

on Rt. Snd can detect this only by re-requesting all bits set to 1 in its messages and, for each comparing the RTT of the first request with the RTT of the second.

We identify the following causes for read errors:

1. RTT fluctuations. Since retrieving a message relies on correctly identifying cache hits and misses, any overlap in the RTT between Rcv and Rt and between Rcv and Pr could cause a read error. These errors are not detectable, and cannot be addressed by simply re-sending interests.
2. Interests from other consumers. Some consumers may request a data packet that correspond to a bit in the message set to 0, and have it added to Rt's cache. We assume that this happens with negligible probability, since Snd and Rcv exchange messages using a set of data packets that are not popular.
3. Packet loss (data packets). If a data packet is dropped on the path from Pr to Rt, it can be safely be re-requested by Rcv without altering the original message. However, if it is dropped on its way from Rt to Rcv, the corresponding message bit will be set to 1 regardless of its original value. Rcv can only distinguish between the two cases – and determine the correct value of the corresponding message bit b – when b is read as 0.
4. Packet loss (interests). When interests are dropped on their way from Rcv to Rt (if the corresponding data packet is in Rt's cache) or to Pr (if it is not), Rcv cannot retrieve the corresponding bit. In this case, Rcv can re-issue the same interest without altering the original message, since no data packets have been added to Rt's cache. However, since loss of interest cannot be distinguished from loss of data packet, Rcv may not be able to recover from this error.
5. Rt is rebooted. This causes all data packets in Rt's cache to be deleted, therefore "erasing" all messages from Snd. This can be detected if Rcv knows that $M \neq 0^n$.

Rcv can reduce errors induced by RTT fluctuations using the "scope" field in interests, when Rt is its first-hop router. This field works similarly to the IP TTL field. When scope is set to 2, interests are forwarded for up to one hop. (Values higher than 2 are not allowed [6]). If the Rcv's first hop cannot satisfy the interests, it simply drops it. This way, Rcv does not need to measure any difference in the delay of cache hits and misses, since only cache hits will result in returned content. Moreover, this would allow interest retransmission in case of packet loss, since setting scope to 2 would prevent Rcv's interests from adding any new content into the cache. We argue that, however, setting the scope field would make Rcv's activity easier to detect.

Delay-Based: PIT. As in to the previous technique, write errors correspond to interests sent by Snd and are not added to Rt's PIT. The main cause for write errors is loss of the interest from Snd to Rt. This cannot be detected on time by Snd, since the same interest must be issued by Rcv before the corresponding data packet is received by Snd.

On the receiver side, errors may have the following causes:

1. RTT fluctuations. Similarly to the previous technique, significant fluctuations of RTT can introduce read errors.
2. Packet loss (either interests or data packets). In case of packet loss, Rcv will learn no information about the corresponding bit in the covert message. Moreover, re-transmitting an interest may provide no useful information, since by then the PIT entry corresponding to the original interest from Snd, if any, will be either expired or removed.
3. Interests from other consumers. Other consumers may issue the same interests that Snd and Rcv are using to covertly exchange information. However, this happens with negligible probability, because: (1) data packets used to covertly publish messages are non-popular, and (2) interests from other consumers must be issues a few milliseconds before Rcv issues its interests.
4. Lack of synchronization between Snd and Rcv. Depending on the topology, Snd and Rcv must be tightly synchronized, i.e., roughly within half RTT between Snd and Pr. Lack of synchronization may lead to a high rate of read errors.
5. Message expiration. Even though this technically is not a read error, it may happen that Rcv cannot retrieve part of the message on time due to the strict timing requirements.

As before, the scope field can be set in Rcv's interest to reduce error rate.

TDP. Write errors have the same causes, as well as detectability, as the write errors in delay-based cache technique.

Similarly, read errors have the same causes as with delay-based, single-bit cache. However, data packet-pairs provide more robustness against RTT fluctuations and packet loss. Since two subsequent RTTs – one corresponding to a cache hit, and one for a cache miss – are measured for each message bit, the probability of error associated with random RTT fluctuations is greatly reduced. With respect to packet loss, at least one of the data packets corresponding to a single message bit will be returned with relatively high probability. The associated RTT will still allow Rcv to estimate whether it is coming from Rt's cache – although less accurately.

Common-prefix-based Covert Communication. Using this technique, write errors may be introduced by the same events that trigger packet loss in delay-based, single bit cache. With respect to read errors, this technique is significantly more robust than the previous ones because: (1) it does not rely on timing measurements, and is therefore immune to RTT fluctuations; and (2) in case of packet loss (affecting either interests or data packets), Rcv can simply re-issue its interest, without affecting the covert message. Read errors can, however, be introduced by interests from other consumers, when they request content from the namespaces used by Snd and Rcv.

6.1 Error Correction

To address potential read/write errors, Snd can use error-correction codes with CEM. For example, Reed-Solomon error correction codes [22] could be used. We do not investigate this any further, since the goal of this paper is to assess feasibility of the channel and the corresponding error rate.

7. EVALUATION

We implemented a prototype CEC system to evaluate our protocols. In this section we present the results of our experiments. The prototype is based on CCNx [5], an open-source implementation of NDN which runs as an overlay on top of IP. We performed experiments on the two topologies:

- LAN, composed of Snd, Rcv, Rt and Pr within the same broadcast domain. Each party runs a separate instance of CCNx.
- NDN testbed [20], where Snd and Rcv (located in Europe) are connected to the UCLA NDN hub (which acts as Rt), and Pr is connected to the testbed through the UCI hub. UCLA and UCI hubs are one NDN hop apart (ten hops over IP).

Snd and Rcv exchange 1,000-bit messages. Each message is a fresh random bit string. This is representative of the distribution of encrypted messages.

Naturally, our protocols generate communication overhead. We used 41-byte interests and 377-byte data packets (on average). With single-bit transmission (either using PIT and cache), each message bit set to 1 requires Snd to exchange 418 bytes. Regardless of message content, Rcv needs to send/receive 418 bytes per message bit. With the TDP protocol, each message bit costs 418 bytes to Snd and 836 bytes to Rcv. When transmitting multiple bits with a single interest, m message bits cost Snd 418 bytes, and $2^m \cdot 418$ bytes to Rcv. Finally, with CPC both Snd and Rcv exchange 418 bytes for each m-bit word.

In our experiments, Snd can send messages at a rate different from the rate at which Rcv receives them. This is possible due to the state in routers (i.e., cache or PIT, depending on the technique used).

7.1 Evaluation of Delay-Based (Cache) Techniques

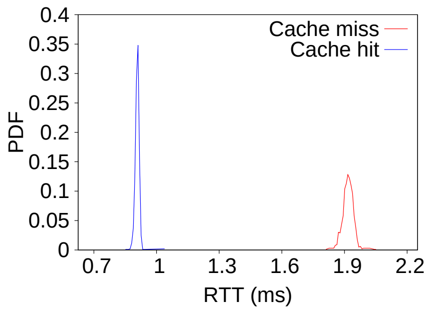
In order to assess feasibility of cache-based techniques, we compared RTT associated with cache hits and cache misses in both LAN and testbed scenarios.

Figure 3 summarizes our findings and represents average values over 100,000 data packets. While there is virtually no overlap between RTT of cache hits and misses in a controlled (LAN) environment, RTT fluctuations on the testbed do not always allow us to distinguish a cache hit from a cache miss. However, the overlap is still relatively small and, as confirmed by further experiments, it is possible to implement a reliable CEC on the testbed.

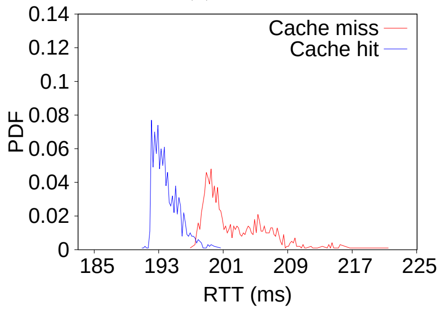
We then looked into how interest sending rate affects RTT. We selected values for t varying from $t_{min} = 0.3$ μs to $t = 5$ ms (see Section 4.4). We performed several experiments, each using 100,000 data packets. Before each experiment, we restarted Rt in order to remove all cache entries. Results are reported in Figure 4.

In LAN (figures 4a, 4b, and 4c), RTTs of cache hits and cache misses are clearly separated, regardless of t. On the testbed (figures 4d, 4e, and 4f), for small values of t, cache hits and misses significantly overlap for messages longer than 200 bits. This suggests that short busts, separated by short pauses, provide lower error rates.

For cache-based CEC, we evaluated read and write errors separately, while varying t and t_{thresh}. To evaluate write errors, Snd published of 100,000 covert bits for each value of t. Covert bits were subsequently requested at a low rate (t = 100 ms) by Rcv. We then estimated how many data packets were not retrieved from cache. Figure 5 summarizes our findings. In this experiment, Rcv introduces a small measurement error. We estimate to be negligible in LAN, and below 1.5% on the testbed. With cache-based CEC,



(a) LAN



(b) Testbed

Figure 3: PDF for cache hit and cache miss.

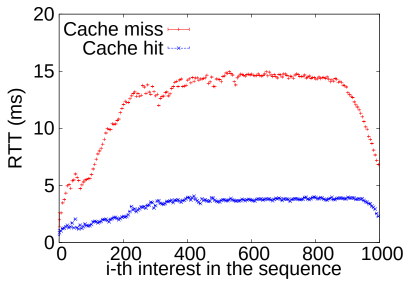
write errors can be completely eliminated if Snd re-issues interests for content that it did not receive; although, writing time increases.

To measure read errors, Snd published 100,000 covert bits, separated in groups of 1,000- bit CEMs, for each value of t and t_{thresh}. Results of this experiment are shown in Figure 6. Due to the clear separation between RTTs associated with cache hits and misses in LAN, read errors were very low for a wide range of parameters (e.g., for t_{thresh} between 1 and 1.5 ms). On the testbed, error was typically between 3% and 5% for t_{thresh} between 191 and 193 ms.

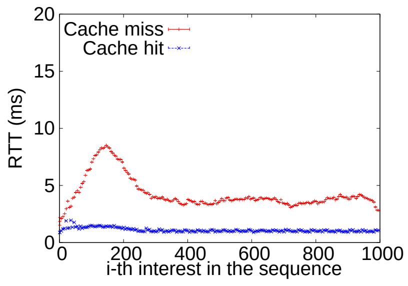
7.2 Evaluation of Delay-Base (PIT) Techniques

We requested the same data packet from both Snd and Rcv at very close intervals (i.e., 0.8 and 1 ms in LAN and 2 ms on testbed), in order to trigger interest collapsing on Rt, and, therefore, a PIT hit. Snd and Rcv were synchronized using a local NTP server; we estimated the time difference between the two hosts to be below 0.2 ms. Our experiments show that is possible to distinguish PIT hits from misses using appropriate intervals between interests from Snd and Rcv. Results of this experiment are shown in Figure 7. However, the separation is less clear than with cache, as shown in the same figure. Moreover, this channel requires much tighter synchronization between Snd and Rcv (i.e., sub-millisecond in LAN, and within 2 ms on testbed). For these reasons, PIT-based CEC are significantly more difficult to implement.

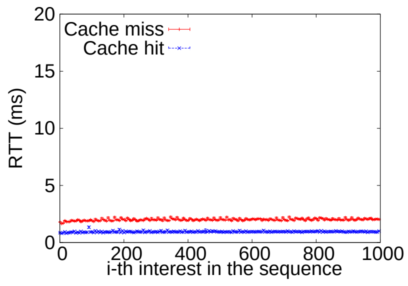
Since Snd and Rcv must operate synchronously and with the same t, we measured read and write errors jointly. For this experiment, the delay between interests from Snd and Rcv is 0.8 ms in LAN, and 8 ms on the testbed. Results are shown in Figure 8. With appropriate choice of the threshold parameter, errors in LAN are negligible, and below 7.5% in



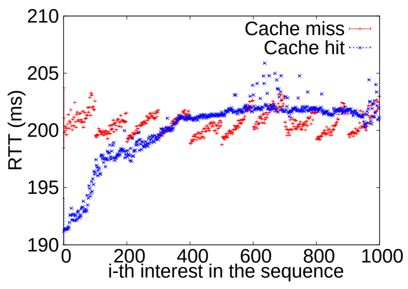
(a) LAN, $t = t_{min}$



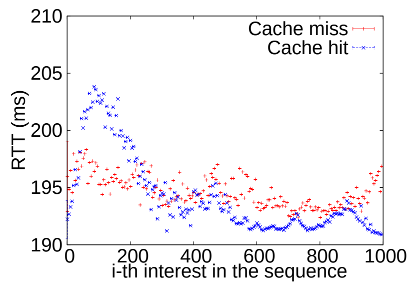
(b) LAN, $t = 0.2$ ms



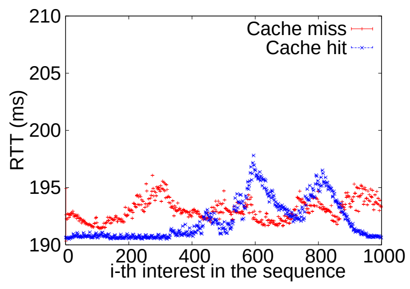
(c) LAN, $t = 0.5$ ms



(d) Testbed, $t = t_{min}$



(e) Testbed, $t = 0.2$ ms



(f) Testbed, $t = 0.5$ ms

Figure 4: RTT for data packets, varying request rate.

the testbed.

7.3 TDP Evaluation

We measured the error rate varying write and read speeds separately for Snd and Rcv. Figures 9 and 10 summarize our findings. On the receiver side, this technique performs better than the cache-hit-based one. For example, for $t = 1.5$ ms in the testbed, the error for TPD is less than 2% (see Figure 10b), while for $t = 3$ (i.e., the same effective bit rate relative to the CEM) in the cache-hit-based technique the error for is more than 4% (Figure 5b).

7.4 Evaluation of Common-Prefix-Based Technique

We set $m = 1$ (i.e., each data packet encodes one bit), in order to encode 1,000-bit CEM using 1,000 data packets. We run separate experiments to evaluate Snd and Rcv errors. As mentioned in Section 6, both parties can avoid packet-loss-induced errors using interest retransmission. For a fair comparison with previous protocols, we test how the common-prefix-based technique performs *without* retransmissions.

Results on write errors, both in our LAN and on the testbed, are identical to those in Figure 10. In fact, Snd performs the same actions to send a CEM. Read errors on the testbed are reported in Figure 11. We omit the plot corresponding to read errors in LAN, since for all tested values of t error rate was below 0.03%. Errors for both Snd and Rcv are due to packet loss.

7.5 Bit Rate and Error Comparison

To simplify comparison of techniques introduced in this paper, we combine effective bit rate and corresponding error for all our protocols in Figure 12. Note that, for TDP, Snd's effective bit rate can be *multiplied* by an arbitrary m, while Rcv's bit rate should be *divided* by 2^m. Analogously, the bit rate for both Snd and Rcv in the common-prefix protocol should be multiplied by m as discussed in Section 5.

8. SECURITY ANALYSIS

We now analyze security of CEC techniques. We start by showing that proposed protocols are retroactively private and secure against message recovery attack. We then conclude with an informal discussion on the detectability and robustness of our approaches.

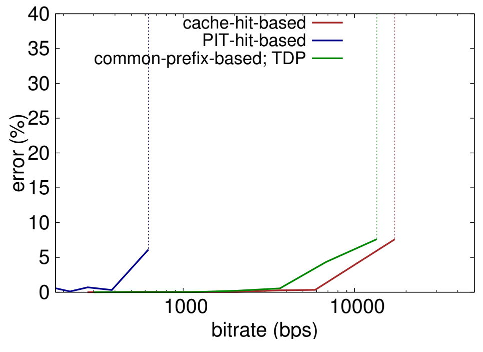
8.1 Retroactive Privacy

Adv has non-negligible advantage over 1/2 in the retroactive privacy game (see Section 3) only if it can infer information about a from interaction with Snd, Rcv and Rt *after* the message M_a has expired. That is, Adv can only interact with protocol participants after data packets used to encode M_a have been removed from Rt's PIT and from all caches.

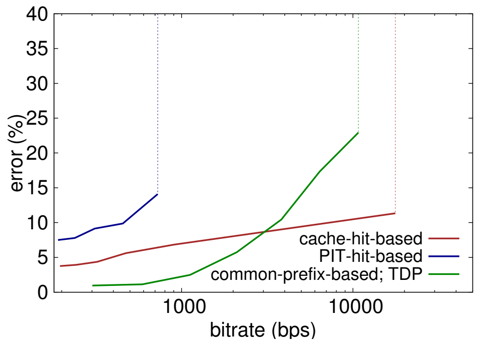
Since Snd and Rcv delete M_a as soon as they (respectively) send and receive it, Adv cannot acquire information about M_a by compromising the two parties. Similarly, NDN routers do not keep track of data packets once they disappear from both PIT and cache. Therefore, after M_a expires, Rt carries no information about the message. As a result, there is simply no information about M_a within the network after the message expires.

8.2 Security Against Message-Recovery Attacks

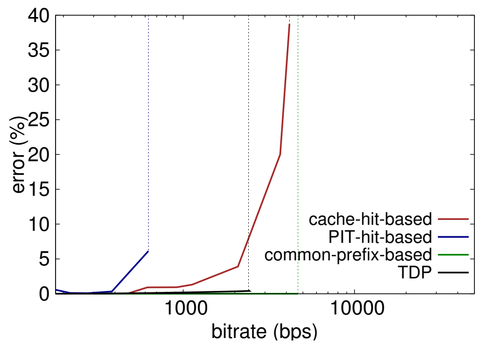
In order to reconstruct a CEM, Adv can probe all NDN routers, and try to identify data packets used for covert communication. However, this approach has two problems: (1) there is no data packet in routers caches for a bit set to 0; therefore, Adv cannot learn information about these bits by



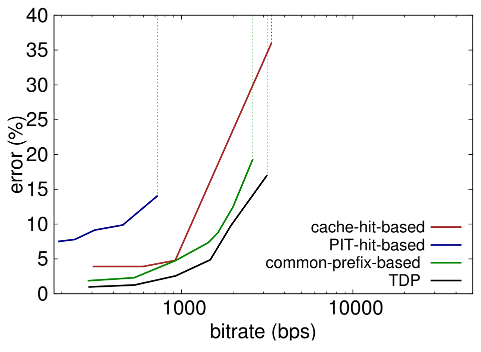
(a) LAN (Snd)



(b) Testbed (Snd)



(c) LAN (Rcv)



(d) Testbed (Rcv)

Figure 12: Performance comparison.

simply observing routers caches. (2) even for a relatively small NDN deployment, the number of routers and the size of their caches makes this attack infeasible.

Another adversarial strategy consists in infiltration of the routing infrastructure: Adv could mount a Sybil attack [28], deploying a large number of malicious NDN routers. We believe that this approach is not feasible, since: (1) Adv cannot deploy an arbitrary number of NDN routers. Even if NDN is implemented as an overlay, routers are identified by their unique IP address. This would force Adv to obtain a very large number of public IP address. (2) Even if the adversary succeeds deploying a large number of routers, it must log all data packets forwarded by all controlled routers. This may not be feasible. (3) Similarly, even if Adv can compromise arbitrary routers, maintaining logs for all forwarded data packets would not be viable.

8.3 Detectability

In order to exchange a message through our protocols, Snd and Rcv do not need to communicate directly, nor they need to be connected through the same NDN router. Moreover, they only interact with the network as prescribed by NDN specifications.

A single-bit message $b = 0$ sent using single-bit transmission via cache or PIT cannot be detected, since Snd performs no action. When $b = 1$, Snd retrieves a non-popular data packet. We believe that, in practice, by flagging all single interests for non-popular data packets as “suspicious”, Adv would incur in an overwhelmingly large number of false alarms. Similarly, a single interest issued by Rcv to retrieve b would be easily hidden by the existing traffic.

When Snd and Rcv exchange messages longer than a single bit, however, their actions become more detectable. In particular, the longer the message, the more likely it is for Adv to correctly identify a CEM between two or more parties. While a single interest for non-popular data packets may not raise any suspect, a long streak of interests for non-popular data packets may be easy to notice. For this reason, Snd and Rcv should limit the size of the exchanged messages to reduce detectability.

Finally, with namespace-based covert communication detectability mostly depends on m and on the size of the covert. In particular, a higher value for m implies lower detectability: less data packets have to be requested to write and read a covert message.

8.4 Robustness

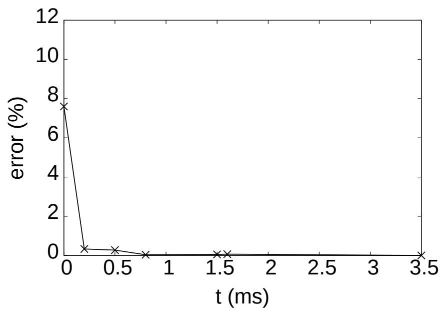
When Rt introduces arbitrary delays to conceal cache hits, our techniques based on measuring time difference between these two events do not work. However, techniques based on PIT and on common prefixes are not affected by cache hit delays, since they either do not rely on cache or do not consider RTT.

Similarly, when the network introduces unpredictable delays on packets (e.g., when traffic intensity has sudden wide fluctuations), common-prefix-based technique may be more appropriate since it does not rely on timing measurements.

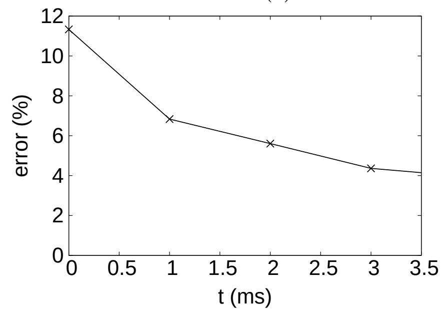
9. RELATED WORK

We divide relevant related work in two classes: *covert communication* and *ephemeral communication*.

Covert Communication. The goal of a covert channel is to conceal the very existence of a covert message by communicating it through legitimate channels [16].

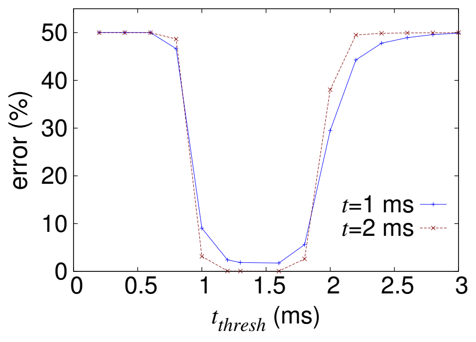


(a) LAN

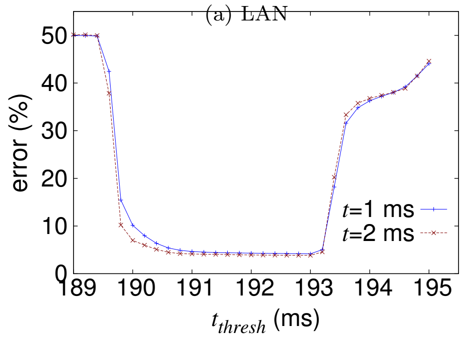


(b) Testbed

Figure 5: Cache-hit-based communication: write error, varying t.



(a) LAN



(b) Testbed

Figure 6: Cache-hit-based communication: read error varying t_{thresh} and t.

In [24], Shah et al. present Jitterbug, a hardware device and a communication protocol that covertly transmit data by perturbing the timing of keyboard events. In particular, the authors design and implement a small hardware *pass-through* device that introduces small – although, measurable – variations in the times at which keyboard events are delivered to the host. When the user runs an interactive communication protocol (e.g., SSH, instant messaging), a receiver monitoring the host's network traffic can recover the leaked data. According to the experimental results reported in [24], the bandwidth offered by Jitterbug is roughly 500 bps over 14 network hops, with 8.9% error rate. In contrast, our technique provide a bit rate of about 15,000 bps in a similar scenario with analogous error rate. Another difference is that with Jitterbug the receiver must be able to intercept network traffic, while our approach can be used by any unprivileged user.

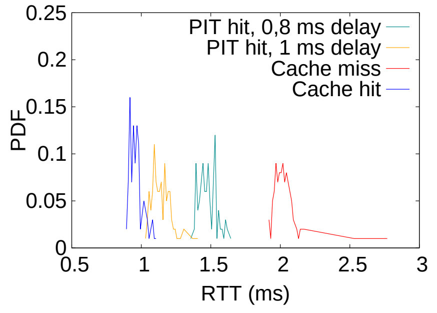
CoCo, introduced in [16] by Houmansadr et al., is a framework for establishing covert channels via inter-packet delays. The sender generates a traffic flow directed to the receiver, then manipulates the flow according to the covert message and a key, shared between the two parties. The coding algorithm used in CoCo ensures robustness of the covert message to perturbations. The authors show statistical evidence on the undetectability of the communication channel. We emphasize that CoCo would not satisfy our requirements because sender and receiver must communicate directly.

Murdoch et al. [18] investigate covert channel implemented by embedding information in random-looking TCP fields. They show that naïve approaches – such as embedding ciphertext in the initial sequence number (ISN) field – can be easily detected. Then, they discuss how to implement networking stack-specific covert channel, which are provably undetectable. Similarly to CoCo, the main difference between our work and the work of Murdoch et al. is that sender and receiver must exchange packets directly.

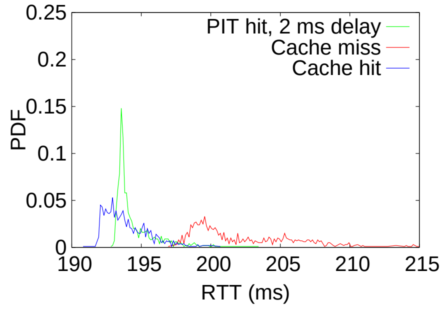
Ephemeral Communication. Geambasu et al. introduced the Vanish system [14], which allows users to publish ephemeral messages. Users encrypt their messages using a random symmetric key. Then, they publish shares of the key (computed using Shamir secret sharing [23]) in random indices in a large, pre-existing distributed hash table (DHT). A DHT is a distributed data structure that holds key-value pairs. Since data on DHTs is automatically deleted over time, shares of the key automatically "disappear". Once enough shares have been deleted, the key – and therefore the encrypted message – is effectively erased.

Wolchok et al. [28] showed that Vanish can be defeated using low-cost Sybil attacks on the DHT. In particular, they exploited one of the design flaws of Vanish, namely the assumption that DHTs are resistant to crawling. This is in contrast with our approach, where monitoring all routers' caches is clearly infeasible. Although the authors of Vanish have since proposed countermeasures [13], these techniques only slightly raise the bar against existing attacks [4].

Castelluccia et al. [4] introduced EphPub, a DNS-based ephemeral communication technique. A publishers encrypts and distributes a message. Then, it distributes the decryption key as follows: for each key bit set to 1, the publisher picks a DNS resolver and uses it to answer a recursive DNS queries for a specific domain. Since DNS resolvers cache responses for a pre-determined amount of time, one or more receivers can subsequently issue *non-recoursive* queries to

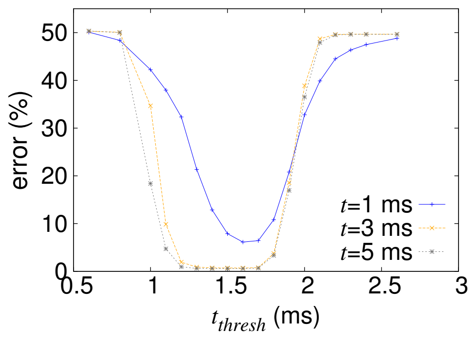


(a) LAN

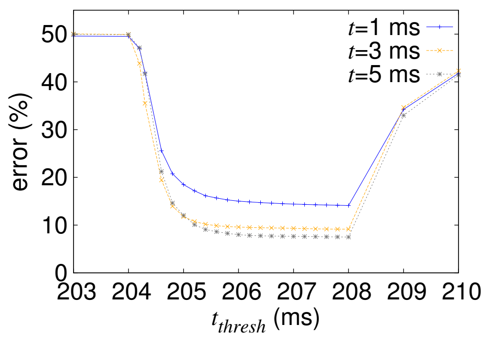


(b) Testbed

Figure 7: RTT for data packets causing PIT collisions.



(a) LAN



(b) Testbed

Figure 8: Joint write and read error varying t in our PIT hit-based protocol.

the same resolver. These queries will be answered only if the corresponding domain-IP pair is in cache. Once enough cache entries expire (or get overwritten), the decryption key – and therefore the published message – disappears.

There are several differences between EphPub and our techniques. First, while EphPub relies on an application-layer service (DNS resolver) to publish an ephemeral piece of data, our techniques leverage routers' PITs and caches, which are part of the routing architecture. Moreover, while EphPub can be blocked by forcing users to use a local DNS server with no cache (e.g., by filtering out DNS queries at the network gateway), our PIT-based technique allows two parties to exchange CEMs even if routers do not provide content caching. Moreover, if EphPub sees wide adoption, there are several concerns (raised also by Castelluccia et al. in [4]) that would impose excessive load on DNS servers, which would then be forced to stop acting as "open" resolvers. In contrast, with our approach, communicating parties do not impose higher-than-usual load on routers: consumers simply use their allocated bandwidth for content retrieval. Furthermore, routers cannot determine the source of data requests (interests do not carry a source address), and therefore always operate similarly to open resolvers. Finally, EphPub does not provides covert communication, since the behavior of two users who communicate via EphPub is difficult to conceal. In fact, "regular" users rarely query multiple remote DNS servers in short bursts. With our techniques, instead, Snd and Rcv do not perform any easily identifiable activity.

Perlman [21] proposed Ephemerizer, a centralized approach to secure data deletion. The goal of Ephemerizer is to find a balance between data availability and the ability to properly delete data. Users encrypt their data using a symmetric encryption scheme. Then they delegate key storage to a trusted third party. This third party destroys cryptographic keys when they "expire", effectively making the original data unaccessible. Compared to [14], [4], as well as to our approach, Ephemerizer requires an always on-line, trusted third party.

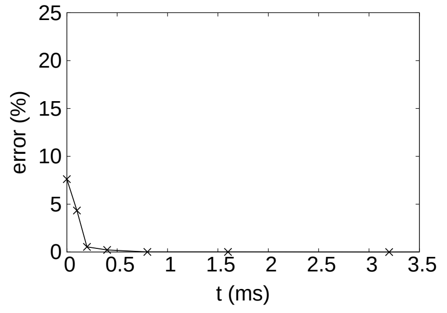
10. CONCLUSIONS

In this paper, we have presented the first evaluation of covert ephemeral communication in NDN. Our techniques do not require Snd and Rcv to exchange any packet directly. Rather, they rely on user-driven state on routers to publish and retrieve covert messages. Messages published with our approach are ephemeral, i.e., they are automatically deleted from the network after a certain amount of time, without requiring any action from Snd or Rcv. Additionally, our delay-based techniques, messages *expire* immediately after being retrieved.

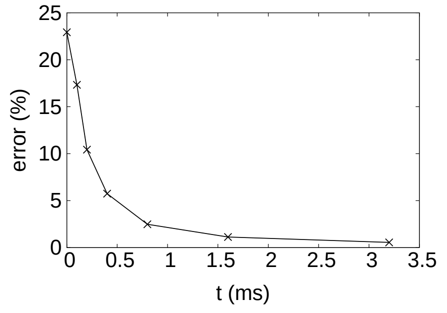
Our techniques are based on fundamental components on NDN, and do not require "abuse" of application-layer protocols. In practice Snd and Rcv only need access to non-popular content.

We performed experiments on a prototype implementation of our protocols. In particular, we measured the the bandwidth and robustness of our approaches on a local (LAN) setup and in a geographically distributed environment – the official NDN testbed. Our experiments confirm that the techniques proposed in this paper provide high bandwidth and low error rate.

11. ACKNOWLEDGEMENTS

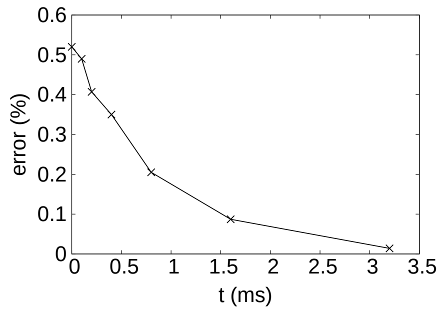


(a) LAN

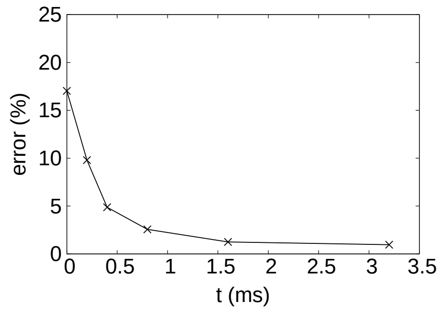


(b) Testbed

Figure 9: Write error with TDP, varying Snd's t.



(a) LAN



(b) Testbed

Figure 10: Read error with TDP, varying Rcv's t.

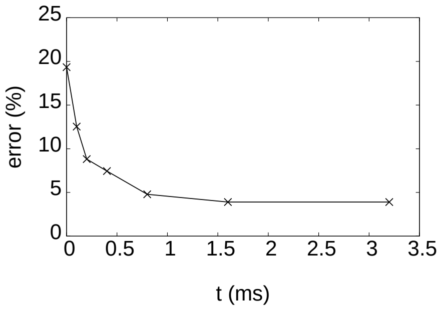


Figure 11: Common-prefix-based protocol: read error varying t.

We would like to thank Christos Papadopoulos, Steve DiBenedetto, Jeff Burke and Alex Horn for providing access to the NDN routers hosted at their respective institutions.

12. REFERENCES

[1] G. Acs, M. Conti, P. Gasti, C. Ghali, and G. Tsudik. Cache privacy in named-data networking. In *the 33rd International Conference on Distributed Computing Systems (ICDCS)*, 2013.

[2] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang. Interest flooding attack and countermeasures in Named Data Networking. In *IFIP Networking*, 2013.

[3] Akamai. `http://www.akamai.com`.

[4] C. Castelluccia, E. D. Cristofaro, A. Francillon, and M. A. Kâafar. Ephpub: Toward robust ephemeral publishing. In *ICNP*, pages 165–175, 2011.

[5] Content centric networking (CCNx) project. `http://www.ccnx.org`.

[6] CCNx Interest Message. `http://www.ccnx.org/releases/latest/doc/technical/InterestMessage.html`.

[7] CCNx Node Model. `http://www.ccnx.org/releases/latest/doc/technical/CCNxProtocol.html`.

[8] A. Compagno, M. Conti, P. Gasti, and G. Tsudik. Poseidon: Mitigating interest flooding DDoS attacks in Named Data Networking. In *LCN*, 2013.

[9] M. Conti, P. Gasti, and M. Teoli. A lightweight mechanism for detection of cache pollution attacks in Named Data Networking. In *COMNET*, 2013.

[10] S. DiBenedetto, P. Gasti, G. Tsudik, and E. Uzun. Andana: Anonymous named data networking application. In *NDSS*, 2012.

[11] Facebook. `http://www.facebook.com`.

[12] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang. DoS & DDoS in named-data networking. In *ICCCN*, 2013.

[13] R. Geambasu, J. Falkner, P. Gardner, T. Kohno, and K. Krishnamurthy. Experiences building security applications on dhts. Technical report, UW-CSE-09-09-01, University of Washington, 2009.

[14] R. Geambasu, T. Kohno, A. A. Levy, and H. M. Levy. Vanish: Increasing data privacy with self-destructing data. In *USENIX Security Symposium*, pages 299–316, 2009.

[15] Google global cache. `https://peering.google.com/about/ggc.html`.

[16] A. Houmansadr and N. Borisov. CoCo: Coding-Based Covert Timing Channels for Network Flows. In *The* 13th *Information Hiding Conference (IH)*, 2011.

[17] Apple itunes. `http://itunes.apple.com`.

[18] S. J. Murdoch and S. Lewis. Embedding covert channels into tcp/ip. In *Information Hiding: 7th International*

Workshop, volume 3727 of LNCS, pages 247–261. Springer, 2005.

[19] Named Data Networking project (NDN). `http://named-data.org`.

[20] NDN Testbed. `http://www.named-data.net/testbed.html`.

[21] R. Perlman and R. Perlman. The ephemerizer: Making data disappear. *Journal of Information System Security*, 1:51–68, 2005.

[22] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8:300–304, 1960.

[23] R. L. Rivest, A. Shamir, and Y. Tauman. How to share a secret. *Communications of the ACM*, 22(22):612–613, 1979.

[24] G. Shah, A. Molina, and M. Blaze. Keyboards and covert channels. In *In Proceedings of the 2006 USENIX Security Symposium (JulyâĂŞAugust*, pages 59–75, 2006.

[25] Gps clock synchronization. `http://www.spectracomcorp.com/Solutions/Applications/GPSClockSynchronization/tabid/100/Default.aspx`.

[26] Google serves 25 percent of North American Internet traffic. `http://www.wired.com/wiredenterprise/2013/07/google-internet-traffic/`.

[27] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp. Backscatter from the data plane – threats to stability and security in information-centric networking. *Computer Networks*, DOI 10.1016/j.comnet.2013.07.009, 2013.

[28] S. Wolchok, O. S. Hofmann, N. Heninger, E. W. Felten, J. A. Halderman, C. J. Rossbach, B. Waters, and E. Witchel. Defeating vanish with low-cost sybil attacks against large dhts. In *NDSS*, 2010.

[29] M. Xie, I. Widjaja, and H. Wang. Enhancing cache robustness for content-centric networks. In *INFOCOM*, 2012.

[30] Youtube. `http://www.youtube.com`.