A Lightweight Certificate-based Source Authentication Protocol for Group Communication in Hybrid Wireless/Satellite Networks

Ayan Roy-Chowdhury
Electrical and Computer Engineering
Institute for Systems Research
University of Maryland
College Park, MD 20742
Email: ayan@umd.edu

John S. Baras
Electrical and Computer Engineering
Institute for Systems Research
University of Maryland
College Park, MD 20742
Email: baras@isr.umd.edu

***Abstract*—In this paper, we describe an efficient authentication protocol for group communication in hybrid wireless networks with a satellite overlay. The proposed protocol uses a new class of lightweight, symmetric-key certificates called TESLA certificate. The certificates bind the identities of the senders to the anchor elements of their key chains; messages from the senders are authenticated by MACs computed with keys from the chain. The satellite is used as the Certificate Authority to generate the certificates. The satellite also acts as the proxy for the senders in disclosing the MAC keys to the receivers in the network. Due to the use of symmetric MAC functions, the proposed protocol is much less expensive in terms of node processing power and energy compared to digital signatures. The use of the satellite as the CA and the proxy allows strong security mechanisms and fast message verification. Through analysis, we show that the protocol is secure against malicious adversaries. We also estimate of the performance of the protocol in comparison to public key-based digital signatures.**

I. Introduction

Large wireless networks have the ability to provide rapid connectivity in disaster areas or military battlefields, or to inter-connect users in far-flung geographical locations. In [1], we have shown that the addition of a satellite overlay to such wireless networks can lead to a great improvement in the network performance. We term this network architecture a *hybrid network*, which has clusters of terrestrial wireless nodes with dual satellite connectivity providing alternate high-bandwidth and robust forwarding paths through satellite links.

Security is a necessary parameter in hybrid wireless networks to protect the communication amongst user nodes from unauthorized access or unauthorized modifications. Many envisioned applications for hybrid networks are collaborative in nature, and it is necessary to ensure that routing control messages and the application data between communicating nodes are properly authenticated to *enable* communication. In this paper we therefore focus on source authentication and associated message integrity protocols for group communication.

Authentication in group communication is traditionally done by digital signatures [2], based on public-key cryptography. However, generation and verification of digital signatures for messages are expensive in CPU cycles and therefore energy expenditure [3]–[5]. In wireless networks where many nodes have limited processor power, storage capacity and available energy, performing digital signature generation and verification frequently can prevent the CPU from other functions and drain the battery quickly. Therefore in hybrid wireless networks it is preferable to use authentication protocols that are based on symmetric cryptographic primitives like Message Authentication Code or MAC (for example, HMAC [6]) - which expend less node energy. However, designing authentication protocols for group communication using symmetric cryptography is a significant challenge. The primary difficulty in designing a scalable scheme is how to create the asymmetry efficiently such that receivers can authenticate the messages without heavy expenditure of system resources.

We propose to achieve authentication using a new class of certificates called *TESLA Certificate*. The TESLA certificate concept was originally proposed in [7], and we have suggested modifications and extensions to it in [8]. We build on our modified TESLA certificate design and take advantage of the presence of the satellite overlay network to propose an authentication protocol for hybrid networks.

In the proposed protocol, source authentication using TESLA certificate is based on MAC computation using keyed hash functions, with delayed disclosure of the key by the Certificate Authority (CA), to achieve the asymmetry required for authentication in group communication. Due to the use of MACs to generate and verify certificates, the scheme is fast, has low processing overhead, and consumes much less energy than digital signature algorithms. The protocol also avoids the assumption that the user nodes have some sort of security association established *a priori*, as many other protocols assume.

The rest of this paper is organized as follows. In section II, we briefly review the TESLA authentication protocol on which TESLA certificate is based. The TESLA certificate algorithm is described in III. We describe the proposed source authentication protocol in section IV.We analyze the security

978-1-4244-2324-8/08/$25.00 © 2008 IEEE.
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2008 proceedings.

$$\xleftarrow{\text{Sequence of key generation}}$$

$$\begin{tikzcd}
s_0 \arrow[d, "F_2"] & s_1 \arrow[l, "F_1"'] \arrow[d, "F_2"] & \dots \arrow[l, "F_1"'] & s_{n-1} \arrow[l, "F_1"'] \arrow[d, "F_2"] & s_n \arrow[l, "F_1"'] \arrow[d, "F_2"] \\
K_0 & K_1 & \dots & K_{n-1} & K_n
\end{tikzcd}$$

$$\xrightarrow{\text{Sequence of key usage}}$$

$$\xrightarrow{\text{Time}}$$

Fig. 1. TESLA key generation

of the proposed protocol in section V. A preliminary analysis of the protocol performance is given in VI. We briefly review related work in authentication algorithms in VII. We conclude with a discussion of our current research efforts in section VIII.

II. Review of TESLA Authentication Protocol

The TESLA source authentication protocol [9] achieves asymmetric authentication between a source and receivers through the use of symmetric cryptographic MAC functions. The asymmetry is obtained through the *delayed disclosure* of the authentication keys.

TESLA divides the time of transmission by the source into n intervals of equal duration. The source generates a random key seed s_n for interval n, and computes a one-way hash chain by repeatedly applying a public one-way function F_1 to s_n. The number of elements of the hash chain correspond to the number of intervals in which the source transmits. The source computes the MAC key for each interval by applying a second public one-way function F_2 to each element of the hash chain. The algorithm is illustrated in fig. 1.

The sender uses the keys in the reverse order of their generation, that is, starting with K_1 in interval 1, followed by K_2 in interval 2, and so on. The sender bootstraps the hash chain by broadcasting to all the receivers the anchor element of the chain, s_0. For each packet generated in time slot i, the sender uses the authentication key K_i to compute a MAC on the packet. The sender discloses K_i at a later time instant by broadcasting the corresponding key seed s_i. Upon receiving s_i, each receiver first verifies the authenticity of s_i by checking $s_i \xrightarrow{F_1} s_{i-1} \xrightarrow{F_1} ... \xrightarrow{F_1} s_0$. If s_i verifies correctly, each receiver can compute K_i: $s_i \xrightarrow{F_2} K_i$ and subsequently use K_i to verify the MAC on the packets received during interval i.

s_i is disclosed only d time slots after i so that no malicious node can compute K_i and forge packets in the intervening period. This is the principle of delayed disclosure of keys.

III. Review of TESLA Certificate Algorithm

In the algorithm described in [7], there is a certificate authority CA who creates certificates for an entity B. During time slot n, the CA generates authentication key aK_{B_n} for B to use to compute the MAC on its messages in that interval. The CA creates a certificate $Cert_{CA_n}(B)$ to bind aK_{B_n} to B for interval n. The CA uses its TESLA key tK_{CA_n} to encrypt aK_{B_n} in the certificate, and uses the same key to compute a MAC on the certificate:

$$Cert_{CA_n}(B) = \\ (ID_B, \{aK_{B_n}\}_{tK_{CA_n}}, n+d, MAC_{tK_{CA_n}}(..)) \quad (1)$$

aK_{B_n} is known only to the CA and B during period n, while tK_{CA_n} is known only to the CA. $n+d$ indicates the time at which the CA will disclose tK_{CA_n} to the nodes, that is, it is the expiration time of the certificate. The CA sends $Cert_{CA_n}(B)$ to B alongwith aK_{B_n}, which is encrypted with key $K_{CA,B}$ that is shared between the CA and B.

In the time interval $\langle n, n+d \rangle$, D sends a request to B for using B's service: $D \rightarrow B$: $(request)$. To authenticate itself to D, B sends an authentication packet containing its certificate and a MAC on the request: $B \rightarrow D$: $(Cert_{CA_n}(B), MAC_{aK_{B_n}}(request))$. When D receives the authentication message, it checks the timestamp of $Cert_{CA_n}(B)$ to make sure it has arrived before time $n+d$. At time $n+d$, the CA discloses tK_{CA_n}. Upon receiving the key, D verifies $Cert_{CA_n}(B)$ by checking the MAC in the certificate using tK_{CA_n}. If the MAC verifies correctly, D obtains aK_{B_n} from the certificate by decrypting with tK_{CA_n}. Subsequently, D checks $MAC_{aK_{B_n}}(request)$ to verify the authenticity of B.

A TESLA certificate allows a node to add authentication to packets for a single period in time. Therefore, a source node B that transmits for multiple time intervals will need several TESLA certificates from the CA. If there are many sources that send data over long intervals, this can add up to a substantial overhead.

IV. Source Authentication Protocol Description

A. Extensions to TESLA Certificate

In the proposed authentication protocol, we use the TESLA certificate concept with the following important modifications:

- we extend the lifetime of the TESLA certificate from single use to multiple uses by combining key chains with the certificate, and
- we disclose TESLA keys used by the source node via proxy, namely, by allowing the satellite to broadcast the sender's MAC keys to the receivers.

Using the satellite as the proxy for the terrestrial nodes for TESLA MAC key disclosure, reduces the delay involved in message authentication, and also mitigates the processing load of the senders.

B. The Satellite as the CA

The TESLA certificate algorithm requires the presence of a dedicated Certificate Authority (CA) to generate the certificates. In our protocol, the CA broadcasts the TESLA keys of the source nodes to the network at periodic key disclosure intervals. In the hybrid network topology, we use the satellite for providing CA services. The reasons for using the satellite as the CA are: (i) the satellite is a network node that is always available, connected to the entire network, and is physically secure; (ii) the satellite has higher computing power with on-board processing capability and higher storage compared to terrestrial wireless nodes; and, (iii) its energy is renewable via solar power. Therefore the satellite can perform processing-intensive cryptographic operations more efficiently compared to the terrestrial nodes, and without depleting its energy. We thus consider the satellite as the root CA in our authentication protocol design.

C. Protocol Assumptions

In order to describe the operation of the authentication protocol, let us consider a group of three wireless nodes A, B and C, where A sends authenticated messages to B and C.

We make the following assumptions about the initial security setup of the network:

- all terrestrial nodes have limited energy and processing power, and none has any pre-existing security information about the others;
- the satellite is trusted by all other nodes;
- the public key $+K_{CA}$ of the CA is available to all nodes;
- all nodes are time-synchronized with the CA;
- each node X shares a unique secret key $K_{CA,X}$ with the CA;
- one-way functions F_1 and F_2 are publicly available;
- message transmission from A to B, C start at time t_0;
- time is divided into intervals, each of duration Δ.

D. Setup: Key Chain Generation by CA and Source Node

During the initial setup, before any messages are transmitted in the network, the CA and all sources generate the key chains that each use for message authentication.

The CA uses a key chain $\{tK_{CA,i}\}$ where $i = 1,..,N$ to authenticate the TESLA certificates that it creates. The CA starts with a random seed $s_{CA,N}$ and applies one-way function F_1 to $s_{CA,N}$ to form a hash chain (2):

$$s_{CA,0} \xleftarrow{F_1} s_{CA,1} \xleftarrow{F_1} ... \xleftarrow{F_1} s_{CA,N-1} \xleftarrow{F_1} s_{CA,N} \quad (2)$$

where $N > 0$ and its value depends on the length of each time interval and the total duration that the CA node will be present. We assume that in each time interval, the CA uses only one key for computing the MACs on all the messages it generates in that time interval Therefore, if the total time of CA's functionality is T and the interval for key disclosure is d, we have $N = \frac{T}{d}$.

Subsequently the CA applies function F_2 to each element of (2) to obtain $tK_{CA,i}$:

$$\begin{tikzcd} s_{CA,0} & s_{CA,1} \arrow[l, "F_1"'] \arrow[d, "F_2"] & ... \arrow[l, "F_1"'] & s_{CA,N} \arrow[l, "F_1"'] \arrow[d, "F_2"] \\ & tK_{CA,1} & ... & tK_{CA,N} \end{tikzcd} \quad (3)$$

$s_{CA,0}$ is the *anchor element* of the CA's authentication key chain. All TESLA certificates and signed messages from the CA are authenticated using the anchor element during the protocol run. $s_{CA,0}$ is broadcast to the network in time $t < t_0$ (4):

$$CA \rightarrow network\colon\ (s_{CA,0}, SIGN_{-K_{CA}}(..)) \quad (4)$$

The anchor element is authenticated using a digital signature. All network nodes receiving the broadcast message can verify the signature on the message using the public key $+K_{CA}$ of the CA. If the signature is verified, the nodes store $s_{CA,0}$ in local memory.

In a manner similar to the above, each source node A generates a random seed $s_{A,n}$ and applies one-way function F_1 to $s_{A,n}$ to form a hash chain. A subsequently applies F_2 to each key $s_{A,i}$ of the chain and obtains $s'_{A,i}$ (5).

$$\begin{tikzcd} s_{A,0} \arrow[d, "F_2"] & s_{A,1} \arrow[l, "F_1"'] \arrow[d, "F_2"] & ... \arrow[l, "F_1"'] & s_{A,n-1} \arrow[l, "F_1"'] \arrow[d, "F_2"] & s_{A,n} \arrow[l, "F_1"'] \arrow[d, "F_2"] \\ s'_{A,0} & s'_{A,1} & ... & s'_{A,n-1} & s'_{A,n} \end{tikzcd} \quad (5)$$

Here $n > 0$ is equal to the number of unique MAC keys that A expects to use for authenticating its messages. We assume that in each time interval Δ, a source uses only one key for computing the MACs on all the messages it generates in that time interval Therefore, if the total time of A's transmission is T, we have $n = \frac{T}{\Delta}$.

At time $t < t_0$, A sends $s_{A,n}$ and n to the CA, alongwith details on A's key disclosure interval. The message from A to the CA is secured using the shared secret $K_{CA,A}$ between A and the CA. The CA can obtain all the elements of A's TESLA key chain from $s_{A,n}$ and n, as in equation (5).

On successful verification of A's identity, the CA generates the TESLA certificate for A. The key $s_{A,0}$ is included in the certificate as the anchor element of A's key chain. It is encrypted using key $tK_{CA,1}$ from the CA's key chain. The certificate also includes the identity of the source node A and the time $t_0 + d$ upto which the certificate is valid, i.e., after time t_0+d, key $s_{A,0}$ is made public to the group and it can no longer be used for new messages. The certificate also contains a MAC for authentication, computed on the previous elements using $tKCA, 1$. For added security, the certificate might also contain CA's public-key signature on all the previous elements (6).

$$Cert_{CA}(A) = (ID_A, \{s_{A,0}\}_{tK_{CA,1}}, t_0 + d, MAC_{tK_{CA,1}}(..), SIGN_{-K_{CA}}(..)) \quad (6)$$

$$CA \rightarrow A : Cert_{CA}(A) \quad (7)$$

Here $d \geq \Delta$ is the key disclosure delay for the CA TESLA signature key, and $tK_{CA,1}$ is the CA MAC key for the time period $\langle t_0, t_0 + d\rangle$. A schematic of the TESLA certificate for A is given in figure 2.

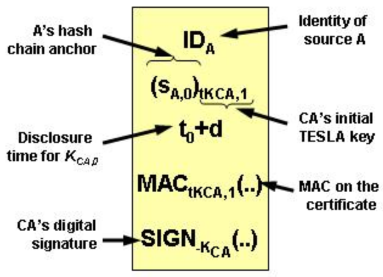

Fig. 2. TESLA Certificate for node A

E. Message Transmission from Source to Receiver

A sends messages to B and C starting in the time interval $\langle t_0, t_0 + d\rangle$. A computes a MAC over the message m_0 using $s'_{A,0}$ and includes its TESLA certificate $Cert_{CA}(A)$ with the message.

$$A \rightarrow \{B, C\}\colon \{M_0 | M_0\colon \left(m_0, MAC_{s'_{A,0}}(m_0), Cert_{CA}(A)\right)\} \quad (8)$$

Each of B and C checks the *freshness* of the certificate by checking the timestamp of $Cert_{CA}(A)$ to make sure it has arrived within the period $\langle t, t_0 + d\rangle$. The receivers also check that $s'_{A,0}$ is not publicly known, i.e., $MAC_{s'_{A,0}}(m_0)$ cannot yet be computed by them. If all the checks pass, B and C store M_0 in their respective buffers, else they discard the message.

F. Message Authentication at Receiver

At time $t_1 = t_0 + d$, the CA broadcasts the key $s_{CA,1}$:

$$CA \rightarrow network\colon\ (\langle t_0, t_0 + d\rangle, s_{CA,1}, SIGN_{-K_{CA}}(..)) \quad (9)$$

If receiver B or C has received the anchor element $s_{CA,0}$ (4), they can check the authenticity of $s_{CA,1}$ by verifying $s_{CA,1}$ against $s_{CA,0}$: $s_{CA,1} \xrightarrow{F_1} s_{CA,0}$. Otherwise, B or C can verify $s_{CA,1}$ from the signature using $+K_{CA}$. If verification is successful, each receiver derives $tK_{CA,1}$ from $s_{CA,1}$ (3) and uses $tK_{CA,1}$ to verify the MAC on $Cert_{CA}(A)$. If the MAC is correct, each receiver obtains $s_{A,0}$ from $Cert_{CA}(A)$ by decrypting with $tK_{CA,1}$. B or C obtains $s'_{A,0}$ from $s_{A,0}$ (5), then checks $MAC_{s'_{A,0}}(m_0)$ using $s'_{A,0}$ and accepts m_0 if the MAC verifies correctly. Each receiver saves $Cert_{CA}(A)$ and the anchor element $s_{A,0}$ of A's key chain in long-term memory - they are used for authenticating future keys and messages from A.

Messages from A to B and C in subsequent time intervals are authenticated using the corresponding key of A's key chain. A does not have to include its TESLA certificate in messages subsequent to M_0, under the assumption that every receiver has received M_0 correctly. For example, in the period $\langle t_i, t_i + \Delta\rangle$, message M_i from A to B would look like:

$$A \rightarrow B\colon \{M_i | M_i\colon \left(m_i, MAC_{s'_{A,i}}(m_i)\right)\} \quad (10)$$

At time $t_i + d$, *the CA broadcasts $s_{A,i}$ to the network.* Since $d > \Delta$, when $s_{A,i}$ is disclosed, A is no longer using $s'_{A,i}$ for computing the MACs on its messages. Any receiver B that receives the CA broadcast, verifies that $s_{A,i}$ indeed belongs to A's MAC key chain as: $s_{A,i} \xrightarrow{F_1} s_{A,i-1} \xrightarrow{F_1} \ldots \xrightarrow{F_1} s_{A,0}$ where $s_{A,0}$ has already been verified from $Cert_{CA}(A)$. Figure 3 gives a timing diagram representation of the protocol.

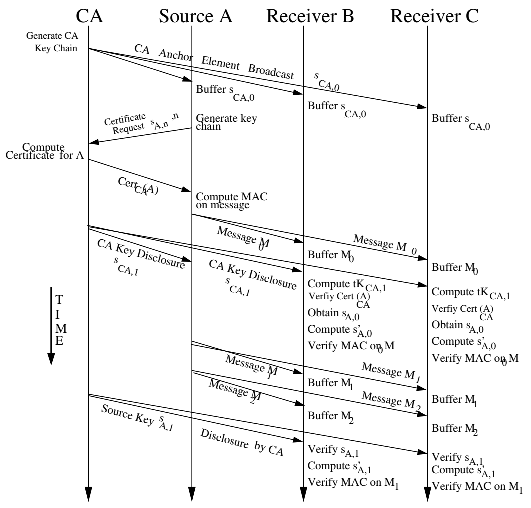

Fig. 3. Authentication protocol: time diagram

After the initial anchor element broadcast message from the CA signed with $-K_{CA}$, subsequent key disclosure messages from the CA can be authenticated using one-way chains. For example, CA discloses the key $s_{CA,i}$ used in period $\langle t_i, t_i + d\rangle$ at time $t_i + kd$, $k > 0$. Receiver B can verify that $s_{CA,i}$ belongs to CA's one-way chain:

$$s_{CA,i} \xrightarrow{F_1} s_{CA,i-1} \xrightarrow{F_1} \ldots \xrightarrow{F_1} s_{CA,0} \quad (11)$$

where $s_{CA,0}$ has been verified before using $+K_{CA}$.

The algorithm requires that all terrestrial nodes are able to communicate directly with the satellite, as shown in figure 3. Due to the broadcast nature of the satellite transmission, and its large footprint, that is a valid assumption to make. It should be noted, however, that only the source nodes need to transmit to the satellite, all other nodes only receive from satellite transmission. Therefore, the source nodes need to have additional energy for transmitting. In situations where that is a constraint, the network architecture can be slightly modified to include one or more higher-powered *gateway* nodes through which all uplink transmissions to the satellite are done. Our proposed algorithm will work in the modified architecture with minor additions.

V. Security Analysis: Prevention of Authentication Attacks

We consider the case where a malicious node X in the network attempts to create fake packets from a source to the receiver(s). Without loss of generality, we consider one source A is sending autheticated data to one receiver B. We assume that X can hear packet transmissions from A, and can also transmit to B. X can also receive the broadcast messages from the CA. Therefore, shortly after time $t_0 + d$, X has knowledge of $Cert_{CA}(A)$, message M_0 from A to B, $s_{CA,0}$ broadcast by

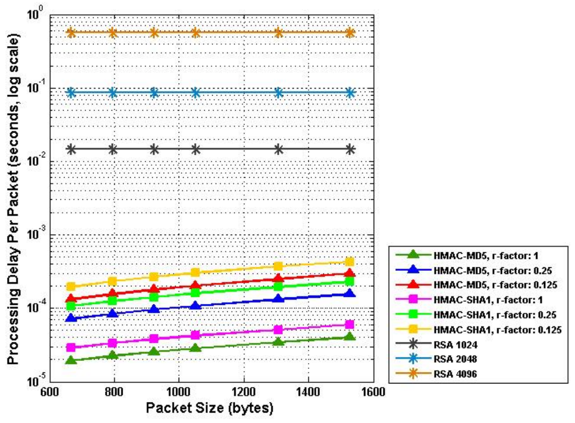

Fig. 4. Comparison of authentication processing delay

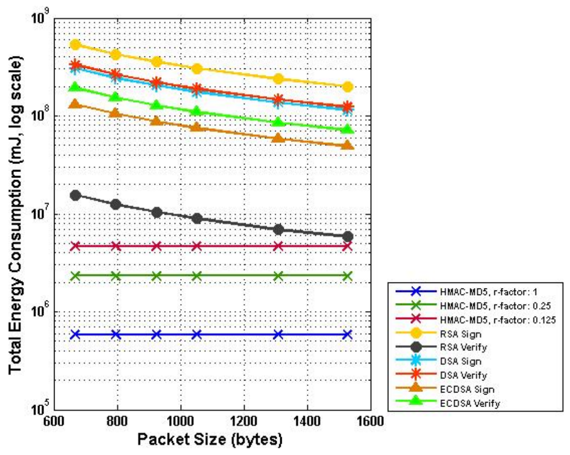

Fig. 5. Energy required for authenticating 500MB message

the CA, and $s_{A,0}$ from the certificate. X can verify that $s'_{A,0}$ belongs to the authentication hash chain of A by performing the verification procedure. Having obtained a verified element of A's authentication chain, X can attempt to spoof messages as coming from A, starting at time $t_0 + kd$, where $k > 0$. To achieve this, X needs to generate $s_{A,k}$ from $s_{A,0}$ where $s_{A,k} = F_1^{-k}\{s_{A,0}\}$. Due to the one-way property of F_1, this is of complexity $O\left(2^K\right)$, where each element of the hash chain is K bits and K is large (128 to 160 bits). This is thus computationally infeasible for X.

If X attempted to generate a fake certificate for A as in (6), or spoof CA key disclosure broadcast message similar to (9), it will need to know the private key $-K_{CA}$ of the CA with which the certificates and the CA anchor element broadcast message are signed. As per our assumption of the security of the CA, $-K_{CA}$ is known only to the correct CA, and therefore X would not be successful in this attack.

VI. Performance Analysis of extended TESLA Certificate Algorithm

Compared to asymmetric source authentication using public-key cryptography, our algorithm offers much greater savings in power expenditure and processing delay for authentication. An analysis of the processing delay overhead of the proposed protocol, used with HMAC-MD5 or HMAC-SHA1, and its comparison to the processing delay for RSA signatures, is given in figure 4, for a 500MB message on a 500MHz PentiumIII machine. The delay figures for HMAC-MD5 and HMAC-SHA1 are computed based on the approximation that each operation is executed in one processor clock tick for the 500MHzPIII processor. The delay figures for RSA for the 500MHzPIII processor are from [10]. The *r-factor* in the graphs refer to the degree of non-repudiation provided by a probabilistic non-repudiation algorithm that we use, and is related to the number of TESLA MACs attached to each message. r-factor 1 indicates one MAC per message, which is described in the algorithm presented in this paper; while r-factor 0.25 refer to 4 MACS per message, and so on.

Figure 5 compares the amount of energy that would be required to authenticate a 500MB message on a representative handheld computer [11] for different authentication algorithms. The base figures for energy expenditure of different cryptographic operations of the handheld are obtained from [12]. Clearly, authenticating a 500MB message without additional energy sources is not possible except for the extended TESLA protocol. The graphs validate our claim that the energy consumption of the extended TESLA protocol is significantly less in comparison to standard signature protocols. Here we assume that upto 50% of the node energy is spent in authenticating the data packets. Figure 5 compares only authentication protocols; the graphs do not include the energy expense of source transmission to the satellite, or nodes receiving from the satellite. However, transmission/reception is a constant expense irrespective of the protocol used.

In the proposed protocol, the source performs one public-key signature verification (6), and each receiver performs one public-key signature verification in either (4) or (9). Neither any source nor any receiver has to generate public-key signatures at any time. All other messages from the CA and the sources are authenticated using symmetric MACs. Compared to authentication using digital signatures, this represents a substantial savings in computation power and delay. Moreover, sources and receivers do not have to perform clock synchronization directly with one another, synchronizing with the CA is a necessary and sufficient condition for the protocol. This saves additional message rounds and protocol complexity, and also breaks the cyclical dependency between authentication and clock synchronization.

We have done further simulations to evaluate the protocol performance. Due to space constrains, they are not shown here but will appear in a future publication.

VII. Related Work

There has been significant research on efficient multicast source authentication algorithms based on symmetric cryptography that attempt to minimize the computation expense on the devices. Canetti et al [13] proposed one of the early solutions to use symmetric MACs for multicast source authentication. In their scheme, the source has l keys and computes l MACs on each packet. Each recipient holds a subset of the l keys, and verifies the MAC according to the keys it holds. The authentication protocol has probabilistic security and also requires the multicast group members to store a large number of keys. [14] has proposed a method known as *stream signing*, where one regular digital signature is transmitted at the beginning of a stream, and each packet either contains a cryptographic hash of the next packet, or a one-time public key using which the one-time signature on the next packet can be verified. This approach requires reliable packet transmission, since the loss of even one packet means that the information required to authenticate future packets will be lost. For most multicast protocols, such reliability cannot be guaranteed, since the transmission protocol is UDP, which is best-effort. [15] has proposed an approach where the source collects the packets in a time-interval into an authentication tree and signs the root of the tree. The root signature and hash information on the nodes of the tree are included in each transmitted packet. The signing and verification operations are thus amortized over many packets, and the protocol operations are one to two orders of magnitude faster compared to individual packet signatures. Rohatgi has proposed a hybrid scheme [16] using off-line/on-line signature generation scheme for creating k-time public/private key pairs so that the cost of signature generation can be amortized over k signatures. The size overhead of the proposed scheme is still considerable on a per-packet basis (of the order of 300 bytes per packet). Anderson et al have proposed the Guy Fawkes protocol in [17], where each message contains a hash commitment to a secret that is revealed in the next message. The secrets are not related to one another, and the authentication mechanism cannot tolerate packet losses - if a commitment is

Research in security for satellite networks have focused mostly on key management and data encryption for dynamic multicast groups in satellite networks [18], [19], or on key distribution and secure communication for both unicast and multicast [20]. None of the papers have addressed the problem of source authentication for group communication in satellite networks.

VIII. Conclusion

In this work, we have proposed a source authentication protocol for group communication in wireless/satellite hybrid networks. The protocol uses efficient, symmetric-key based TESLA certificates and takes advantage of the presence of the satellite overlay network to delegate processing-intensive operations to the secure satellite node and also reduces the delay in authentication by allowing the satellite to broadcast the sender's keys. The protocol is therefore suitable for resource-constrained nodes in hybrid satellite/wireless networks.

Acknowledgment

The material is based upon work supported by National Aeronautics and Space Administration under award No. NCC8235. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Aeronautics and Space Administration.

References

[1] A. Roy-Chowdhury and J. S. Baras, "Improving network performance in hybrid wireless networks using a satellite overlay," in *Proc. 13th Ka and Broadband Communications Conference*. Turin, Italy: Istituto Internazionale delle Comunicazioni (IIC), September 24-26 2007.

[2] N.I.S.T., "Digital signature standard (dss)," May 19 1994.

[3] P. Prasithsangaree and P. Krishnamurthy, "On a framework for energy-efficient security protocols in wireless networks," *Elsevier Computer Communications*, vol. 27, pp. 1716–1729, 2004.

[4] S. Seys and B. Preneel, "Power consumption evaluation of efficient digital signature schemes for low power devices," in *Proc. 2005 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (IEEE WiMob 2005)*, vol. 1. IEEE, 2005, pp. 79–86.

[5] W. Freeman and E. Miller, "An experimental analysis of cryptographic overhead in performance-critical systems," in *Proc. 7th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOT'99)*. College Park, MD, USA: IEEE, October 1999, pp. 348–357. [Online]. Available: citeseer.ist.psu.edu/freeman99experimental.html

[6] H. Krawczyk, M. Bellare, and R. Canetti, *HMAC: Keyed-Hashing for Message Authentication*, IETF RFC 2104, February 1997.

[7] M. Bohge and W. Trappe, "TESLA certificates: an authentication tool for networks of compute-constrained devices," in *Proc. of 6th international symposium on wirless personal multimedia communications (WPMC '03)*, Yokosuka, Kanagawa, Japan, October 2003.

[8] A. Roy-Chowdhury and J. Baras, "A certificate-based light-weight authentication algorithm for resource-constrained devices," Center for Satellite and Hybrid Communication Networks, University of Maryland College Park, Tech. Rep. CSHCN TR 2005-4, 2005.

[9] A. Perrig, R. Canetti, D. Song, and J. D. Tygar, "The TESLA broadcast authentication protocol," *RSA Cryptobytes*, Summer 2002.

[10] X. Ding, D. Mazzocchi, and G. Tsudik, "Equipping smart devices with public key signatures," *ACM Trans. Internet Technology*, vol. 7, no. 1, p. 3, 2007.

[11] "Compaq iPAQ Pocket PC H3600 series," http://h18002.www1.hp.com/products/quickspecs/10632_div/10632_div.HTML#%QuickSpecs.

[12] N. Potlapally, S. Ravi, A. Raghunathan, and N. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 2, pp. 128–143, Feb. 2006.

[13] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, "Multicast security: A taxonomy and some efficient constructions," *Proceedings of INFOCOMM '99*, March 1999.

[14] R. Gennaro and P. Rohatgi, "How to sign digital signatures," in *Advances in Cryptology - CRYPTO '97*. Springer-Verlag Berlin Heidelberg, 1997, pp. 180–197.

[15] C. Wong and S. Lam, "Digital signatures for flows and multicasts," in *Proc. IEEE ICNP '98*. Austin, USA: IEEE, October 1998.

[16] P. Rohatgi, "A compact and fast hybrid signature scheme for multicast packet authentication," in *Proc. Computer and Communications Security Conference (CCS'99)*. Singapore: ACM, 1999.

[17] R. Anderson, F. Bergadano, B. Crisp, J. Lee, C. Manifavas, and R. Needham, "A new family of authentication protocols," *ACM Operating Systems Review*, vol. 32, no. 4, pp. 9–20, 1998.

[18] M. P. Howarth, S. Iyengar, Z. Sun, and H. Cruickshank, "Dynamics of key management in secure satellite multicast," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 2, pp. 308–319, February 2004.

[19] H. Cruickshank, M. Howarth, S. Iyengar, Z. Sun, and L. Claverotte, "Securing multicast in DVB-RCS satellite systems," *IEEE Wireless Communications*, pp. 38–45, October 2005.

[20] L. Duquerroy, S. Josset, O. Alphand, P. Berthou, and T. Gayraud, "SatIPSec: an optimized solution for securing multicast and unicast satellite transmissions," in *22nd AIAA International Communications Satellite Systems Conference and Exhibit 2004*, no. AIAA-2004-3177, Monterey, California, 9-12 May 2004.