Broadcast Authentication With Preferred Verifiers

Mahalingam Ramkumar
Department of Computer Science and Engineering, Box 9637, Mississippi State University
Mississippi State, MS 39762, USA. (Email: ramkumar@cse.msstate.edu)

(Received Oct. 19, 2005; revised and accepted Nov. 23, 2005)

Abstract

We introduce a novel cryptographic paradigm of broadcast authentication with "preferred" verifiers (BAP). With BAP, the message source explicitly targets a set of one or more verifiers. For an attacker, forging authentication data of a source, for purposes of fooling preferred verifiers may be substantially more difficult than fooling other (non-preferred) verifiers. We investigate broadcast authentication (BA) with *hashed random pre-loaded subsets* (HARPS), which caters for such a distinction. HARPS, provides for efficient broadcast authentication, with and without preferred verifiers. We argue that BA using HARPS has some very desirable features that make it well suited for securing ad hoc routing protocols. We also briefly discuss some applications of the novel paradigm of BAP in the context of ad hoc routing protocols.

Keywords: Broadcast authentication, key pre-distribution, probabilistic key pre-distribution

1 Introduction

A broadcast authentication (BA) scheme, permits any node to verify the authenticity of the source of the broadcast. This can be achieved using digital signatures if public key cryptography is used. However, in many application scenarios resource constraints may prohibit the use of asymmetric cryptographic primitives.

The main contribution of this paper is an efficient BA scheme, HARPS-BA, utilizing only symmetric cryptographic primitives, based on a probabilistic key predistribution scheme (PKPS). Similar to BA techniques suggested by Alon et al. [1] and (expanded further by) Canetti et al. [4], HARPS-BA is achieved by appending many message authentication codes based on shared secrets (or HMACs) in such a way that *any* verifier would be able to verify a *subset* of the appended HMACs. The RPS-BA scheme by Canetti et al. [4] is based on RPS (random pre-loaded subsets), a PKPS proposed by Dyer et al. in [9]. The proposed scheme, HARPS-BA, is however based on a more recently proposed PKPS, HARPS (hashed random pre-loaded subsets) [27] by Ramkumar et al. Henceforth, for both schemes we shall refer to the appended authentication data (or HMACs) as the "signature" of the node[1].

The *strength* of (or the security offered by) any BA scheme is a measure of the difficulty an attacker faces in forging the signature of an arbitrary source, in order to fool arbitrary verifiers. The *complexity* of a scheme is a measure of the computational complexity (for the source and verifiers), and the bandwidth overheads. The *efficiency* of a broadcast authentication scheme is then a ratio of the strength to its complexity. We show that HARPS-BA is more efficient than RPS-BA.

While the purpose of broadcast is to reach every possible recipient, in real world applications each broadcast may have a different amount of "significance" to different recipients. For instance in applications involving multihop ad hoc networks [29], routing information in each node may be broadcast to the entire network, malicious broadcasts (with forged authentication) is more likely to affect the nodes in the immediate neighborhood than nodes further away. Under such circumstances, it would be useful if broadcast authentication could cater for the "higher strength" required for some "preferred" verifiers.

In addition to the providing more efficient general purpose broadcast authentication than the scheme in [4], HARPS supports this paradigm of "preferred" verifiers (henceforth referred to as BAP). When a broadcast is targeted to one or more preferred verifiers, the appended authentication data (or "signature") may be considerably more difficult for an attacker to forge, for the purpose of fooling (any of) the preferred verifiers.

BAP bears some resemblance to "signatures with *designated* verifiers" [6, 16]. However, while designated signatures can be verified *only* by designated verifiers, BAP can be verified even by non-preferred verifiers (nodes which are *not* explicitly targeted). Moreover, as we shall demonstrate later, while BAP is substantially stronger than "plain" BA (against attempts to fool preferred verifiers), BAP is only *marginally* weaker than plain BA against attempts to fool other *non-preferred* verifiers.

[1]This "signature" does *not* cater for non-repudiation.

The rest of this paper is organized as follows. In Section 2 we provide a brief review of probabilistic KPSs. In Section 3 we analyze the performance of HARPS-BA and RPS-BA and HARPS-BAP (BA with preferred verifiers). In Section 4 we discuss the need for BA for securing ad hoc routing protocols and compare and contrast various approaches to realize this important security association. We then discuss how HARPS-BA (and HARPS-BAP in particular) have many desirable features that make it very suitable for securing ad hoc routing protocols. Conclusions are offered in Section 5.

2 Probabilistic Key Predistribution Schemes (PKPS)

A key pre-distribution scheme (KPS) consists of a key distribution center (KDC) and N nodes with unique IDs. The KDC chooses a set of secrets $\mathbb{S}$. A node with ID A is provided with a set of secrets $\mathbb{S}_A$, which is a function of the KDC's secrets $\mathbb{S}$ and the ID of the node. Any two nodes A and B can discover a unique secret K_{AB} that no other node can discover. However, as the secrets distributed to each node are not independent, an attacker who has gained access to secrets stored in many nodes can compromise all the KPS secrets. A n-secure KPS can resist compromise of all secrets from n nodes.

Unlike *deterministic* n-secure KPSs which provide unconditional assurances as long as n or less nodes are compromised, *probabilistic* KPSs (PKPS) provide only probabilistic assurances. For a PKPS, by exposing secrets from n nodes (say $\mathfrak{O} = \{O_1 \cdots O_n\}$), an attacker can discover *shared secrets* between arbitrary nodes (say the secret K_{AB} between nodes A and B where $A, B \notin \mathfrak{O}$) with a probability p. However, an attacker may have to expose secrets from a substantially larger number $n_s >> n$ nodes (say $\mathfrak{O}' = \{O_1 \cdots O_{n_s}\}$) to expose *all* secrets in some node $A \notin \mathfrak{O}'$ with probability p. Thus PKPSs are aptly described as (n, n_s, p)-secure. Alternatively, we could also describe a PKPS as (n, p)-secure against "eavesdropping" attacks (where the motivation of an attacker is to expose shared secrets between nodes) and (n_s, p)-secure against "synthesis" attacks (compromising all secrets of a node using secrets exposed from *other* nodes).

The first PKPS in the literature, LM-KPS, was proposed by Leighton and Micali [19]. In the LM-KPS the secrets provided to each node are repeated hashed versions of the KDC's secrets. The second PKPS was proposed in 1995 by Dyer et al. [9].

Dyer's scheme, like the matrix key pre-distribution scheme by Gong et al. [13], and the scheme by Mitchell et al. [22] relied on allocation of subsets of secrets to each node from the KDCs pool of secrets. While the matrix scheme uses very simple (and inefficient) strategies for the actual *allocation* of subsets, more complex and efficient strategies were investigated by Mitchell et al. [22], influenced by the seminal work of Erdos et al. on subset intersections [10]. Dyer et al. recognized that the complex allocation strategies which make the Mitchell et al. scheme [22] impractical could be easily substituted with *random* allocation of subsets with very little penalty, and provided an elegant analysis of the security of schemes (which we shall refer to as RPS - for "random pre-loaded subsets") employing random subset allocations. More recently, Dyer's RPS has been employed by various researchers for securing sensor networks [5, 8, 11, 20, 26, 31] and ad hoc networks[2].

PKPS thus exploit two fundamental "dimensions" for their security

1) the *uniqueness of intersections* of large subsets consisting of *independent* keys, and

2) *generating many keys* from each independent key.

The RPS scheme by Dyer et al, based on random allocation of subsets, utilizes the first dimension. The LM-KPS utilizes the second dimension (each independent key chosen by the KDC is hashed many times to derive many *dependent* keys). More recently, Ramkumar et al. proposed HARPS (HAshed Random Pre-loaded Subsets) [27] which makes use of *both* dimensions. Consequently, RPS and LM-KPS are special cases of HARPS.

HARPS is defined by the set of parameters (P, ξ, L) and two functions $(F(), h())$.

1) The KDC chooses and indexed set of P secrets $K_1 \cdots K_P$.

2) Each node is provided with a *hashed* subset of $k = \xi P$ keys on an *average*, where $\xi \leq 1$ is the *probability* with which a key corresponding to some index $1 \leq i \leq P$ is assigned to some node.

3) The public random function $F()$ determines

   a. if a key corresponding to some index $1 \leq i \leq P$ is assigned to a node,

   b. and the "hash depth" of such a key. The hash depth refers to the number of times a key is hashed repeatedly using the cryptographic hash function $h()$. We shall represent by ${}^{j}K_i = h^j(K_i)$, the result of repeated hashing of K_i, j times, using the hash function $h()$.

Thus if $\{A_1 \cdots A_{k_A}\}$ are the indexes assigned to node A and $\{a_1 \cdots a_{k_A}\}$ their respective hash depths, the set of k_A pre-loaded secrets $\mathbb{S}_A$ assigned to node A are

$$\mathbb{S}_A = \{{}^{a_1}K_{A_1}, {}^{a_2}K_{A_2}, \ldots, {}^{a_{k_A}}K_{A_{k_A}}\}.$$

LM-KPS is a special case of HARPS with $\xi = 1$, and RPS is a special case of HARPS with $L = 0$ (or keys are not hashed before pre-loading).

While the primary intention of RPS (and HARPS) was mutual authentication, Alon et al. [1] and Canetti et al.

[2]Unfortunately, most papers in the current literature employing this idea have overlooked Dyer's contribution.

[4] investigated the use of RPS for broadcast authentication. With RPS, for mutual authentication of nodes A and B, the secret K_{AB} is derived from all shared secrets (the indexes of which could be easily determined by using the public one-way function $F()$ as $F(A) \cap F(B)$). In HARPS, for every shared index the node with the lower hash depth hashes it keys repeatedly to reach the same hash depth as the corresponding secret with the other node. Thus all shared indexes, now at the same hash depths, are used for deriving the secret K_{AB}. Just as RPS is used for BA, HARPS can also be used for BA.

3 Broadcast Authentication Using PKPSs

The basic idea used in BA with PKPSs is very simple. The source of the broadcast appends the message with many key based message authentication codes, or HMACS - one corresponding to each of the k keys it possesses (or P keys if the source is the KDC). For RPS, any verifier will be able to verify the HMACs corresponding to all the keys the verifier *shares* with the message source.

It is assumed that the KPS secrets (HARPS / RPS *key-rings*) are stored in "tamper-resistant" and "read-proof" devices. However, an attacker with sufficient resources may be able to compromise the secrets stored in a finite number of devices. Thus while the security of broadcast authentication employing digital signatures is based on the assumption that an attacker *cannot compromise the private key* of a node (which the attacker is trying to impersonate), the security of broadcast authentication using KPSs relies on the assumption that an attacker may not be able to expose secrets from a many "tamper-resistant" and "read-proof" devices. With probabilistic KPSs, an attacker who has exposed secrets from some n devices can impersonate arbitrary nodes with some probability p.

The advantages of HARPS-BA over RPS-BA is a result of the use of the "additional dimension" - repeated hashing of pre-loaded keys. In HARPS-BA the source has the added flexibility of *choosing the hash depth* of the keys to be used for HMACs. For example, if the hash depth of some key (say corresponding to index $1 \leq i \leq P$) is d (or if the source has the key ${}^{d}K_i$) the source can use any ${}^{d_s}K_i$ for the HMAC, where $d_s \geq d$. Thus even though some verifier may have a key corresponding to the i^{th}index - say the key ${}^{d_v}K_i$, the verifier will be able to verify the HMAC only if $d_v \leq d_s$. Obviously, if the source chooses $d_s = L$ for every key it has, any verfier having a secret corresponding to the index will be able to verify the HMAC. In other words, HARPS-BA with $d_s = L$ is the same as RPS-BA.

The strategy therefore is to choose an optimal "signing" depth for each key, which we shall represent by L_p. Once the optimal signing depth is fixed, a source can choose that depth for index i if the hash depth d of its key ${}^{d}K_i$ is such that $d \leq L_p$. Otherwise, the node chooses the lowest depth it can - the depth d.

Note that the number of HMACs a verifier can verify with HARPS-BA is between $P\xi^2/2$ (for the choice of $L_p = 0$) to $P\xi^2$ (for $L_p = L$, which is the same as RPS-BA). Also note that while the security of mutual authentication and BA are identical for RPS, this is not the case for HARPS. Specifically, in RPS both mutual authentication and BA are based on all secrets two nodes share. For HARPS, while for mutual authentication the nodes can utilize all shared indexes, for BA the verifier may not be able to verify *all* secrets corresponding to the shared indexes. However, under the *optimal* choice of L_p, it is still less likely (as we shall see) that a coalition of attackers could forge any HMAC.

The freedom available to the source to choose the hash depth also makes the novel cryptographic paradigm, broadcast authentication with *preferred* verifiers (BAP), possible. For BAP the strategy is to choose the hash depths to "suit" the preferred verifiers. Furthermore, BAP will affect only the hash-depths of the keys that the source shares with the preferred verifiers. For other HMACs the source will employ the same strategy as regular BA (optimal L_p). Apart from the fact that the strength of other HMACs are not affected, non-preffered verifiers may still be able to authenticate some HMACs meant for preferred verifiers. Consequently, we shall see that the verification of BAP by non-preferred verifiers (or BAP-NP) is only marginally weaker than regular HARPS-BA. At the same time, BAP verified by preferred verifiers is substantially stronger.

It is important to note that the "freedom" in choosing the hash depth is actually regulated by deterministic strategies, which may be imposed and periodically modified by the KDC. Once the strategy is fixed any node will be able to verify the signature as every node will know what depth should be used for each HMAC. Thus for BAP, all verifiers (preferred and non-preferred) also need to know the identities of the preferred verifiers.

Note that in our formulation the number of appended HMACs (k_A when the source is node A) can be anywhere between 0 and P (or $0 \leq k_A \leq P$). However, with a high probability $k_A \approx \xi P$. The signature of a node A, for a message M is thus

$$\mathfrak{S}_A(M) = [H_1 \parallel H_2 \parallel \cdots \parallel H_{k_A}],$$

where $H_i = h(A \parallel M \parallel {}^{x_j}K_{A_j}))$, $a_j \leq x_j \leq L$. The primary reason for including[3] the ID of a node for calculation of the $H_i, 1 \leq i \leq k$ in each HMAC is to ensure that the attacker cannot "pool" authentication data from different nodes for the same message M to forge the signature just by "requesting" such nodes to authenticate M. The only way for the attacker to forge messages is by 1) actually tampering with devices and exposing buried secrets, or 2) "guessing" the HMACs.

[3]Time-stamps and a random nonce could also be included for preventing replay attacks.

3.1 Security Analysis

The analysis of security - the probability that an attacker cannot for the HMACs of a source for the purpose of fooling a specific verifier - follows the same lines as the analysis in [4].

The HMACs corresponding to any of the P keys (say the key corresponding to index i, $1 \leq i \leq P$) is "safe" (cannot be forged by the attacker) if the following five conditions are satisfied:

- C1: The source has a key corresponding to the index (probability ξ if the source is a peer node, and probability 1 if the source is the KDC).
- C2: The verifier has a key corresponding to the index i (probability ξ).
- C3: The hash depth of the i^{th}key with the verifier, d_v, is *not* greater than the hash depth (say d_s) used by the source for that index. (probability d_s/L).
- C4: The attacker's pool of secrets (accumulated from n nodes) does not have any key ${}^xK_i$ where $x \leq d_s$.
- C5: The attacker cannot "guess" the HMAC corresponding to the i^{th}key. If each HMAC is b bits long, the probability that the attacker cannot guess the HMAC is $\alpha(b) = (1 - 2^{-b})$.

If all the conditions **C1** through **C5** are satisfied for some index $1 \leq i \leq P$, the corresponding HMAC is "safe" (cannot be forged by the attacker). Let us represent by ϵ', the probability that the i^{th}HMAC is safe. Even if one HMAC is safe, the verification fails (or the attacker fails to fool the verifier). Thus the probability that the attacker can forge the signature of a node in order to fool a specific verifier is

$$p'_F = (1 - \epsilon')^P.$$

Let ϵ represent $\lim_{b\to\infty} \epsilon'$ (or if it is infeasible for any attacker to *guess* the HMAC bits). Thus $\epsilon = \epsilon'/\alpha$ and the corresponding probability of forgery is $p_F = (1 - \epsilon)^P$.

Note that p'_F is the *per-message forgability* probability, for which all conditions **C1** to **C5** need to be satisfied. However p_F is the probability that the attacker cannot expose all keys needed for calculating the HMACs of a source that a *particular* verifier can verify (on an average $\xi^2 P$ secrets). Thus for evaluation of p_F (or ϵ) we only need to take conditions **C1** to **C4** into account.

Considering that

$$p_F = (1 - \epsilon)^P \approx (1 - \epsilon'/\alpha)^{P/\alpha},$$

we can easily see that bandwidth-computation trade-offs are possible. In other words, instead of using large b for each HMAC, we can use say $b = 1$ bit for each HMAC and increase P (and consequently the number of HMACs that have to be evaluated) by a factor $1/\alpha(1) = 2$ (or b bit for each HMAC and increase P by a factor $1/\alpha(b)$).

3.1.1 Verifier Complexity

The complexity of the verification process is perhaps more crucial for any BA scheme than the complexity of signing. The complexity for the source is evaluation of $k = \xi P$ HMACs. The complexity of verification on the other hand is verification of $\xi k = \xi^2 P$ (the average number of *shared* keys) HMACs. It can be easily shown (see [4]) that the optimal choice of ξ for some n is $\xi \propto 1/n$. Under this choice $k = O(n)$ and $P = O(n^2)$. Thus the verification complexity is $\mathbb{O}(1)$. In other words the security of PKPS based BA schemes can be increased arbitrarily *without* increasing the verifier complexity.

3.2 Performance Analysis

3.2.1 Broadcast by KDC

If the KDC chooses a strategy of choosing hash depth of L_p for each MAC key, any node can verify the i^{th} key with a probability $\xi \frac{L_p}{L}$ (the probability that a verifier has the key is ξ, and the probability that the hash depth of the key is not greater than L_p is $\frac{L_p}{L}$. We shall represent the binomial probability that *exactly* u out of n nodes have the key with index i as

$$B_\xi(n, u) = \binom{n}{u} \xi^u (1 - \xi)^{n-u}.$$

Also, let $G_L(d) = \frac{L-d}{L}$. Thus,

$$\epsilon(n, L_p) = \left(\xi \frac{L_p}{L}\right) \sum_{u=0}^{n} B_\xi(n, u) G_L(L_p)^u.$$

For RPS-BA on the other hand,

$$\epsilon(n, L_p = L) = \xi(1 - \xi)^n.$$

3.2.2 Broadcast by Peer Nodes

The source's strategy in this case is to choose an optimal hash depth of L_p *whenever possible*. However, the source can choose depth L_p only for *some* keys - the source node would have roughly $\frac{kL_p}{L}$ keys with hash depth less than or equal to L_p, and $\frac{k(L-L_p)}{L}$ keys with hash depth greater than L_p. Or the source uses depth L_p with probability $\frac{L_p}{L}$, and uses some depth $j > L_p$ with probability $\frac{1}{L} \forall j > L_p$.

If the hash depth used is L_p, the probability that a verifier (who has the i^{th}key) can verify the HMAC is $\frac{L_p}{L}$. Similarly, if the chosen hash depth is $l > L - p$, the probability that a verifier can verify the HMAC is $\frac{l}{L}$. Thus

$$\begin{aligned}\epsilon(n, L_p) &= \left(\xi \frac{L_p}{L}\right)^2 \sum_{u=0}^{n} B_\xi(n, u) G_L(L_p)^u \\ &\quad + \sum_{l=L_p+1}^{L} \xi^2 \frac{l}{L^2} \sum_{u=0}^{n} B_\xi(n, u) G_L(l)^u.\end{aligned}$$

For RPS-BA,

$$\epsilon(n, L_p = L) = \xi^2(1 - \xi)^n.$$

3.2.3 Joint Verification by J Peers

In this case, any verifier accepts the signature as authentic only if $J-1$ other verifiers confirm the authenticity of the signature. Thus the attacker coalition has to fool *all* J nodes in order to be successful. For HARPS-BA, the probability that exactly j out of J verifiers have the i^{th}key is $B_\xi(J,j)$, and the probability that at least one of the j keys has hash depth less than or equal to l is $1-G_L(l)^j$. Thus

$$\epsilon_J(n,L_p) = A+B, \text{ where}$$

$$\begin{aligned} A &= \xi\frac{L_p}{L}\sum_{j=1}^{J} B_\xi(J,j)(1-G_L(L_p)^j)\sum_{u=0}^{n} B_\xi(n,u)G_L(L_p)^u, \\ B &= \frac{\xi}{L}\sum_{l=Lp+1}^{L}\sum_{j=1}^{J} B_\xi(J,j)(1-G_L(l)^j)\sum_{u=0}^{n} B\xi(n,u)G_L(l)^u. \end{aligned}$$

It can be easily seen that for RPS-BA

$$\epsilon J(n,L_P=L) = \xi(1-(1-\xi)^J)(1-\xi)^n.$$

3.3 Broadcast Authentication with Preferred Verifiers

In the BAP mode of operation, the signature of a source is targeted towards some preferred verifiers. Once again there are two different approaches for targeting BA 1) signatures that will be verified *independently* by T targeted verifiers and 2) signatures meant for joint verification by J targeted verifiers. In both cases the hash depths of each key uses for the HMACs is chosen to "suit" the preferred verifiers. Obviously, in order to determine which hash depths should have been used by the source, the verifiers need to know the identities of the preferred verifiers. Thus the authentication data has to explicitly include the IDs of the preferred verifiers. The BAP-"signature" $\mathfrak{S}'_A(M)$ for a message M from a source A, targeted to verifiers $V_1\cdots V_G$ is therefore of the form

$$\mathfrak{S}'_A(M) = [V_1 \parallel \cdots \parallel V_G \parallel H_1 \parallel H_2 \parallel \cdots \parallel H_{k_A}].$$

T Independent Preferred Verifiers: For the case where there are T preferred verifiers who independently verify the signature the strategy is to choose the hash depths such that each of the T preffered nodes can verify as many HMACs as possible. Thus for the i^{th}key if the hash depth of source is s, and if j out of T preferred verifiers have the i^{th}key with hash depths $v_1\cdots v_j$, the source would choose the depth as $\max(s, v_1\cdots v_j))$.

The probability that the source, and exactly j of the T verifiers have the i^{th}key is $\xi B_\xi(T,j)$. Under this condition, the source would choose a depth l for the i^{th}key if

1) Source has hash depth l (probability $\frac{1}{L}$) and all j (out of T) verifiers have hash depths less than or equal to l (probability $(1-g(l))^j = \frac{l^j}{L^J}$), or

2) Source has hash depth less than l (probability $\frac{l-1}{L}$) and j verifiers have a maximum hash depth equal to l (probability $\frac{l^j-(l-1)^j}{L^j}$).

We shall represent the conditional probability of choosing depth l as $P_j(l)$, where

$$\frac{1}{L}\frac{l^j}{L^j} + \frac{(l-1)}{L}\frac{\left(l^j-(l-1)^j\right)}{L^j}.$$

Also note that under the condition that j of T verifiers have the i^{th}key, the probability that a *particular verifier* has the I^{th}key is $\frac{j}{T}$. Thus

$$\epsilon_T(n) = \xi\sum_{j=1}^{T}\frac{j}{T}B_\xi(T,j)\sum_{l=1}^{L}P_j(l)\sum_{u=0}^{n}B_\xi(n,u)G_L(l)^u.$$

J Joint Preferred Verifiers: On the other hand, for the case where J preferred verifiers jointly verify the signature, the strategy is to choose the hash depths such that *at least* one of the J nodes can verify any HMAC. Thus for the i^{th}key if the hash depth of the source is s, and if j out of J verifiers have the i^{th}key with hash depths $v_1\cdots v_j$, then the source would choose the hash depth as $\max(s, \min(v_1\cdots v_j))$.

The source can use the i^{th}key with probability ξ, and the probability that j out of J verifiers have the i^{th}key is $B(J,j,\xi)$. Under this condition, the source would choose a depth l for key i under two conditions:

1) Source has depth l (probability $\frac{1}{L}$) and the minimum depth of j nodes for key i (which have key i) is less than or equal to j - which happens with probability $(1-G_L(l)^j)$.

2) Source has depth less than l (probability $\frac{l-1}{L}$) and the minimum hash depth for the i^{th}key among the j nodes is *exactly* l. The probability that the minimum depth is greater than $l-1$ is $G_L(l-1)^j$, and the probability that the minimum depth is greater than l is $G_L(l)^j$. Thus the probability that the minimum depth is equal to l is $G_L(l-1)^j - G_L(l)^j$.

Let $Q_j(l)$ denote the conditional probability that the source chooses depth l for any key. We have

$$\begin{aligned} Q_j(l) &= \frac{1}{L}\left(1-G_L(l)^j\right) \\ &+ \frac{l-1}{L}\left(G_L(l-1)^j - G_L(l)^j\right), \end{aligned}$$

and thus

$$\epsilon_J(n) = \xi\sum_{j=1}^{J}B_\xi(J,j)\sum_{l=1}^{L}Q_j(l)\sum_{u=0}^{n}B_\xi(n,u)G_L(l)^u.$$

3.3.1 BAP-NP: Verification of BAP by Other Verifiers

Corresponding to the two cases (BAP with J jointly targeted nodes and BAP with T individually targeted nodes) we have to consider the strength of the signature for purposes of verification by no-preferred verifiers - or BAP-NP. The expressions for the probability that the i^{th}HMAC is safe have three terms. The first term accounts for the keys that are affected due to targeting. The second and third terms account for other keys - keys for which the source has a hash depth less than or equal to L_p (in which case the source could employ depth L_p) and keys for which the source has depth greater than L_p (in which case the source would use its hash depth for that key).

For the case of verification by *other* verifiers of a BAP *individually targeted* to T verifiers, the expression for the probability that the i^{th}key is safe, is

$$\begin{aligned}\epsilon_T(n, L_p) &= \xi \sum_{j=1}^{T} B_\xi(T,j) \sum_{l=1}^{L} P_j(l) \frac{l\xi}{L} \sum_{u=0}^{n} B_\xi(n,u) G_L(l)^u \\ &+ \frac{\xi L_p}{L} B_\xi(T,0) \frac{\xi L_p}{L} \sum_{u=0}^{n} B_\xi(n,u) G_L(L_p)^u \\ &+ \frac{\xi}{L} B_\xi(T,0) \sum_{l=L_p+1}^{L} \frac{\xi l}{L} \sum_{u=0}^{n} B_\xi(n,u) G_L(l)^u.\end{aligned}$$

For the case of verification by other verifiers of a BAP jointly targeted to J verifiers, the expression for the probability that the i^{th}key is safe, is

$$\begin{aligned}\epsilon_J(n, L_p) &= \xi \sum_{j=1}^{J} B_\xi(J,j) \sum_{l=1}^{L} Q_j(l) \frac{l\xi}{L} \sum_{u=0}^{n} B_\xi(n,u) G_L(l)^u \\ &+ \frac{\xi L_p}{L} B_\xi(J,0) \frac{\xi L_p}{L} \sum_{u=0}^{n} B_\xi(n,u) G_L(L_p)^u \\ &+ \frac{\xi}{L} B_\xi(J,0) \sum_{l=L_p+1}^{L} \frac{\xi l}{L} \sum_{u=0}^{n} B_\xi(n,u) G_L(l)^u.\end{aligned}$$

3.4 Performance Analysis

The performance of BA and BAP are depicted in six plots in Figure 1. In all six plots the x-axis is n - the number of nodes an attacker has compromised (and exposed all secrets buried in them). The y-axis is a measure of the strength of the BA scheme. More specifically the y-axis is $\log_2(1/p_F) = -\log_2 p_F$ - which is the "bit-security" of the BA scheme. For example $p_F \approx 10^{-20}$ is roughly equivalent to 64-bit security - $\frac{1}{2^{64}} \approx 10^{-20}$ is the probability with which one can successfully "guess" a 64 bit key. The plots correspond to p_F for $(P = 2048, k = 256)$ or equivalently p'_F for large b or p'_F for $(P = 2048/\alpha(b), k = \xi P/\alpha(b))$ for any b. For all cases (except for broadcasts by the KDC), the complexity for the source is evaluation of $\xi P = 256$ HMACs on an average, and the verifier complexity amounts to verification $\xi^2 P/2$ to $\xi^2 P$ (or 32 to 64) HMACs on an average.

The plots represent the strength of BA / BAP employing HARPS for various choices of L_p, and under different scenarios investigated earlier. Also as HARPS-BA with $L_p = L$ is identical to RPS-BA the plots also provide a comparison of RPS and HARPS for BA.

The **top-left** plot is for broadcast by the KDC. Choice of large $1 \leq L_p \leq L$ (closer to L) offers higher bit-security for *small* values of n while smaller value of L_p does better for larger n. With HARPS an appropriate value of L_p can be chosen based on the "threat level," which is obviously not possible for RPS (once deployed it may not be easy to modify ξ). The **top-right** plot depicts BA by peers (for now ignore the dark solid line marked BAP). Similar to broadcasts by KDC, lower values of L_p are more useful for larger n.

While the performance advantage of HARPS over RPS is not substantial for any fixed L_p (the important advantage comes out the ability to choose L_p dynamically) for the case of *joint* verifiers, HARPS is significantly more secure than RPS. This can be seen from the **middle-left** plot from the lines corresponding to RPS (dotted lines) and HARPS (solid lines) for $J = 2$ and $J = 5$, where J is the number of joint verifiers (ignore the dashed lines marked BAP).

As the number of joint verifiers increase the task of the attacker becomes harder, as the attacker has to be able to forge more and more HMACs (more HMAC will be verified). Note that for $n > 22$ HARPS with just 2 joint verifiers performs better than RPS with 5 joint verifiers. In the limit - either when there are are unlimited number of verifiers (or if the attacker desires to fool *all* possible verifiers, or equivalently, if the attacker desires to fool the KDC) the attacker has to forge *all* k HMACs appended by the source (or compromise *all* k keys of the source).

Recall that such "synthesis" attacks are in general much more difficult for PKPSs. For a (n, n_s, p)-secure PKPS, the number n_s is the number of nodes an attacker needs to compromise for synthesis attacks with probability p. While for all PKPSs $n_s >> n$, it is more true for HARPS. For RPS while n_s is an order of magnitude higher than n for HARPS n_s is over 2 orders of magnitude higher. This is the reason for the substantial advantage of HARPS over RPS for the case of joint verifiers.

The next plot (**middle-right**) depicts the limits of performance for HARPS and RPS (the limit being the case when the broadcast is verified by the KDC or an infinite number of joint verifiers). The figure also shows the case for $J = 20$ for HARPS and RPS (once again ignore the dashed line marked BAP). Two noteworthy points are as follows:

1) For $J = 20$ RPS is almost saturated (with very little room for improvement with further increase in J), while for HARPS there is still considerable room for improvement.

2) The room for improvement indicates that even with large J a number of HMACs are still unverified. Thus even with a large number of attackers it may be very difficult for the coalition to compromise *all* secrets. In other words, the security of HARPS deteriorates *very gracefully* with increasing n.

Another advantage of HARPS over RPS comes out of the ability to choose preferred verifiers. The solid line

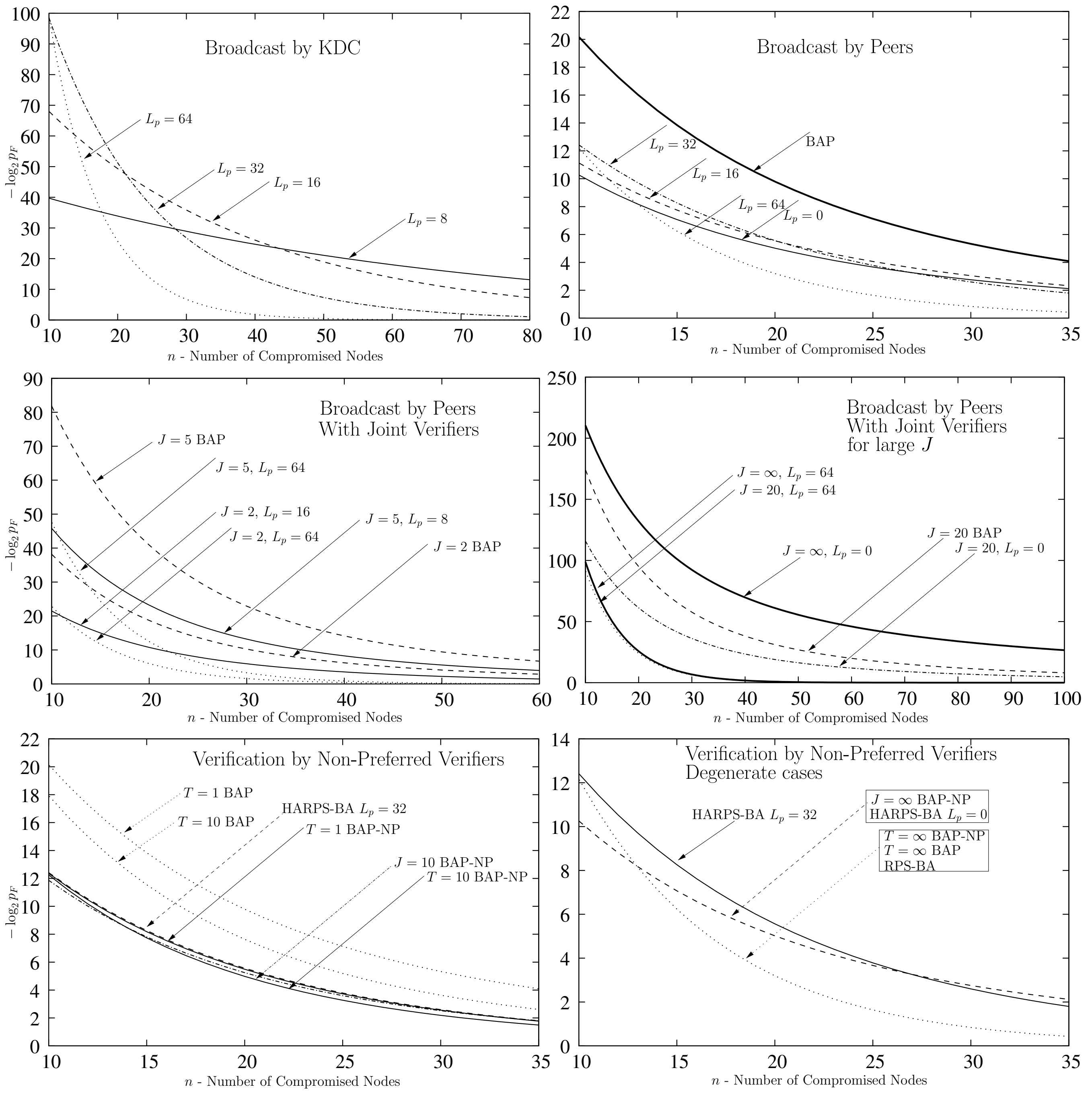


Figure 1: Performance of HARPS-BA. The plots corresponds to the probability p_F (probability that the attacker can expose all secrets required for calculation of the HMAC) for $P = 2048, \xi = 1/8$ and $L = 64$. Alternately the plots also represent p'_F (per-message forgability probability) for $P' = P/\alpha(b)$ where each HMAC has b bits (for example $P = 4096$ and $b = 1$).

in the **top-right** plot corresponds to BAP with one preferred verifier. Note that BAP is substantially stronger than BA for all n. In other words, it is significantly more difficult for an attacker to forge HMACs for the purpose of fooling preferred verifiers.

We also analyzed two situations involving multiple preferred verifiers - J joint preferred verifiers or T preferred verifiers who independently verify the signature (when $T = J = 1$ both situations are identical). The lines corresponding to preferred verifiers are labelled BAP. We also need to consider the strength of BAP when verified by non-preferred verifiers (labelled BAP-NP).

The **bottom-left** plot provides a comparison of the strengths of different strategies. As T is increased the signature becomes less targeted. Thus BAP with $T = 10$, as expected, is not as strong as BAP with $T = 1$. However BAP with $J = 10$ joint verifiers would be substantially stronger than BAP with $J = 1$. The **middle-left** plot shows that BAP for $J = 5$ is substantially stronger than "plain" BA with $J = 5$. Also from the plot in **middle-right** we can see that for large J, BAP approaches the limit (strength of verification by the KDC) faster than BA.

Note that in the **bottom-left** plot the performance of HARPS-BA is practically indistinguishable from BAP-NP (BAP verified by non-preferred verifiers) for $T = 1$. In other words, there is practically *no disadvantage to targeting*. While it is substantially more difficult to fool targeted verifiers, for fooling *other* verifiers it is still almost as hard as regular BA. For the case BAP-NP for $T = 10$ and $J = 10$ we can however observe a marginal loss of strength.

The degenerate[4] cases corresponding to a "large" number of targeted verifiers are depicted in the plots in **bottom-right**. We can see that for the case of BAP with $T \to \infty$ the source would end up choosing $L_p = L$ for *all* HMACs - which reduces to RPS-BA. Thus RPS-BA also corresponds to the case of BAP for $T \to \infty$, and simultaneously BAP-NP for $T \to \infty$. On the other hand, if a signature is targeted to many joint verifiers the source would end up using the least possible hash depth for all HMACs (or $L_p = 0$). Or BAP-NP for $J \to \infty$ is identical to HARPS-BA with $L_p = 0$.

The primary advantages of HARPS-BA over RPS-BA can therefore be summarized as follows:

1) With HARPS-BA Lp can be regulated even after the deployment to improve security.

2) Significantly improved performance under joint verification

3) More graceful degradation of security as n increases, and

4) Ability to impose preferred verifiers to improve security.

[4]Obviously targeting is pointless when the number of nodes targeted becomes very large.

4 BA for Ad Hoc Routing Protocols

A multi-hop ad hoc network (MH-AHN) can be seen as a collection of nodes, where each node has it unique "view" of the world around it. Each node has some *information* not available to other nodes, which needs to be disseminated throughout the network. For ad hoc routing protocols, the information is the topology of the network. The primary goal of all efficient routing protocols is to *maximize information transmission* while *minimizing the necessitated use of resources* like bandwidth / computational overheads.

Efficient solutions to the problem of routing in MH-AHNs can be challenging especially due to constraints on computational and bandwidth overheads. This problem is rendered even more complex under the presence of malicious nodes that could propagate misleading information. It is well known that this "Byzantine Generals" problem[5] is more tractable if source authentication is possible [18]. Thus in order to prevent malicious nodes from modifying data reported by other nodes all data can be cryptographic-ally authenticated. Various secure extensions to routing protocols, cocnsisting primarily of adding cryptographic authentication to existing routing protocols have been considered in the literature [14, 15, 23, 28].

As the data originating from any node may be necessary for many other nodes, broadcast authentication is a very useful security association for securing ad hoc routing protocols. However, the fact that the data advertised or broadcast by some node has been cryptographic-ally validated does *not* imply that the data itself is valid. This would only be true under cases where the devices are completely *trusted*, and only such trusted devices are provided with access to secrets which could be used for authentication. Trusted devices [25] will possess "unflinching morals" ("thou shalt always route packets" and "thou shalt not advertise false routing tables" [21]).

Practical realization of trusted devices calls for assurances of *tamper-resistance* and *read-proofing* [12]. For scenarios where fool-proof *tamper-resistance* is not possible, nodes with secrets that have turned malicious (by tampering with the software that controls the functioning of the device), could advertise misleading information that are cryptographic-ally well authenticated. Similarly under scenarios where *read-proofing* of secrets is not possible, secrets extracted from trusted devices could be used by malicious devices to impersonate trusted devices.

In the absence of fool-proof guarantees for tamper-resistance and read-proofing there are three possible approaches - with increasing levels of complexity - to mitigate the damage that can be imposed by malicious devices.

1) detecting malicious intents (or inconsistencies in data

[5]Reaching a consensus among distributed units if some of them give misleading answers. The classical problem concerns generals plotting a coup, where some generals may be "moles."

advertised) and dropping such packets,

2) identifying the perpetrator unambiguously (by the node which detects the inconsistency), and drop all further packets from the perpetrator, and

3) providing culpable proof of the perpetrators malicious behavior to ensure that he / she can be ejected (revoked) from the system.

Even the first step is possible (using redundancies in the routing protocol) only if we are able to *restrict the size of attacker coalitions.* The second step - unambiguously identifying the perpetrator, is more difficult (requires more redundancies). It is easy to see the analogies between the "detection of inconsistency" / "attribution of malicious intents" (to their perpetrators) and the well known principles of *error-detection* and *error-correction* respectively. While mere detection of errors (analogous to detection of inconsistencies in advertised routing data) may not be prohibitively expensive, even when there are multiple errors (or many colluding nodes) error-correction (analogous to detection of the actual sources of inconsistency and thereby attribute malicious intents) can be prohibitively expensive when many errors have to be tolerated (or when there are multiple colluding nodes).

Even under such an event, for example when node A is convinced that node B is malicious, perhaps the best that node A can do is to make an entry of this fact and in future ignore (drop) all packets sent by B. While theoretically many nodes detecting malicious intent of some node could jointly take the third step, such approaches may be very susceptible to denial of service (DoS) attacks.

In other words, the risk of an attacker getting *caught in the act*, and especially being *held accountable* for his deeds, can be very low in *practical* MH-AHNs. Thus there is acute need for increasing the resistance to tampering attempts, of devices taking part in such mutually co-operative activities.

4.1 BA Schemes for Ad Hoc Routing Protocols

A more common approach for broadcast authentication is by techniques employing asymmetric cryptography, or more specifically, digital signatures. Apart from obvious increase in computational complexity, another disadvantage of such an approach is the overheads that may be required for dissemination of public keys or bandwidth overheads for certificates that should accompany each signature. Furthermore, the use of computationally intensive techniques may also increase the susceptibility to simple denial of service attacks.

BA using PKPSs have two very important advantages over certificates based signature schemes. Firstly, the verification complexity is very low (while the complexity of signing is $O(n)$, where is the size of attacker coalition that should be tolerated, the verification complexity is $O(1)$). Secondly, there is no need for dissemination of certificates as the ID of a node itself is the "certified public key" - a node claiming to possess an ID A should back up its claims by demonstrating that it possesses the keys corresponding to $F(A)$.

Another approach to broadcast authentication using only symmetric cryptographic primitives is based on one-way hash chains [17] and delayed disclosure [2, 7, 24]. In such schemes, the source creates a one-way hash chain and uses a value from the hash chain to calculate the HMAC for a message to be authenticated. Later the pre-image of the value used is made public. At this point the verifiers are assured that the source that transmitted the first message was the one who released the pre-image as no one else can compute the pre-image (under a secure one-way function) of a disclosed value. Alternately, the HMAC may be based on a pre-image which is guaranteed to have *not* been made public *at the instant of time a verifier receives the message* with the HMAC. This requirement is referred to as the "security condition" (see [24] for more details). Verification of the security condition calls for some kind of time synchronization between the nodes.

One implication of the "security condition" is that messages signed in this fashion cannot be re-used. For example, if A receives a message M from B in the form of $M_A = [M \parallel HMAC_K(M)]$ at some time t when the security condition was satisfied (or K, the key used for the HMAC, could *not* have been made public before time t), while A is convinced of the authenticity of the message M, A cannot convince some other node (say C) of the authenticity of M_A at some time $t_1 > t$ (as K may have been made public before time t_1).

A more practical implication of the inability to re-use authenticated information is reduced efficiency of routing protocols. For instance in source routing based protocols where nodes along some verified path have authenticated information regarding the existence of such a path, such information cannot be authenticated by other nodes which may need to use the path (or a segment of the path).

Another disadvantage of hash chain based schemes is that the problem of dissemination of commitment keys can be even more expensive than dissemination of public key certificates, as commitment values need to be refreshed periodically (every time the hash chain expires).

However, certificate-less public key methods, or identity based encryption schemes (IBE) [3, 30] are seen as increasingly more suitable for ad hoc networks. For IBE schemes, like KPSs, the ID itself can serve as the "signed public key certificate." Unfortunately, they also suffer the problems inherent to public key methods - increased computational complexity (or increased susceptibility to DoS attacks). Furthermore, while for public key schemes like RSA it may be possible to substantially reduce verifier complexity (by choosing small public exponents), this is not possible for IBE based methods. Thus while the problem of dissemination of keys can be overcome, IBE techniques may exacerbate the problem of susceptibility to DoS attacks.

4.2 Collusion Resistance

The price paid for the advantages offered by PKPSs is their susceptibility to collusions. However, note that *irrespective* of the cryptographic technique used, an attacker who has exposed all secrets even from a few tens of devices can render the entire deployment inoperable for all practical purposes. For instance, the attacker could build an *unlimited* number of devices which could impersonate any of the (few tens) of devices. Such malicious devices could be geographically spread around[6].

The extent of attacks that can be inflicted by attackers with the compromised nodes (nodes from which the attacker has exposed all secrets from) is the same, irrespective of whether KPS is used or public key cryptography is used. However, *when an attacker has compromised a large number of nodes*, the use of KPSs opens up a *new line* of possible attacks for the attacker - impersonating nodes that have *not* been physically compromised using keys from nodes that have been compromised. Nevertheless, if the deployment is unusable if an attacker has compromised n (say a few tens) nodes irrespective of the KDS used, then as long as the KPS is n-secure, the lack of collusion resistance is not a serious disadvantage.

Another seemingly major disadvantage of KPS based BA schemes stems from the fact that when attackers have compromised many nodes, they may be able to engage in attacks *without* the risk of *disclosing* their actual identity (or the identity of the nodes they have compromised). If we employ public key based BA schemes however, if some inconsistent information is propagated by an attacker that carries the signature of some node A it means that node A is malicious (or its secret compromised by an attacker) and therefore has to be revoked. For KPSs on the other hand, that the signature of node A is appended to some malicious information could imply *either* 1) node A has been compromised, *or* 2) the attacker has compromised *many other* nodes.

We argued earlier that the problem of unambiguous attribution of malicious intents is infeasible in *any* case, especially when there are multiple colluding nodes[7] (as will be the case when an attacker has compromised many nodes to take advantage of the weakness of KPSs). Thus in practical AHNs the fact that the use of KPSs may render attribution (or the ability of nodes to unambiguously determine *which* node(s) is (are) responsible for spreading the malicious misinformation) infeasible, is *not* a severe disadvantage.

[6]Thus of the two assurances required (read-proofing and tamper-resistance) read-proofing is even more crucial. After all if a device has been tampered with to modify its behavior (or "morals"), the result is just one malicious device. On the other hand if all secrets have been exposed from a device, the result is an unlimited number of malicious devices whose "morals" are restricted only by the imagination of the attacker.

[7]Even detection of inconsistencies could be impossible - let alone unambiguously identifying the source of the error - when there are many colluding nodes and the protocol overheads are restricted.

4.3 HARPS-BA for Routing Protocols

Thus with mandatory measures for protection of secrets, PKPSs are an attractive option for securing ad hoc routing protocols. With such assurances, what we desire form PKPSs is *increased resistance to node compromises.* As was demonstrated in the previous section, the degradation of HARPS-BA is significantly more graceful than RPS-BA. Also note that the extent of attack an attacker can inflict depends on how many verifiers the attacker can fool with a particular "authenticated" message. With HARPS (compared to RPS) it is substantially more difficult to fool many verifiers.

As a more concrete example, let us consider a scenario where the nodes use HARPS-BA with $L = 64$, $P = 2048$, $\xi = 1/8$ (or $k = 256$ - each signature consists of kb bits on an average). In a situation where an attacker with 10 neighbors desires to impersonate another node, the attacker can fool one of his 10 neighbors if $p_F \approx 1/10$. To achieve this desired result the attacker has to compromise *all* secrets from about 28 nodes (compared to 20 nodes for RPS).

The extent of the damage that can be inflicted by an attacker depends on the number of nodes an attacker can fool. For fooling 50% / 90% / 99% of the nodes the number of nodes the attacker has to compromise is roughly 55 / 115 / 210 nodes for HARPS (compared to 28 / 42 / 60 for RPS-BA).

For most attacks on routing protocols, the attacker will need to know his capabilities (that he can impersonate some nodes for fooling some nodes) before he can carry out an attack. Thus for most situations the attacker cannot rely on "guessing" HMACs of bits corresponding to the keys he has not compromised. Note that if the attacker needs to guess some bits he will not even know a priori if his attack will be successful. Furthermore, for any attack to have a large enough impact the attacker will need to *consistently* propagate "authenticated" misleading information. In other words, the attacker cannot rely on *per-message forgability* probability to realize his goals. Thus using one-bit HMACs does not pose a severe disadvantage. Reducing b does not affect p_F - the probability that the attacker has access to all secrets a source and a verifier share - it only increases[8] the per-message forgability probability p'_F. Thus depending on the extent of confidence in our ability to provide assurances of read-proofing / tamper resistance, the bandwidth overheads for PKPSs "signatures" (ξP bits if $b = 1$) can be considerably lower than that of digital signatures.

4.3.1 BAP in Routing Protocols

We saw that BAP has no associated disadvantage - as the reduction in security is marginal even for non-preferred verifiers (BAP-NP) compared to plain BA, while significantly increasing the strength of the verification by pre-

[8]Note thatin Section 3.1 only the first four conditions **C1** to **C4** determine p_F. The size of b influences only **C5** and hence p'_F.

ferred verifiers. A "hidden" bandwidth overhead associated with BAP is that the BAP signature needs to explicitly *specify the list of targeted nodes.* However even this additional overhead may not be necessary in many situations. More specifically, in many situations the *context* of the message can unambiguously identify the choice of the targeted verifiers. Some examples of situations where the nodes to be targeted can be easily determined from the context are as follows:

1) For authentication of source routed packets all nodes in the path could be targeted.

2) In AHNs employing clustering, the cluster-head could be a preferred verifier for all messages.

3) In scenarios where link-state information is exchanged by neighbors (but not relayed further) it is possible to randomly designate some preferred verifiers (say a subset of its neighbors) for each source. The exact choice of the subset could also be dictated by a one-way function of the message that is authenticated to ensure that nodes cannot "choose" preferred verifiers. Such preferred verifiers could be afforded the power (temporarily and randomly) to broadcast notifications of authentication failure.

For the numerical example considered earlier ($P = 2048$, $\xi = 1/8$ and $L = 64$) if one verifier is deemed as a preferred verifier the attacker needs to compromise over 40 nodes to accomplish $p_F = 1/10$. However for fooling the *specific* preferred verifier with probability 1/2 the attacker has to compromise over 72 nodes (as opposed to just 28 nodes for RPS-BA with $p_F = 1/2$ and 55 for HARPS-BA). For fooling two specific preferred verifiers with $p_F = 1/2$ the attacker needs to expose all secrets from over 100 devices.

5 Conclusions

We argued that BA schemes based on probabilistic key pre-distribution schemes have many inherent advantages, especially for securing ad hoc routing protocols. We presented an efficient BA scheme, which also caters for a novel cryptographic paradigm of broadcast authentication with preferred verifiers (BAP), using hashed random pre-loaded subsets (HARPS).

It was shown that BAP can be substantially stronger against attempts by attackers to fool preferred verifiers, than general-purpose BA using HARPS. Simultaneously BAP is only *marginally* weaker than BA against attempts to fool other verifiers. The ability to impose preferred verifiers can be used advantageously in any ad hoc routing protocol for improving the strength of the authentication.

It was also quantitatively demonstrated that even for general purpose BA, the flexibility offered by HARPS to choose an optimal hash depth for the keys used for MACs, depending on the prevailing threat level, and its significantly more graceful degradation of security with node compromises, makes HARPS substantially stronger than similar schemes based on RPS [1, 4]). Specifically, the task of the attacker is rendered significantly more difficult when the attacker desires to fool multiple verifiers with the same message.

While we considered only peer-to-peer and peer-to"jointly-verifying-peers" scenarios for BAP, it is also possible for the KDC to employ BAP, explicitly specifying T independent preferred verifiers or J jointly-verifying preferred verifiers. Further, for our analysis of "jointly verified BAP" we assumed that all J joint verifiers are also preferred verifiers. In practice, it is also possible that only some of the joint verifiers are preferred verifiers. However, extensions of the analysis to all these cases are straight-forward.

Similar to the concept of explicitly targeting nodes, the source may also explicitly *exclude* nodes. This can be done by choosing hash depths to ensure that excluded nodes may not be able to verify the authentication data with high certainty (an excluded node will still be able to verify MACs corresponding to keys it shares for the source node, for which it has a lower hash depth than the source node). Yet another strategy may be to let the message itself dictate the hash depths that need to be used. Some of our current research efforts include investigation of different such strategies and their potential applications in securing ad hoc routing protocols.

The specific contributions of this paper are as follows

1) A more detailed analysis of the basic BA scheme proposed in [4] by Cannetti et al. While the focus of the basic scheme in [4] was broadcast by a single source, our focus was on broadcast by multiple sources[9] - or broadcast by peers. We also provided a more rigorous analysis of bandwidth - computational complexity trade-offs.

2) A novel BA scheme, HARPS-BA as an extension of the basic scheme in [4].

3) A novel cryptographic paradigm of BA with preferred verifiers

4) Extensive discussions of suitability of different types of broadcast authentication schemes for securing ad hoc routing protocols.

5) The use of the novel paradigm of BAP in ad hoc routing protocols.

References

[1] N. Alon, "Probabilistic methods in external finite set theory," in *Extremal Problems for Finite Sets,* pp. 39-57, 1991.

[9]While [4] also provides an extension of their basic scheme for broadcasts by multiple sources, it is not well suited for our application of focus - securing ad hoc routing protocols.

[2] R. J. Anderson, F. Bergadano, B. Crispo, J. H. Lee, C. Manifavas and R. M. Needham, " A New Family of Authentication Protocols," *ACM Operating Systems Review*, vol. 32, no. 4, pp. 9-20, 1998.

[3] M. Bohio and A. Miri, "An authenticated broadcasting scheme for wireless ad hoc network," in *Second Annual Conference on Communication Networks and Services Research (CNSR'04)*, pp. 69-74, 2004.

[4] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, "Multicast security: A taxonomy and some efficient constructions," in *Proceedings of INFOCOMM'99*, vol. 2, pp. 708-716, 1999.

[5] H. Chan, A. Perrig, and D. Song, "Random key pre-distribution schemes for sensor networks," *IEEE Symposium on Security and Privacy*, Berkeley, California, pp. 197-202, May 2003.

[6] D. Chaum, "Designated confirmer signatures," *Eurocypt 1994*, LNCS 950, pp. 86-91, Springer-Verlag, 1995.

[7] S. Cheung, "An efficient message authentication scheme for link state routing," in *Proceedings of the 13th Annual Computer Security Applications Conference*, San Diego, California, pp. 90-98, Dec. 1997.

[8] W. Du, J. Deng, Y. S. Han, and P. K .Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communication Security*, pp. 42-51, 2003.

[9] M. Dyer and T. Fenner, A. Frieze and A. Thomason, "On key storage in secure networks," *Journal of Cryptology,* vol. 8, pp. 189-200, 1995.

[10] P. Erdos, P. Frankl, and Z. Furedi, "Families of finite sets in which no set is covered by the union of r others," *Isreal Journal of Mathematics*, vol. 51, pp. 79-89, 1985.

[11] L. Eschenauer and V.D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the Ninth ACM Conference on Computer and Communications Security*, Washington DC, pp. 41-47, Nov. 2002.

[12] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, and T. Rabin, "Tamper Proof Security: Theoretical Foundations for Security Against Hardware Tampering," in *Theory of Cryptography Conference*, Cambridge, MA, pp. 258-277, Feb. 2004.

[13] L. Gong, and D. J. Wheeler, "A matrix key distribution scheme," *Journal of Cryptology*, vol. 2, no. 2, pp. 51-59, 1990.

[14] Y. C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," in *The 8th ACM International Conference on Mobile Computing and Networking*, Atlanta, Georgia, pp. 142-153, Sept. 2002.

[15] Y. C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks," in *Ad Hoc Networks*, **I**, pp. 175-192, 2003.

[16] M. Jakobsson, K. Sako and R. Impagliazzo, "Designated verifier proofs and their applications," *EUROCRYPT 1996*, LNCS 1070, pp. 143-154, Springer-Verlag, 1996.

[17] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770-772, Nov. 1981.

[18] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382-401, July 1982.

[19] T. Leighton and S. Micali, "Secret-key agreement without public-key cryptography," *Advances in Cryptology, CRYPTO 1993*, pp. 456-479, 1994.

[20] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," in *Proceedings of the 10th ACM Conference on Computer and Communication Security*, Washington DC, pp. 52-61, 2003.

[21] J. Lotspiech, S. Nusser, and F. Pestonoi, "Anonymous trust: Digital rights management using broadcast encryption," *Proceedings of the IEEE*, vol. 92, no. 6, pp. 898-909, 2004.

[22] C. J. Mitchell and F.C. Piper, "Key storage in secure networks," *Discrete Applied Mathematics*, vol. 21, pp. 215–228, 1995.

[23] P. Papadimitratos and Z. J.Haas, "Secure routing for mobile ad hoc networks," in *Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference(CNDS 2002)*, San Antonio, Texas, pp. 46-55, 2002

[24] A. Perrig, R. Canetti, D. Song, and D. Tygar, "Efficient and secure source authentication for multicast," in *Network and Distributed System Security Symposium, NDSS '01*, pp. 33-44, Feb. 2001.

[25] A. Pfitzmann, B. Pfitzmann, M. Schunter, and M. Waidner, *Mobile User Devices and Security Modules: Design for Trustworthiness*, IBM Research Report RZ 2784 (#89262) 02/05/96, IBM Research Division, Zurich, Feb. 1996.

[26] R. D. Pietro, L. V. Mancini, and A. Mei, "Random key assignment for secure wireless sensor networks," in *2003 ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 62-71, Oct. 2003.

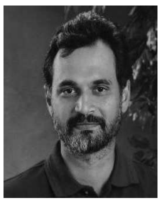
[27] M. Ramkumar and N. Memon "An efficient key predistribution scheme for MANET security," *IEEE Journal on Selected Areas of Communication*, vol. 23, no. 3, pp. 611-621, Mar. 2005.

[28] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, E. M. and Belding-Royer, "A secure routing protocol for ad hoc networks," in *Proceedings of 2002 IEEE International Conference on Network Protocols (ICNP)*, pp. 47-56, Nov. 2002.

[29] Web Link, http://www.ietf.org/html.charters/manet-charter.html

[30] Y. Zhang, W. Liu, W. Lou, Y. Fang, and Y. Kwon, "AC-PKI: Anonymous and certificateless public key infrastructure for mobile ad hoc networks," in *IEEE International Conference on Communications (ICC'05)*, Seoul, Korea, pp. 3515-3519, May 2005.

[31] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing pair-wise keys for secure communication in ad hoc networks: A probabilistic approach," George Mason University, Technical Report ISE-TR-03-01, Mar. 2003.



Mahalingam Ramkumar has been an Assistant Professor of Computer Science and Engineering in Mississippi State University, Mississippi, USA, since August 2003. He obtained his PhD in Electrical Engineering from New Jersey Institute of Technology in January 2000. He was a co-founder and Chief Technology Officer of PixWave.com, from March 2000 to September 2002. He was a Research Professor in the Department of Computer and Information Sciences Department, Polytechnic University, Brooklyn, NY, from September 2002 to August 2003.

His research interests include Cryptography, key distribution, security under resource constraints, ad hoc and sensor networks, digital rights management and data hiding.