Department of Communications and Networking

Best-Effort Authentication for Opportunistic Networks

John Solis, Philip Ginzboorg, N. Asokan, Jörg Ott

Aalto University

SCIENCE + TECHNOLOGY

WORKING PAPERS

Best-Effort Authentication for Opportunistic Networks

John Solis
Sandia National Labs
Livermore CA, USA
jhsolis@sandia.gov

Philip Ginzboorg, N. Asokan
Nokia Research Center
Helsinki, Finland
{*philip.ginzboorg,n.asokan*}*@nokia.com*

Jörg Ott
Aalto University
School of Science and Technology
Espoo, Finland
jo@netlab.tkk.fi

***Abstract*—Prior work has shown that uncontrolled messaging in ad-hoc opportunistic networks results in a disproportionate sharing of network capacity. Solutions based on publicly verifiable sender authentication mechanisms, such as those in [1], require complete message transmission for proper verification. However, introducing message fragmentation, e.g., to optimize limited transmission opportunities, could negatively impact resource management. Unverifiable message fragments that are assigned lower priorities are discarded sooner and reduce delivery ratios. This paper investigates the extent of this impact and shows that publicly verifiable fragment authentication mechanisms are needed to retain the benefits of fragmentation as well as resource management.**

We take this a step further and show through simulations that strong authenticity guarantees for intermediaries are unnecessary in our specific scenario. *Best-effort* authentication schemes, where the probability of false positives is much higher than in typical authentication schemes, are sufficient. This suggests the existence of other scenarios where this approach is also applicable. We describe various best-effort authentication techniques and analyze their computation and space savings.

***Keywords*-best-effort security; opportunistic networks; fragment authentication**

I. Introduction and Motivation

Uncontrolled messaging by *resource hogs* results in disproportionate allocation of the network resources required to serve those nodes. Even a small number of hogs greatly reduces message delivery ratios for the remaining users [1]. However, this can be controlled through resource management schemes: Nodes group themselves into cooperative domains that prioritize incoming messages based on sender *authentication*. Messages sourced from fellow domain members are given higher priority [1].

Research has shown that message fragmentation can improve delivery ratios in opportunistic networks through better utilization of available contact opportunities [2]. Fragmentation in the presence of resource hogs implies the need for publicly verifiable *fragment authentication*. Without fragment authentication, intermediaries assign fragments (even from a valid a domain member) the lowest priority. The end result is lower delivery ratios.

Fragment authentication has not been widely addressed in opportunistic networks (see [3]). Very strong authenticity guarantees — i.e. limiting false positive (impersonation) probabilities to less than 2^{-80} [4] — may not be needed in our application scenario. We thus began with the conjecture that a small number of false positives will not significantly impact overall delivery ratios of domain members.

We investigate *best-effort* authentication where the probability of false positives is allowed to be much higher than in typical authentication schemes. This allows a trade-off between degree of authenticity for reduced transmission and computation overhead. The caveat with applying best-effort authentication is that a technique fit for one scenario may be unfit for another.

Motivated by these concerns, we raise the following questions:

1) How seriously does a lack of fragment authentication impact the effectiveness of resource management techniques against resource hogs?
2) To what extent can the benefits of resource management techniques be retained with best-effort authentication?
3) What are the overhead costs incurred by full and best-effort fragment authentication schemes?

To investigate these questions we extended the environment simulated in [1] to support fragmentation. We confirmed that resource management schemes *are negatively impacted* when intermediaries are unable to verify fragmented messages. We also learned that our target network scenario can operate using best-effort authentication techniques with false positive rates as high as 60%. Although this parameter is ultimately under the control of the network operators, we assume 60% is a reasonable level for our particular scenario. Finally, we describe two additional techniques for implementing best-effort fragment authentication, analyze computational and space overhead, and present an informal security analysis.

II. Opportunistic Networks and Fragmentation

A. Fragmentation in brief

The DTN architecture RFC [5] describes two methods for message fragmentation in opportunistic networks: *proactive* and *reactive*. Proactive fragmentation occurs (only) at the source node prior to message transmission. *Reactive fragmentation* occurs immediately after a transmission link on the message path is interrupted. The receiving node

constructs a new partial message from all received data, while the sender retains the full message.

Our investigations focus on the second method due to its general applicability and exhibited performance gains in [2]. In particular, we focus on a variant of reactive fragmentation where messages are fragmented at sender-defined boundaries. We use the term *scrap* to refer to the smallest authenticatable unit implied by fragment authentication schemes, e.g., "toilet-paper" [6] or Merkle hash tree [7]. (Those schemes are described towards the end of Section II-C.) With this approach, new partial fragments are only created from *complete* scraps received.

B. Simulated Environments

The environment of [1] was extended to support fragmentation and best-effort authentication. Parameters were selected to simulate a mobile ad-hoc DTN scenario where people travel throughout city streets using their personal devices to exchange photographs and short videos.

- *Node Hardware:* Nodes communicate over 2 Mbit/s bi-directional wireless links with messages generated at a rate of one message per node per hour. Message source and destination are selected uniformly at random while the message size distribution varies with each mobility model (see below).
- *Routing Algorithm:* The binary mode version of Spray-and-Wait [8], with 10 message copies, was selected as the primary routing algorithm because it exhibited the highest delivery ratios while being most impacted by resource hogs. In binary mode, half of available message copies are transferred to the next hop until a single copy remains. At this point, nodes revert to direct-delivery mode.
- *Domain Settings:* Nodes are divided into two disjoint groups: those that belong to a co-operative domain and those that fall outside that domain. The former contains 90% of the total network while the outsider domain contains the remaining 10%. Domain members cooperate to increase their delivery ratios by prioritizing fellow domain-member messages. For instance, outsider messages are the first to be discarded when allocating new storage space. Outsiders are uncooperative and handle all messages with equal priority.

We investigated two synthetic mobility models and one real-world trace model corresponding to different contact patterns. Certain parameters, e.g., message size distribution and network load, varied between the models as follows:

- *Random Waypoint(RWP):* The simplest mobility model restricts movements to an area of 1km$\times$1km. Points are selected uniformly at random and nodes behave as described in the general synthetic parameters described in Section II-B.
- *Map-Based:* The map-based mobility model restricts movement to existing roads, highways, and pedestrian paths in the downtown area of Helsinki, Finland. Destination points are selected uniformly at random with a bias towards several points of interest, e.g., movie theaters and metro stations.

In the synthetic models above, a total of 250 nodes move at a speed selected uniformly at random between 0.5-1.5 m/s. Upon reaching the destination a node pauses for 0-120 seconds (randomly selected) before continuing. Message sizes are distributed uniformly at random between 500KB-5MB with a time-to-live of two hours. Message generation is restricted to the first ten (out of twelve) hours. This gives all messages an opportunity to traverse the network.

- *RollerNet Trace:* RollerNet traces are real-world contact patterns extracted from an experiment conducted by Tournoux, et al [9]. A total of 62 skaters in the Paris roller tour were given Bluetooth equipped iMotes [10] to record opportunistic sightings of neighboring Bluetooth devices.

Although over 1,000 unique devices were recorded, many were only seen in passing. Our simulations only consider the 517 devices seen more than once. We also mention that trace data only includes devices directly seen by the iMotes. It does not include contact opportunities between the other devices. Instead of inferring contacts, we use the trace data as provided and restrict message generation to the first 62 nodes. The remaining nodes act as gateways that receive and forward data.

The RollerNet tour lasted three hours and is much shorter than the 12 hour duration used in the synthetic traces. In order to see the same effects of fragmentation and resource hogs we changed message size range to 10MB-15MB and increased the storage buffer to 100MB.

C. Simulation Results

Impact of Fragmentation: The first set of simulations, summarized in Table I, show how fragmentation improves delivery ratios, when all nodes are honest. Those improvements, are due to better utilization of contact opportunities, and mirror the results in [2].

Mobility Model	Fragmentation[a]	
	No	Yes
Map-Based	.306 [.006]	.567 [.009]
RWP	.261 [.004]	.487 [.015]
RollerNet	.340 [.003]	.518 [.001]

[a]Standard deviation in [ ]

Table I
FRAGMENTATION IMPROVES MESSAGE DELIVERY RATIOS

Resource Management Schemes: We now investigate the effect of resource hogs and see if the coarse-grained resource management of [1] improves delivery ratios.

In coarse-grained resource management, nodes group themselves into domains and give priority to other domain

members. Nodes discard messages to make space for incoming messages only when the buffer is currently full. Domain members attempt to discard all low priority (non-domain) messages before high priority messages. If only high priority messages remain then the oldest message is discarded first.

Table II shows resource management schemes restoring delivery ratios – cooperating domain members protect their messages from resource hogs with high messaging rates.

Mobility Model	No hogs	10% hogs	
		Resource Management	
		No	Yes
Map-Based	.306 [.006]	.262 [.005]	.288 [.010]
RWP	.261 [.004]	.230 [.006]	.251 [.004]
RollerNet	.340 [.003]	.302 [.023]	.329 [.014]

Table II
RESOURCE MANAGEMENT COUNTERS RESOURCE HOGS AND IMPROVES DELIVERY RATIOS

Fragment Authentication and Resource Management: We now investigate the effect of fragmentation when combined with resource management schemes based on full message authentication. Table III shows that delivery ratios are similar when hogs are not present. There is a slight improvement, however, in delivery ratios from Table I. This is a result of resource management: fragments of domain members and all messages (fragmented or not) from outsiders are assigned lower priority by intermediaries. The increased delivery ratio for domain members is at the expense of a decreased ratio for outsiders.

When resource hogs are introduced the increased congestion causes more of the domain member's fragments to be discarded; and the delivery ratio r drops in all mobility models.

We conclude that a lack of fragment authentication severely reduces the effectiveness of resource management schemes. Still, when we compare Table I and Table III, we see r is better in this scenario than when there is no fragmentation at all.

Mobility Model	No Hogs	10% Hogs
Map-Based	.575 [.013]	.397 [.006]
RWP	.500 [.004]	.319 [.004]
RollerNet	.569 [.001]	.452 [.014]

Table III
FRAGMENTATION COMBINED WITH RESOURCE MANAGEMENT LACKING FRAGMENT AUTHENTICATION IS HARMFUL WITH 10% HOGS

If we add fragment authentication to the resource management schemes we can restore delivery ratios to the baseline case when no resource hogs are present (Table IV, right data column). This is a direct result of intermediaries once again correctly prioritizing messages from domain members. We conclude that fragmentation must be coupled with fragment authentication. This approach optimizes transmission opportunities while simultaneously protecting network resources.

Mobility Model	10% hogs	
	Fragment Authentication	
	No	Yes
Map-Based	.397 [.006]	.468 [.009]
RWP	.319 [.004]	.373 [.006]
RollerNet	.452 [.014]	.440 [.020]

Table IV
ADDING FRAGMENT AUTHENTICATION RESTORES MESSAGE DELIVERY RATIOS

In all of the above scenarios we modeled fragment authentication with strong, i.e., cryptographic, security guarantees. But strong security guarantees may not be required for our specific application scenario. It is possible that authentication mechanisms, with weaker levels of security, are sufficient and can retain the benefits gained by resource management techniques. In the next section we extend our initial results by investigating best-effort authentication.

III. BEST-EFFORT AUTHENTICATION

Intuitively, a best-effort fragment authentication scheme guarantees authenticity of a fragment with a certain probability: valid fragments are guaranteed to be verified (i.e., no false negatives). An invalid fragment may, however, be verified with probability p (resulting in a false positive). We reason that dropping a valid fragment is more harmful than carrying an invalid fragment. The prior case results in being unable to reconstruct a message while the latter increases reconstruction effort for the destination node; that node must determine which fragments are valid. In essence, a few false positives are acceptable, while false negatives are not.

We emphasize here that best-effort authentication is only intended for resource management in *intermediaries*. It does *not* replace traditional end-to-end message integrity schemes. Best-effort authentication, when used in conjunction with a traditional end-to-end integrity scheme, does not affect original message integrity.

A. Best-Effort Resource Management

We now investigate the effect of best-effort fragment authentication in restoring delivery ratios in the presence of resource hogs. To understand the effectiveness of the approach, we focus on normalized delivery ratios ($\hat{r}$). These ratios are normalized with respect to the case where fragmentation is used but no hogs are present (right data column of Table I).

Fragment authentication is modeled as a generic algorithm that accepts invalid fragments with a certain (false-positive) probability rate p varied in increments of 0.2 between zero and one. We assume that p is a fixed system-wide parameter and comment that our goal is to study the effects of best-effort authentication. Although other strategies are possible,

e.g., p values that vary per node, we have not studied the implications of such approaches.

Figure 1 shows the delivery ratios against varying p values for the random waypoint model, map-based mobility model, and RollerNet traces. In each case we assume that 10% of the nodes are resource hogs and keep the remaining parameters unchanged. Note that $p = 0$ corresponds to using strong-authentication and can also be mapped directly to the values in the right column of Table IV. This difference illustrates the effectiveness of resource management schemes.

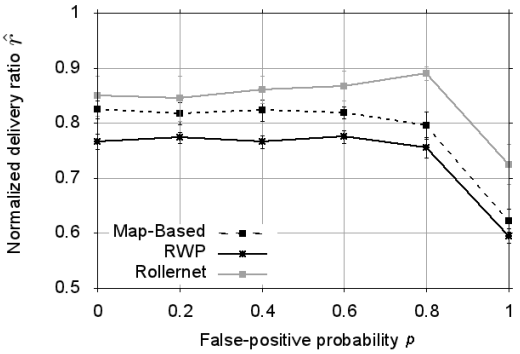

Figure 1. Impact of best-effort authentication on delivery ratios

In each scenario we see false-positive rate increase to as much as $p = 0.6$, with minor impact to $\hat{r}$. At $p = 0.8$, we see a larger, possibly still acceptable, decrease. At $p = 1$, corresponding to no authentication, we once again see the negative impact caused by resource hogs. Equipped with concrete values for p, we can construct a simple best-effort authentication scheme with p as a fixed system-wide parameter known to all domain members.

B. Spot-Checking

The simplest construction involves taking an existing strong authentication scheme and performing verification with probability $1-p$. The actual false positive rate depends on the overall load imposed by the resource hogs. For instance, if half of all bundles seen by an intermediary node with $p = 0.2$ are from resource hogs then the false positive rate is $0.5 \times 0.8 = 0.4$.

Using the same approach with fragment verification, an intermediary verifies only a subset of received fragments without harming the overall network. Spot-checking is simple, easily mapped to the graphs in III-A, and reduces processing overhead for intermediaries. However, it does not reduce transmission or storage overhead.

C. Formal Definition

A best-effort fragment authentication scheme is a tuple of algorithms:

KeyGen(k)
The key generation algorithm is a probabilistic algorithm that computes a public-private key pair: $(pk_{sign}, sk_{sign}) \leftarrow 1^k$, where k is the security parameter of the system. This key provides the strong end-to-end message authenticity and integrity. A suitable k value provides a cryptographic level of security.

Sign(M, sk_{sign}, IV)
Takes an input message $M = s_1|s_2|...|s_n$, where s_i is an individual scrap, a private signing key sk_{sign}, an optional initialization vector(IV), and produces a signature:

$$(\sigma_s, \sigma_{be}, AUX) \leftarrow SIGN(M, sk_{sign}, IV),$$

where σ_s is the output of any cryptographically secure signature scheme, e.g., DSA or RSA, over the whole message and σ_{be} is a secure signature over the vector AUX. $AUX =< aux_1, aux_2, ..., aux_n >$ represents auxiliary data for individual scrap verification.

Verify(M, pk_{sign}, σ_s)
A deterministic algorithm, that given an input message M, public signature verification key pk_{sign}, and possibly valid signature σ_s

$$VERIFY(M, pk_{sign}, \sigma_s) \rightarrow \{TRUE, FALSE\},$$

outputs true if σ_s is a valid signature on M and false otherwise.

VerifyScrap($s_i, pk_{sign}, \sigma_{be}, AUX$)
A probabilistic algorithm, that given an input scrap s_i, public signing key pk_{sign}, valid signature σ_{be}, and auxiliary vector AUX:

$$VRFYSCRAP_p(s_i, pk_{sign}, \sigma_{be}, AUX) \rightarrow \{TRUE, FALSE\},$$

always outputs true if scrap s_i is part of the original message M and true with probability p if it is not. i.e, this construction allows for the possibility of false positives but not false negatives. Note that if $p = 0$ we have a traditional strong fragment authentication scheme.

IV. Constructions

A. Bloom Filter Overview

Bloom filters are probabilistic data structures used for determining set membership. The data structure is a bit array of size m with all bits initially set to zero. An element of a set is added to the data structure by feeding that element to k hash functions. Each function outputs a position in the bit array, which is then set to one. Elements mapping to the same index do not toggle the bit value, i.e., bits remain set.

Given a Bloom filter corresponding to set S, we can test the status of an element by executing $\mathcal{H}()$ and checking if the correct bit is set. False positives are introduced whenever a hash function collision occurs and non-elements mapping to a bit that has been set. To reduce this chance we can increase m or the number of hash functions to k and require that all k bits be set. We illustrate this concept with a simple example in Figure 2. A small set S uses $k = 2$ hash functions to create a Bloom filter.

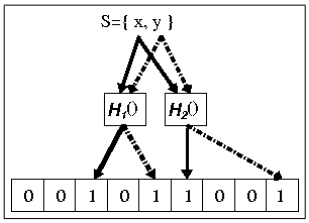

Figure 2. Bloom filter example [k=2]

Bloom filters offer a trade-off between space, computation, and false-positive probability: increasing m decreases the false-positive probability, but also increases storage and transmission requirements. Similarly, increasing the number k of hash functions decreases false-positive probability at the expense of increased computational costs.

B. Best-effort Bloom Filter

We define a best-effort authentication based on Bloom filters as follows:

Algorithm 1 Bloom Filter: Sign

$SIGN(M, sk_{sign}, \perp)$:
  $BF \leftarrow GenerateBloomFilter(M, m, k)$
  $AUX \leftarrow BF$
  $\sigma_s \leftarrow S_{sk_{sign}}(M)$
  $\sigma_{be} \leftarrow S_{sk_{sign}}(BF)$
  return $(\sigma_s, \sigma_{be}, AUX)$

Algorithm 2 Bloom-Filter: Verify Scrap

$VRFYSCRAP(s_i, pk_{sign}, \sigma_{be}, AUX)$:

  $BF \leftarrow AUX$
  if $VERIFY(SIG, pk_{sign}, \sigma_{be}) ==$ **false then**
    return false
  else
    return $TestMembership(BF, s_i)$
  end if

The above algorithms make use of two helper functions. $GenerateBloomFilter(M, m, k)$ takes the scraps of a message M and generates the corresponding Bloom filter of size m with k hash functions using the process described earlier. The $TestMembership(BF, s_i)$ takes an input scrap s_i and checks if the corresponding bits of the k hash functions are set. If all bits are set the method returns $true$ otherwise $false$. Note that this method, with its intrinsic possibility for false positives, make this a best-effort technique.

C. Single-ply Toilet Paper Approach with Truncated Hashes

Using cryptographic hash functions with low output entropy, i.e., limited range size, is considered undesirable for most applications. This decreases the amount of work needed to find collisions or perform pre-image attacks. However, this turns out to be useful for our target scenario: small hash sizes save on bandwidth and transmission overhead. Reducing the size of auxiliary fragment authentication information optimizes the use of limited communication opportunities.

There are always trade-offs when it comes to security and our situation is no different. We want to reduce the output size of a given hash function without compromising the weak authentication scheme. A compromised scheme would allow an attacker to quickly generate a large number of arbitrary fragments that pass verification. The goal is to make fragment forgeries as difficult as possible for an adversary while keeping hash size small.

In our target scenario adversaries would attempt first pre-image attacks[1] in order to replace scraps of a valid message. Given an n bit output hash a successful attack takes 2^n operations. Using any cryptographically secure $H()$ we can construct a truncated hash function, $H_{tr}()$, by simply truncating the output of $H()$ from n bits to n_{tr} bits.

In practice, if we truncate the output of SHA-1 from 160-bits to 80-bits the level of pre-image security is 2^{80}, a number generally recognized as being beyond the capabilities of modern computing hardware. This allows us to cut transmission overhead in half while still maintaining an acceptable level of best-effort authentication. We will now use the concept of truncated hashes to construct a simple best-effort authentication scheme.

The "toilet paper" method, first proposed in the Delay-Tolerant Networking Research Group mailing list, is a proactive approach to fragment authentication. Messages are divided into scraps, i.e., individually authenticatable units. Attached to each scrap is its corresponding digital signature. Intermediaries verify arbitrary scraps by verifying the accompanying signature. The drawback to this approach is the large overhead cost of sending and verifying $O(n)$ signatures.

To reduce this overhead we propose a "single-ply" toilet paper approach based on truncated hashes. The sender computes a truncated hash for each fragment, signs a list of all hashes concatenated together, and attaches the signature to

[1]Given h, find m where $H(m) = h$

the original message. When an intermediate node fragments the message it includes the hashes of all fragments *not* being forwarded. This allows the receiving node to still verify the signature by computing the hash values of received fragments and combining with the hash values received. In general, fragment verification requires knowing hash values for all fragments not currently possessed. This results in the worst case $n-1$ auxiliary hash values.

We define our single-ply toilet paper approach formally in Algorithm 3.

Algorithm 3 Single-Ply Toilet Paper: Sign

$SIGN(M, sk_{sign}, \perp)$:
  $SIG \leftarrow \perp$
  $AUX \leftarrow \perp$
  for $i = 1$ to n **do**
    $SIG \leftarrow \{SIG | H_{tr}(f_i)\}$
    for $j = 1$ to n **do**
      if $j \neq i$ **then**
        $AUX_i \leftarrow \{AUX_i | H_{tr}(f_j)\}$
      end if
    end for
  end for
  $\sigma_s \leftarrow S_{sk_{sign}}(M)$
  $\sigma_{bf} \leftarrow S_{sk_{sign}}(SIG)$
  return $(\sigma_s, \sigma_{bf}, AUX)$

Algorithm 4 Single-Ply Toilet Paper: Verify Scrap

$VRFYSCRAP(f_i, pk_{sign}, \sigma_{be}, AUX_i)$:

  $SIG \leftarrow \perp$
  for $j = 1$ to n **do**
    if $j = i$ **then**
      $SIG \leftarrow \{SIG | H_{tr}(f_i)\}$
    else
      $SIG \leftarrow \{SIG | AUX_{i,j}\}$
    end if
  end for
  return $VERIFY(SIG, pk_{sign}, \sigma_{be})$

V. Related Work

The goal of best-effort authentication is to reduce computation and transmission overhead by relaxing the full authentication assumption used in resource management schemes. Although our focus is on related work in fragment authentication we emphasize that best-effort authentication is a general concept.

The fundamental problem behind fragment authentication is how to establish that a given fragment (or set of fragments) is part of an original authenticated message. The following results provide strong authentication for individual fragments:

Fragment authentication can be viewed as an instance of the secure set membership problem: Given a set of elements, S, provide a proof for an element $s_i \in S$ that any party can verify. Cryptographic accumulators combine all elements of S into a single fixed-size accumulator and generate a short witness for each $s_i \in S$. The witness can be used to verify the membership status of s_i. A survey of accumulator schemes can be found in [11]. Dynamic accumulators, proposed by Camenish and Lysyanskaya [12], allow elements to be added or removed without recomputing over the entire set S. Both accumulator types, albeit computationally expensive, are viable solutions for fragment authentication in opportunistic networks.

There has been a number of results focusing on opportunistic environments. The Delay-Tolerant-Network (DTN) security architecture draft [6] defines a straightforward *toilet-paper* approach. A source host fragments a message and attaches a digital signature, e.g, RSA, to each fragment. This makes each fragment individually and independently verifiable but incurs the overhead of $O(n)$ signatures where n is the total number of fragments.

Partridge [3] proposes cumulative authentication and function definitions. In the former, the main idea is to authenticate a cumulative set of fragments instead of individual fragments. An authenticator function outputs an authenticator that is a function of all fragments up to f_i, i.e., $a_i = A(f_0, f_1, ..., f_i)$. This reduces computation when receiving contiguous sequences of data but requires fragments to follow the same path.

Function definitions take an initialization vector,iv, and fragment f_i as inputs and output the fragment offset i: $A(iv, f_i) = i$. [3] shows how to construct A from traditional hash functions. However, this requires a large iv and overhead ends up being comparable to the toilet-paper approach.

Another approach is using Merkle Hash Trees (MHT) as described in [7]. Messages are proactively divided into fragments and the signed root of the corresponding MHT is attached as the authentication token of the message. Senders transmit fragments sequentially and include co-paths of any fragments not yet received. When the communication link is interrupted, the receiving node discards any partially received fragments and uses the last valid co-path to verify the remaining fragments.

To verify an individual fragment a sender must include the hashes along the co-path to the root. Consider the example shown in Figure 3. In order to verify f_1 a receiver needs the co-path hashes, identified as the circled nodes in the tree, to correctly compute the root hash. If the computed hash does not equal the signed root hash then the fragment is invalid. Note that MHT transmission overheard amortizes nicely when fragments are forwarded sequentially. For example, if fragments f_0 through f_3 have been received then the co-path

for verifying f_3 is a single hash: h_{4-7}.

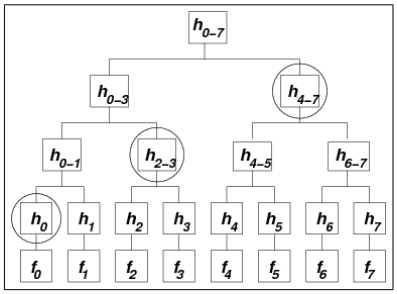

Figure 3. Merkle Hash Tree Computations

Another closely related area is authentication of streaming data, where a source streams data that receivers authenticate. Timed Efficient Stream Loss-tolerant Authentication (TESLA)[13] is one approach where the sender computes a message authentication code (MAC) over each packet using keys from a one-way hash chain. The sender reveals the key of period i during period $i + 1$. Delayed release of keys requires: (1) receivers to buffer data and (2) loose time synchronization. Key release based on time is not feasible in opportunistic networks where contact link durations are unknown.

Another approach includes simple hash chaining[14]. Packet i includes a hash of packet $i + 1$. A valid signature over the first packet authenticates the entire chain. The generalized approach in [15] includes multiple hashes/chains to facilitate recovery when packets are lost. Unfortunately, this does not help in situations where connections are lost and cannot be restored.

The primary drawback of streaming data authentication is the requirement for packets to be sent and verified sequentially. Although, the generalized approach allows sending from arbitrary points in the stream, it requires a large overhead since first signed packet must contain hashes of all subsequent packets. Both approaches are undesirable in DTNs where intermediary fragments must be independently verifiable and overhead needs to be kept low.

Finally, Son *et al* [16] propose a similar idea to reduce transmission overhead. The authors considering a scenario messages have several multiple message authentication codes (MACs) attached. Bloom filters can be used to combine the MACs and reduce transmission overhead. We focus on fragmentation and discuss the security implications of using Bloom filters in opportunistic networks.

VI. Transmission and computation overhead

In this section we give the transmission and computation overhead per message of both strong and best-effort authentication constructions; total overhead in a network is a multiple of per message overhead. Transmission overhead is analyzed in terms of the increase in message size. We follow the notation in Table V for the remainder of this section.

Notation	Description
n	Total number of scraps
σ	Size of a secure signature
H	Output size of strong hash
h	Output size of truncated hash
p	False Positive Rate

Table V
Notation

We first analyze the transmission and computation overheads of traditional strong authentication schemes:

Toilet-paper: Total transmission overhead is one signature per scrap: $n * \sigma$. Total computation overhead is n signature verifications.

Merkle Hash Tree: The transmission overhead depends on the number of co-path hashes sent. If radio contact duration over the opportunistic link is known a priori, then the exact number of scraps and co-path hashes can be sent. In general, contact durations are not known and nodes must interleave scraps with co-path hashes. Table VI shows how the data in Figure 3 is sent during each transmission over a single link. Transmission cut after stage i means all fragments up to f_i are verifiable by the receiver.

Stage	Data
1	$f_0, \sigma, H_{0-7}, H_1, H_{2-3}, H_{4-7}$
2	f_1
3	f_2, H_3
4	f_3
5	f_4, H_5, H_{6-7}
6	f_5
7	f_6, H_7
8	f_7

Table VI
Fragment authentication during data transmission

Total transmission overhead: $\sigma + (n \times H)$. The verification overhead for a complete message is one signature verification and $2n - 1$ hashes to compute the full hash tree.

Spot checking: As described above, the spot checking best-effort authentication technique reduces the computation overhead of intermediate nodes by a factor $1 - p$; it does not affect the transmission overhead.

Truncated hashes: Using truncated hashes halves the transmission overhead of hash values; it does not affect the computation overhead of intermediate nodes.

Bloom filter: The transmission overhead depends on n, p, and the number of hash functions, k. The exact size can be computed as: $m = 1 - (1 - p^{1/k})^{1/kn}$.

For comparison, we may want to fix m and compute the corresponding p: $p = (1 - (1 - \frac{1}{m})^{kn})^k$

The verification overhead for a complete message is one signature verification and $k * n$ hashes.

Example values: Assume a strong 1024-bit RSA signature, SHA-1 (160 bit output) for H, and a message size of 5 MB with 100 KB scraps. We now graph the total transmission overhead (in bytes) of a Bloom filter as a function of p with six values of k in Figure 4. This figure shows the overhead savings that can be achieved when we vary the value of k to achieve a desired p.

Please note that these savings are small relative to our assumed scrap or message size. Therefore, in this case, saving 600 bytes per message is unlikely to noticeably increase delivery ratios. While not much is gained in our target scenario, there may be situations where overhead is important. As the ratio between scrap size and message size decreases the importance of overhead increases.

In general, the savings in transmission time with best effort authentication depend on the typical message size, the false positives rate p and the technique used. (For example, spot checking does not reduce transmission time). We have shown that, independently of mobility model, significant savings, as compared to MHT, are possible for certain message size ranges and p values, when Bloom filter (best-effort) authentication is used. Whether those message size ranges are typical or not, depends on the use case. The savings of best effort authentication are in (i) computation overhead and (ii) transmission overhead. Both savings impact the battery life of intermediary node.

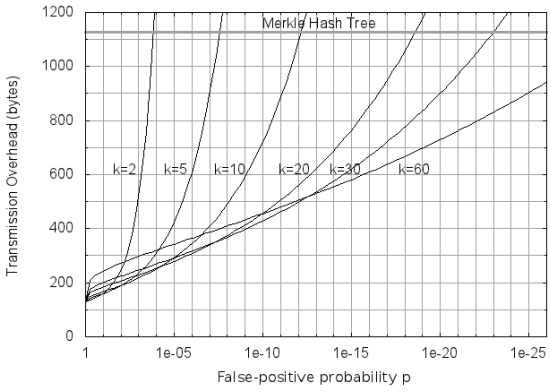

Figure 4. Transmission overhead (with base 1024-RSA signatures) and false positive rate of various Bloom filter k-values

VII. Security Analysis

Adversarial Model: We assume the adversary is not a domain member, but can record messages as a direct recipient or by intercepting transmissions. The adversary does not actively interfere with transmissions, e.g., by frequency jamming. Finally, the adversary has no storage restrictions, but can only perform a polynomial number of computations.

The goal of the adversary is to construct fragments that intermediaries authenticate as belonging to domain members. Forged fragments receive higher priority and improve the personal delivery ratios of the adversary.

Harvesting Attacks: An adversary attempts to forge domain member fragments by harvesting legitimate messages and attempting to impersonate them. The specific threat from harvesting depends on the best-effort authentication mechanism being used:

Spot checking used in conjunction with a strong authentication scheme does not increase the attack advantage if the underlying cryptographic primitive is secure. The probability of successfully delivering a forged fragment depends on p and i, where i is the number of hops on the path to the destination. If we assume, for simplicity, that all intermediaries independently verify fragments with the same probability[2] $1-p$, then the probability of a successful attack is p^i.

Harvesting many signed Bloom filters can lead to an offline pre-image attack. An attacker, who computes a valid pre-image, can replace individual scraps of arbitrary messages using the compromised Bloom filter. To limit the damage from this attack we can include a random number in the header as a unique message identifier and also hash it into the Bloom filter. The random number (1) helps with message reconstruction and (2) makes attacks more difficult by binding the authentication token to a specific message. This prevents an adversary from replaying the authentication token in a later session. Unfortunately, it does not protect messages that may have the same Bloom filter.

Colluding Adversary: A second scenario to consider is one where the adversary colludes with the destination node: an adversary uses harvested headers/signatures to send messages to the destination. To the network it will seem like the messages are coming from a legitimate user and receive priority treatment. Unfortunately, intermediaries can have little impact in this situation since the end host is not correctly verifying signatures. Bloom filters must be chosen such that the amount of work needed to perform this attack deters any potential colluders. If $1 * 10^{-24}$ has a comparable level of security to 2^{-80} then, based on Figure 4, we can set $k = 60$ to achieve high security and reduced transmission overhead at the expense of increased computation.

Denial-of-Service Attacks: We also want to consider the possibility of various denial-of-service(DoS) attacks. One approach is the fragment replacement attack, illustrated in Figure 5, where an adversary prevents the forwarding of valid message fragments. Consider a host S who has a message destined for D. When S encounters intermediaries A

[2]In practice $1-p$ can be selected independently by each device.

and I_1 it will forward fragments $\{a, b\}$ and $\{a\}$ respectively. Now assume that A modifies fragment a to a'. If I_2 accepts the invalid fragment a' from A, before communicating with I_1, it will not accept the valid fragment a. I_2 believes it has already received the correct fragment. If D receives an invalid fragment, causing all valid fragments to be ignored, then the original message will never be reconstructed. To further clarify, we emphasize that fragment hashes are only used for message reconstruction and *not* for any underlying routing purposes.

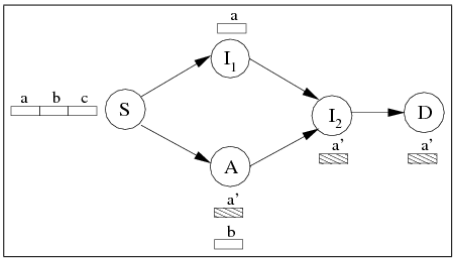

Figure 5. Fragmentation Attack

To avoid this problem we require routing algorithms use hashes when making forwarding decisions. This solves our problem because fragments a and a' hash to different values, and thus, appear as distinct fragments. This pushes the responsibility of detecting invalid fragments to the destination.

The final issue we consider is an adversary who launches a computational DoS attack by generating multiple invalid fragments in hopes of overwhelming the destination. The attacker exploits the fact that the destination must try all possible combinations of fragments claiming to be the same offset.

We can prevent this problem by attaching a signed list of hashes to the original message. To reconstruct a message the destination extracts and verifies the list. Using the list of valid hashes the destination can easily discard invalid fragments and prevent any computational DoS attacks.

Note that the above computational DoS attack is only possible when an adversary is capable of producing a large number of invalid fragments efficiently. This is only possible when, e.g., Bloom filter parameters are such that the false-positive probability is high. If $p \leq 2^{-80}$ this attack is not possible.

VIII. Additional Implications

We have shown an opportunistic networking scenario that does not require strong authentication of messages by intermediate nodes. As another potential application of best-effort authentication, consider the PKI based system proposed in [7], where messages are encrypted using short-lived certificates. Problems arise when users are disconnected from the network when their short-lived certificate expires. Disconnected users, unable to retrieve new certificates, are revoked from the system and unable to send new messages. Sending messages under expired keys only results in them being immediately discarded by intermediaries.

However, intermediaries using best-effort authentication can accept messages (with expired certificates) without negatively impacting delivery ratios. Recently expired certificates, unlikely to be malicious, are likely to be close to their final destination. Intermediaries not overloaded may decide to tolerate the message for possible forwarding.

IX. Conclusion

In this paper, we have investigated the impact of fragmentation on the effectiveness of resource management schemes in the ad-hoc opportunistic network scenario of [1]. We found that in our scenario fragmentation alone is insufficient and must be coupled with fragment authentication schemes to both optimize and protect network resources.

We also observed that strong security guarantees are not required for messages conveyed by intermediate nodes in our opportunistic networking scenario. Benefits of resource management techniques can be retained using "best-effort" authentication mechanisms. With this knowledge, opportunistic network designers can determine how to trade-off between strong cryptographic guarantees and improved delivery ratios through decreased transmission overhead. We speculate that the notion of best-effort authentication may also be relevant in other scenarios, although a careful security analysis is needed for each specific case.

Finally, we described and analyzed two publicly verifiable best-effort authentication methods. We discussed transmission overhead of each and have shown their respective improvements over strong authentication mechanisms.

References

[1] J. Solis, N. Asokan, K. Kostiainen, P. Ginzboorg, and J. Ott, "Controlling resource hogs in mobile delay-tolerant networks," *Computer Communications*, 2009.

[2] M. Pitkanen, A. Keränen, and J. Ott, "Message fragmentation in opportunistic dtns," June 2008, pp. 1–7.

[3] C. Partridge, "Authentication for fragments," *HotNets-IV, The Fourth Workshop on Hot Topics in Networks*, November 2005.

[4] D. Giry and P. Bulens, "Keylength - cryptographic key length recommendation," http://www.keylength.com, 2009.

[5] V. Cerf, S. Burleigh, R. Durst, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-tolerant network architecture," RFC 4838, 2007.

[6] S. Farrell, S. Symington, H. Weiss, and P. Lovell, "Delay-tolerant networking security overview," Internet Draft draft-irtf-dtnrg-sec-overview-03.txt, Work in progress, July 2007.

[7] N. Asokan, K. Kostiainen, P. Ginzboorg, J. Ott, and C. Luo, "Applicability of identity-based cryptography for disruption-tolerant networking," in *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*. New York, NY, USA: ACM, 2007, pp. 52–56.

[8] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proceedings of ACM SIGCOMM Workshop on Delay-Tolerant Networking (WDTN)*, 2005.

[9] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. D. de Amorim, and J. Whitbeck, "The accordion phenomenon: Analysis, characterization, and impact on dtn routing," in *Proc. IEEE INFOCOM*, 2009.

[10] I. Corporation, "Intel motes," http://techresearch.intel.com/articles/Exploratory/1503.htm, 2009.

[11] N. Fazio and A. Nicolosi, "Cryptographic accumulators: Definitions, constructions and applications," Paper written for course at New York University: www.cs.nyu.edu/~nicolosi/papers/accumulators.pdf, 2002.

[12] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," in *Proceedings of Crypto 2002, volume 2442 of LNCS*. Springer-Verlag, 2002, pp. 61–76.

[13] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The tesla broadcast authentication protocol," *RSA CryptoBytes*, vol. 5, p. 2002, 2002.

[14] R. Gennaro and P. Rohatgi, "How to sign digital streams," in *CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*. London, UK: Springer-Verlag, 1997, pp. 180–197.

[15] S. Miner and J. Staddon, "Graph-based authentication of digital streams," in *SP '01: Proceedings of the 2001 IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2001, pp. 232–246.

[16] J.-H. Son, S.-W. Seo, U. Kang, J.-G. Choe, H.-K. Moon, and M.-S. Lee, "Representation of multiple message authentication codes using bloom filters," in *International Technical Conference on Circuits/Systems, Computers and ommunications (ITC-CSCC)*, 2006.

Aalto-ST 19/2011

ISBN 978-952-60-4287-9 (pdf)
ISSN-L 1799-4896
ISSN 1799-490X (pdf)

Aalto University
School of Electrical Engineering
Department of Communications and Networking
www.aalto.fi

BUSINESS +
ECONOMY

ART +
DESIGN +
ARCHITECTURE

SCIENCE +
TECHNOLOGY

CROSSOVER

DOCTORAL
DISSERTATIONS