AUTHENTICATED AND EFFICIENT KEY MANAGEMENT FOR WIRELESS AD HOC NETWORKS

Stefaan Seys and Bart Preneel

Katholieke Universiteit Leuven, ESAT–SCD/COSIC
Kasteelpark Arenberg 10, B–3001 Heverlee, Belgium
{Stefaan.Seys, Bart.Preneel}@esat.kuleuven.ac.be

In this paper we present a key management protocol for wireless ad hoc multi-hop networks. Two objectives were crucial in our design: (1) distributed trust to ensure robustness, and (2) strong authentication to prevent the battery drain attack. We achieve distributed trust by presenting a hierarchical and distributed public key infrastructure for ad hoc networks. Our PKI has been designed to map onto hierarchical ad hoc networks, while maintaining global connectivity and flexibility. If a misbehavior detection scheme is present on the network, then the security of our PKI can be improved through collaboration with this scheme. Next to this PKI we propose a mechanism to securely establish and maintain link keys between the different nodes in the network.

1. INTRODUCTION

In an ad hoc network, there is no fixed infrastructure such as name servers or switches to set up connections. A new connection is created as soon as a mobile device (referred to as node) enters the vicinity of one or more other nodes. Mobile nodes that are within each other's radio range communicate directly through wireless links, while those that are further apart rely on other nodes to redirect and forward their messages (multi-hop routing). We allow that the wireless nodes can move around, this will not always be the case, but generally the nodes in the network will be portable and can move in and out the network at all time. Although our protocol is suited for general wireless ad hoc multi-hop networks, we will focus on distributed sensor networks (DSNs) [1, 3, 10, 13] to describe our key management scheme. These wireless ad hoc networks will typically consists of 1000's of low power nodes, with limited communication means and CPU power. A typical application we envision, is for example pollution monitoring in the soil

or in sewers. The government or a company could drop a batch of sensors in the sewer and use them to track the pollution, following the flow of the sewer. We assume that these sensor nodes are bought in large batches from well-known vendors that cannot afford to sell malicious nodes. In this way we can be assured that the initial set of deployed nodes are well-behaved.

A lack of security in sensor products and ad hoc networks in general can potentially inhibit large-scale adaptation, since users are rightly concerned about hackers compromising their home and their privacy. Privacy and integrity are nice, but keeping the network service running is more important. In this paper we propose a key management protocol that is designed to be resistant against denial of service attacks. Two objectives were crucial in our design: (1) distributed trust to ensure robustness, and (2) strong authentication to prevent the battery drain attack, first mentioned by Stajano and Anderson [16]. A misbehaving node can mount this denial-of-service attack by routing seemingly legitimate traffic through a number of nodes in an attempt to wear down the batteries of the other nodes.

Miniature Sensor Nodes and Control Nodes. A sensor network typically consists of at least the following elements: *sensor nodes* that collect and process environmental data (also called source nodes), and *control nodes* that query the network for information, run control algorithms, and command specific actions to be performed. These nodes can also act as bridge nodes, allowing the sensor network to interact with traditional wired or wireless networks.

The majority of the nodes are miniature sensor nodes, spread randomly over the target area. These sensor nodes have the following properties – (1) Low power and Peanut CPU: since the sensor nodes have to be small, they will have an equally small battery or solar panel. Moreover the battery will not be replaceable since the nodes may be physically unreachable once they have been deployed. The computational power of the sensor nodes is equally limited. (2) Low range and low bit rates: this is a direct consequence of the low power constraint; the energy needed for transmission is roughly equivalent with the fourth power of the distance ($E \approx d^4$) of the transmission. The second type of nodes we consider are control nodes with additional computational power and energy supplies.

All nodes in the network are equal concerning the data link layer, this does not mean that we cannot have a *functional* hierarchical structure, i.e., sensor and control nodes. In fact our design is based on a functional hierarchy *with an arbitrary number of levels.*

Security Threats. Ad hoc networks are susceptible to the same threats as more conventional networks: passive eavesdropping, active impersonation, message replay and distortion, etc. The specific properties of ad hoc networks do have an impact on the security requirements though. Denial of service, for example, is no longer only a matter of network connectivity and installing new patches for security bugs. Battery exhaustion could effectively destroy a network node if recharging is impossible. Another issue is the relatively poor physical protection of deployed nodes. Some nodes will probably be easy to capture and compromization of secret information on these nodes cannot be ruled out. Another consequence is that using a single certification authority or key distribution center may result in system failure if this single node is compromised or destroyed. On the other hand, ad hoc network have inherent link redundancy and this can be exploited to improve robustness of the system.

Security Goals. The main goal of our protocol is to securely establish and manage cryptographic keys in DSNs. The protocol has been designed to achieve the following goals:

- Sustain link attacks ranging from passive eavesdropping to active impersonation and message replay.
- Once sensor nodes are deployed in the field, they may be compromised. Therefore, we also consider attacks launched from within the network by compromised nodes.
- Secret information (keys) can be extracted from stolen nodes. This should not lead to network wide security compromization (as is the case with solutions that depend on a system wide mission key). This also means that no single node is trustworthy, but we can trust an aggregation of nodes.
- A DSN is dynamic because of changes in both its topology and its membership. Trust relationships among nodes may also change. Our protocol should adapt on-the-fly to these changes.

2. PUBLIC KEY INFRASTRUCTURE (PKI) FOR AD HOC NETWORKS.

Design. In this paragraph we propose a distributed and hierarchical PKI. The basic principle behind our design is *distribution of trust* and *robustness*. On the top layer of the hierarchy we have a master certification SK that is used to issue certificates for the public keys of the nodes on level 1. Next to this, all nodes

layer	certification keys	certificates		
level 0	$SK_0 = (s_1^0, \ldots, s_n^0)$			
level 1	$SK_1 = (s_1^1, \ldots, s_n^1)$	$\mathsf{Cert}_0(PK_{1,1})$	$\ldots$	$\mathsf{Cert}_0(PK_{1,n})$
level 2	$SK_2 = (s_1^2, \ldots, s_n^2)$	$\mathsf{Cert}_1(PK_{2,1})$	$\ldots$	$\mathsf{Cert}_1(PK_{2,n})$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
level m	—	$\mathsf{Cert}_{m-1}(PK_{m,1})$	$\ldots$	$\mathsf{Cert}_{m-1}(PK_{m,n})$

Figure 1: Distributed and hierarchical PKI for ad hoc networks. Shares of the private key SK_l at layer l are signed by the private key SK_{l-1} of layer $l-1$.

on level 1 get a share of the layer 1 certification key SK_1. Similar, level 2 nodes receive a certificate signed by SK_1 and a share of the layer 2 certification key SK_2. This process is continued until the desired number of levels in the hierarchy is reached (see figure 1).

Distribution of trust is achieved using threshold cryptography [4, 5]. We use an $(n, t+1)$ threshold scheme that allows n parties to share the ability to create a digital signature, so that any $t+1$ parties can perform this operation jointly, whereas it is infeasible for at most t parties to do so. In our case, the certification keys SK_i are divided into n shares $(s_1^i, s_2^i, \ldots, s_n^i)$. If a node at some level requires a certificate, it will contact $t+1$ nodes of the previous level (up) to gather $t+1$ partial signatures and combine them to compute the signature for the certificate.

Robustness is an inherent property of an $(n, t+1)$ threshold scheme because only $t+1$ working and honest nodes are necessary to sign a certificate. Some threshold schemes exploit redundancies in the partial signatures and use error correcting codes to mask incorrect partial signatures [8]. With these schemes a correct signature is obtained despite a small number of partial signatures being incorrect – this means that our scheme will recover from corrupted nodes that return incorrect partial signatures. Once a corrupted node has been detected, it can be removed from the system using proactive threshold schemes that enable share refreshing [7, 9] and reconfiguration from an $(n, t+1)$ to an $(n', t'+1)$ scheme [6]. These schemes allow nodes to compute new shares from old ones in collaboration[1] without disclosing the certification key. Because these new shares are independent of old ones, an adversary cannot combine old shares with new shares to recover the certification key or generate a signature. Reconfiguration can be used to adapt the system to changes in the network topology. For example,

[1] Again at least $t+1$ honest nodes are necessary to generate fresh shares.

the system may start with a $(9, 4)$ configuration. After detection of a corrupted node and failure of another, the shares are refreshed and reconfigured to a $(7, 3)$ configuration. Because of the hierarchical structure it is possible to issue new certificates: higher level nodes can collaborate to securely establish a new shared certification key SK'_i, and issue new certificates for all nodes on a lower level. Note that this is possible to do this without revealing the actual secret key to any of the collaborating nodes [2]. Finally, public verifiable secret sharing schemes [14, 15] can be used to detect incorrect shares, for example during the share refreshing.

Pre-deployment PKI Setup. As mentioned in the introduction, we suppose that the targeted ad hoc network will have a functional hierarchy of arbitrary level (our example sensor network has two functional levels: control and sensor nodes). In order to setup the proposed PKI, nodes of different levels will receive (pre-deployment) different certificates and shares of certification keys:

- Nodes on the bottom of the hierarchical structure[2] receive no shares (because there is no lower level to sign certificates for).
- All other nodes receive a share of the certification key that corresponds with their hierarchical level (level 1 nodes receive layer 1 shares, etc.).
- All nodes receive a "bottom layer" certificate. This certificate is needed to be able to setup keys that will be used to send data (data is always transmitted as if all nodes belong to the bottom layer).
- Higher level nodes are allowed to request lower layer shares and certificates.
- All nodes have a copy of all the certification *public* keys.

If (like in our sensor network example) only two levels are present, we suggest to create three virtual levels: the sensor nodes are level 2 (bottom layer) nodes, while the control nodes function both as level 1 and level 0 nodes simultaneously. This makes the whole PKI much more flexible compared to using only 2 levels.

Dynamic Behavior and Collaboration with Misbehavior Detection. After the initial deployment of the network, nodes can move in and out the network, nodes can be destroyed, become corrupted, request a new certificate, etc. Our PKI should be able to cope with these issues. New nodes will simply have a valid certificate installed pre-deployment. We already explained how the system

[2]Sensor nodes in our example.

can adopt itself if nodes are removed from the system by reconfiguration of the threshold scheme.

Our protocol can also collaborate with a misbehavior detection system [11]. Again we use the hierarchical structure of the network and of our PKI. If nodes at some level detect misbehavior of other nodes, than they will report this to the nodes of the previous level (up) with a signed complaint. If $t + 1$ nodes have received sufficient complaints then they can collaborate and sign and distribute an eviction notice. The list of eviction notices can then be used to exclude nodes from the network or treat them with more caution. Once the list exceeds a certain limit, fresh certificates can be signed for the remaining trusted nodes.

3. LINK KEY ESTABLISHMENT

Once all certificates and shares are in place all nodes will establish symmetric link keys to authenticate and possibly encrypt all data traffic. Because of space limitations, we will only sketch the basic outline.

- Link keys are established using a flooding system. The flooding is initiated by nodes on the top of the hierarchical structure. These nodes broadcast a signed wake-up call to their surrounding nodes. All nodes receiving this call will initiate an authenticated key establishment protocol, using their certificates to prove their identity [12, ch. 12]. The certificates used to setup link keys are always "bottom layer" certificates. Using "bottom layer" enables higher level nodes to hide themselves in the network, this makes it much harder for adversaries to locate them. Using the same type of certificates throughout also makes implementation easier.
- In the next step all already connected nodes broadcast their own signed wake-up call to their neighbors, and so on. The wake-up calls contain information on links the nodes already established. The result of this is that all nodes now share a symmetric key with all their neighbors, and as an important side-effect also have reliable routing information because of the information contained in the wake-up calls.

Strong authentication of wake-up calls is very important to effectively counter the battery drain attacks. If, for example, there is no replay detection, an adversary could reuse the same wake-up call over and over again until he drained the battery of all receiving nodes. What's worse, these nodes will answer this wake-up call and broadcast their own wake-up call, possibly reaching the entire

network. Replay prevention is usually based on some use of *state information*, like counters, challenges or time stamps. We propose a scheme to achieve replay detection of virtually all messages in an extended version of this paper.

Dynamic Behavior. After nodes leave or move in the network, certain links will no longer be available. To reestablish broken chains, we allow that nodes broadcast a reconnect call. Neighboring nodes are free to discard or answer this reconnect call and establish a link key. Periodically, the top level nodes can rebroadcast a signed wake-up call and refresh all link keys.

5. CONCLUSIONS

In this paper, we have analyzed the security threats ad hoc networks face and have extracted the resulting security goals that should be met. We have presented a complete key management scheme for ad hoc networks. Our scheme exists of two parts: a hierarchical distributed public key infrastructure and a protocol to securely establish link keys. Because of the distribution of trust and secret information throughout the network, we avoid a single point of failure or a single point of attack (like a central certification authority). Our PKI is based on threshold cryptography to achieve robustness, flexibility and a high level of security. Finally, our scheme can also be used on other types of networks as long as there are a sufficient number of nodes to distribute trust.

ACKNOWLEDGMENTS

This work was supported by the Concerted Research Action (GOA) Mefisto-2000/06 of the Flemish Government.
Stefaan Seys is funded by a research grant of the Flemish Institute for the Promotion of Industrial Scientific and Technological Research (IWT).
Dr. Bart Preneel is professor at the Katholieke Universiteit Leuven, Belgium.

REFERENCES

[1] F. Bennett, D. Clarke, J.B. Evans, A. Hopper, A. Jones, and D. Leask. Piconet: embedded mobile networking. *IEEE Personal Communications*, 4(5):8–15, October 1997.

[2] D. Boneh and M. Franklin. Efficient generation of shared rsa keys. *Journal of the ACM (JACM)*, 48(4):702–722, July 2001.

[3] DARPA. DARPA/ITO SensIT project. Online: `http://www.darpa.mil/ito/research/sensit/index.html`.

[4] Y. Desmedt. Threshold cryptography. *European Transactions on Telecommunications*, 5(4):449–457, July 1994.

[5] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 307–315, Santa Barbara, California, U.S.A., August 1989. Springer-Verlag.

[6] Y. Desmedt and S. Jajodia. Redistributing secret shares to new access structures and its applications. Technical Report ISSE_TR-97-01, George Mason University, July 1997.

[7] Y. Frankel, P. Gemmel, P. MacKenzie, and M. Yung. Optimal resilience proactive public-key cryptosystems. In *Proceedings of the 38th Symposium on Foundations of Computer Science*, 1997.

[8] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In U. Maurer, editor, *Advances in Cryptology – Proceedings of Eurocrypt '96*, number 1070 in Lecture Notes in Computer Science, pages 354–371, Zaragoza, Spain, May 1996. Springer-Verlag.

[9] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive secret sharing or: how to cope with perpetual leakage. In D. Coppersmith, editor, *Advances in Cryptology – CRYPTO '95*, number 963 in Lecture Notes in Computer Science, pages 457–469, Santa Barbara, California, U.S.A., August 1995. Springer-Verlag.

[10] J.M. Kahn, R.H. Katz, and K.S.J. Pister. Next century challenges: Mobile networking for "Smart Dust". In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom '99)*. ACM Press, August 1999.

[11] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in ad hoc networks. In *Proceedings of Mobile Computing and Networking (MOBICOM '00)*, pages 255–265, 2000.

[12] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[13] Picoradio. Picoradio, the berkeley wireless research center picoradio project. Online: `http://bwrc.eecs.berkeley.edu/Research/Pico_Radio/`.

[14] Berry Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic voting. In *Advances in Cryptology – CRYPTO'99*, number 1666 in Lecture Notes in Computer Science, pages 148–164, Berlin, 1999. Springer-Verlag.

[15] M. Stadler. Publicly verifiable secret sharing. In *Advances in Cryptology – Eurocrypt '96*, number 1070 in Lecture Notes in Computer Science, pages 190–199. Springer-Verlag, 1996.

[16] Frank Stajano and Ross Anderson. The resurrecting duckling: Security issues in ad-hoc wireless networks. In B. Christianson, B. Crispo, and M. Roe, editors, *Proceedings of the 7th International Workshop on Security Protocols*, Lecture Notes in Computer Science. Springer-Verlag, 1999.