International Journal of Distributed Sensor Networks, 1: 3–15, 2005

ISSN: 1550-1329 print/1550-1477 online
DOI: 10.1080/15501320490522803

Taylor & Francis
Taylor & Francis Group

New Directions in Sensor Network Key Management

DAVID W. CARMAN

McAfee Research, McAfee Inc., Rockville, MD 20850, USA

Sensor networks require efficient, low latency key management techniques that enable strong security and tolerance of node compromise. Conventional interactive approaches using public key certificate-based key management techniques are not communications efficient and are very time-consuming. Protocols that leverage elliptic curve cryptography reduce communications but still require considerable interactive exchange. Noninteractive techniques that leverage identity-based public-key cryptography show considerable promise, but these techniques are relatively immature and require considerable computations. Conversely, random key predistribution techniques reduce computations, but at the expense of many interactions. In this paper, we describe recent work in the cryptographic community that combines the benefits of both identity-based cryptography and random-key predistribution into a framework we call identity-based random-key predistribution (IBRKP). IBRKP establishes pair-wise keys with virtually no extra communications and provides security versus node memory trade-offs for the sensor network designer to engineer.

Keywords Wireless sensor networks; key predistribution; security; key management

1. Introduction

Do we need to worry about the security of our sensor networks? Because sensor networks are usually designed to produce something of value—information—the answer is often "Yes." Although privacy is a commonly mentioned goal [1], authenticity and availability of sensor network data are needed as well [2]. Cryptographic security services such as encryption and message authentication directly help achieve these security goals. Reliable security mechanisms, such as the Advanced Encryption Standard (AES) [3] and hashed-based message authentication codes (HMACs), perform adequately in most distributed sensor network environments. However, the security of these mechanisms is dependent on key management, which can be particularly challenging in the distributed sensor network environment.

Traditional public-key infrastructure solutions are unsuitable for managing keys in distributed sensor networks. Sensor network devices, hereinafter called *nodes*, are severely resource constrained, lacking sufficient computational and communications capability to establish keys in an energy-efficient and timely manner. Public-key- and

Disclaimer: Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation therein. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

Address correspondence to David W. Carman, McAfee Research, McAfee Inc., 15204 Omega Drive, Suite 300, Rockville, MD 20850, USA, E-mail: David_Carman@McAfee.com

secret-key-based alternatives offer approaches that better preserve these precious resources.

The rest of this paper is organized as follows. Section 2 discusses the distributed sensor network environment and conventional key management solutions. In Sec. 3, we discuss the advantages of using elliptic curve cryptography. Section 4 introduces the advantages of identity-based cryptography. Section 5 reviews random-key predistribution techniques that have been recently proposed. In Sec. 6, we describe an identity-based random-key predistribution technique, including its security and implementation properties. Finally, Sec. 7 and 8 provide conclusions and references, respectively.

2. Background

Sensor networks are inherently susceptible to attack, because their constituent nodes are often located in areas out of the control of the sensor network owner [4]. Electronic attacks by an adversary, such as eavesdropping and forgery, can be mitigated using security mechanisms that employ cryptographic keys [5]. Sensor networks can efficiently achieve privacy protection by using encryption algorithms such as the Advanced Encryption Standard (AES). Similarly, forgery can be efficiently countered through the use of message authentication algorithms, such as HMAC-SHA-1 [6], or specialized modes of AES, such as AES-XCBC-MAC [7]. These existing security mechanisms are generally regarded as secure and usually consume less energy and latency than the corresponding communications components that transmit and receive the protected sensor data [8].

Although many electronic attacks can be mitigated using existing security mechanisms, sensor nodes may be subject to physical attack and compromise by an adversary. Of greatest concern is the malicious recovery of a sensor node's cryptographic keys, because their privacy is crucial to the security of the encryption and message authentication mechanisms. Key compromise can potentially negate an entire sensor network's communications security.

A high-level strategy for mitigating key compromise is to limit the amount of data that a given node's key protects, thus limiting the data revealed or forged by that key's compromise. Minimizing the data protected by a key can be accomplished by changing keys often, using different keys between communicating pairs, or using different keys to protect different network layers (e.g., link, network, and application layers). Mitigating node compromise through the management of key creation, use, rekeying, and revocation, while addressing the unique challenges of the sensor network environment, remains a core security problem for distributed sensor networks.

To achieve higher granularity in cryptographic key usage, an "active" key management approach must be pursued. An active approach means that separate keys are established to protect unique security associations for each communicating pair, each group, each time period (e.g., 8 hours), and each layer (e.g., link, network, application). Active key management helps protect against undetected node compromises by automatically discontinuing use of the compromised key. Static keying approaches, such as predeploying a network-wide key, can result in a single-node compromise, causing widespread security failure.

The more prevalent limitation to providing active-key management in the distributed sensor network environment is limited bandwidth. Conventional active-key management approaches require interactive communications, but distributed sensor networks must operate over constrained and noisy wireless channels with near-earth propagation effects and intermittent connectivity. A surprisingly low effective channel rate results from a combination of lower-rate wireless links, a media access control (MAC) layer shared

among multiple nearby transmitting nodes, and a communications network shared among multiple types of traffic (e.g., routing, data, command, and control). With such low-rate links, distributed sensor network nodes will encounter extremely large network latencies when trying to invoke secure network services over multiple hops. Thus, the most important compromise mitigation challenge is to actively establish, maintain, and revoke keys, while minimizing delay and consumed bandwidth.

Traditional public-key infrastructure solutions are unsuitable for establishing link and network-layer keys in distributed sensor networks. Interactive certificate-based protocols such as Internet key exchange (IKE) [9] and high-assurance IP encryption (HAIPE) provide many desirable key management features, including algorithm flexibility, various authentication technologies, and forward secrecy—an important tool for mitigating node compromise. *Forward secrecy* means that compromise of a long-term private key does not reveal past short-term session keys.

The major downside of interactive key management protocols is that they consume considerable energy and communications bandwidth in distributed sensor networks, while requiring a large amount of time to establish an initial security association [10]. Hill et al. similarly reported the relative expensiveness of communications versus computations, in terms of energy consumption and latency, noting that roughly 100 instructions consume as much energy and execute in the same amount of time as communicating a single bit in a representative sensor network node [11].

3. Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is an alternative public-key algorithm that can be used to construct key management and digital signature protocols. The security of ECC is based on the elliptic curve discrete logarithm problem, which the cryptographic community regards as much more difficult than the integer factorization and discrete logarithm problems that underlie the conventional Rivest-Shamir-Adelman (RSA) and Diffie–Hellman public-key algorithms.

Due to the greater per-bit strength of its underlying security, ECC improves the performance of key management protocols in distributed sensor networks. ECC has two main advantages: (a) ECC public keys are smaller for the same level of security as RSA or Diffic–Hellman-based solutions, thus reducing the number of bits that need to be exchanged; and (b) ECC public-key operations require fewer computations than conventional public-key methods. The National Institutes of Standards and Technology (NIST) drafted key management guidelines [12] that estimate the ECC public-key sizes needed to provide equivalent security to various symmetric and RSA/Diffie–Hellman public-key sizes. For 112-bit security equivalent to the three-key Triple Data Encryption Standard (DES) symmetric encryption algorithm, 2048-bit RSA or Diffie–Hellman is needed, but only 224-bit ECC public-key size is necessary. For the higher security levels needed in the future, the key size advantage of ECC over conventional public-key algorithms becomes more dramatic. When used in a sensor network key establishment protocol, ECC can reduce the overall number of communicated bits by more than half and reduce computation times by over an order of magnitude [13].

Despite the communications and computational reductions offered by ECC technology, interactive ECC-based key establishment protocols, such as IKE, do not reduce the number of expensive multihop interactions. Interactive key management protocols, even when using faster and smaller public-key algorithm alternatives, incur significant network bandwidth and large latencies when employed in distributed sensor networks.

4. Identity-Based Cryptography

Key management protocols that utilize identity-based cryptography offer significant advantages in the distributed sensor network environment over conventional key management protocols. Identity-based cryptography was first proposed by Shamir [14] and allows one to derive a user's public key from his or her identity. Public-key operations can then be performed using the derived public key with the assumption that only the identified user possesses the corresponding private key. Identity-based cryptography obviates the need for public-key certificates, because the binding between a public key and an identity is inherent to the algorithm.

Identity-based cryptography schemes employ a key authority that has the ability to compute a user's public–private key pair from its identity. The key authority derives this capability from a trapdoor function that only it can perform. The key authority distributes private keys to users prior to key establishment but needs not play any role thereafter.

Although revocation can take place in identity-based systems in a fashion similar to certificate-based systems, the need for revocation can be virtually eliminated through the use of short-term (ephemeral) public–private key pairs. By using a public–private key pair valid for only a day or week, the need for revocation is dramatically reduced versus traditional public-key certificate systems. Many more key pairs must be distributed in an ephemeral identity-based cryptosystem than in a traditional public-key system, but this distribution can take place at times convenient to the key authority and sensor node.

Several candidate identity-based cryptographic algorithms have been developed in the past few years that can be used for key management. When leveraging these algorithms for establishing pair-wise keys, they fall into two types: unidirectional and bidirectional. In this context, unidirectional means that the public key derived from the user identity can be used to wrap a secret key, and the wrapped secret key is sent to the user where it can be unwrapped (decrypted) using the corresponding private key. Sometimes called identity-based encryption, unidirectional schemes allow one to use the wrapped secret key to encrypt a message and then send the wrapped key and encrypted message together to the destination. Boneh and Franklin constructed an identity-based encryption (IBE) scheme that uses the Weil pairing and the elliptic curve discrete logarithm problem as its underlying security [15].

Bidirectional schemes allow both the sender and recipient of a message to compute a pair-wise key without exchanging a wrapped key. In these schemes, the sender derives the recipient's public key and then computes the pair-wise key by performing a Diffie–Hellman-like operation using the sender's private key and recipient's public key. The sender uses the computed pair-wise key to encrypt and authenticate messages it sends to the recipient. The recipient needs only the sender's identity (presumably in a plaintext message header) to conversely compute the pair-wise key and to decrypt and verify the message. Maurer and Yacobi described a bidirectional scheme [16] that has been examined for use in a U.S. Army sensor network [10]. With Matt and Cirincione [10] we extended the use of identity-based group keying with the use of one-time identity-based keys to create a protocol that meets the security requirements of Army sensor networks, while significantly reducing energy and latency costs versus the existing certificate-based approaches by up to an order of magnitude.

One advantage of bidirectional schemes over unidirectional schemes is that the wrapped key need not be sent to the recipient to decrypt and verify the message. This is particularly beneficial in sensor networks, where data may be divided into many small packets, and the addition of a wrapped key to each packet would be a significant penalty. Unidirectional-based systems could mitigate this issue by engaging in a protocol whereby

the recipient acknowledges receipt of the wrapped key, but such protocols require additional messaging and state maintenance that may be undesirable.

One impediment to using identity-based cryptography in sensor networks is the significant number of computations required to perform identity-based operations. Although deriving a node's public key is usually not too taxing, the associated operations using that public key are at least as computationally intensive as traditional public-key operations. For instance, the computational performance of Boneh and Franklin's IBE is comparable to that of ElGamal encryption in F_p^* [15], whereas the computational performance of Maurer–Yacobi is worse than either RSA or Diffie–Hellman for the same size modulus. Identity-based cryptographic operations may be appropriate on sensors with capable general-purpose processors or dedicated acceleration hardware, but they are unsuitable for low-cost sensors that contain limited processing capabilities.

5. Random-Key Predistribution

Random-key predistribution is an alternate key management technique that significantly reduces sensor-key management computations. First proposed by Eschenauer and Gligor [17], this scheme relies on a key authority to randomly generate a pool of secret keys and predistribute a randomly selected subset of these keys, called a *key ring*, to cach sensor node in the network. In the key setup phase, sensor nodes broadcast key identifiers to their neighbors, allowing nodes to discover which, if any, secret keys are held in common. Sensor nodes that share at least one common secret pool key can establish a probabilistically secure link.

If the number of predistributed secret pool keys is small, two sensor nodes may not hold any pool keys in common. In this case, a security association can be established if a connected path of secure links can be formed. The probability that a security association can be established between two nodes is dependent on the pool size, key ring size, and the expected number of neighbors, or *degree*, of a node.

The security of created links is probabilistically determined from the pool size, key ring size, and number of compromised sensor nodes. The adversary's goal is to gain possession of the common keys shared by a targeted pair of nodes. To accomplish this, an adversary attempts to compromise other sensor nodes, thus gaining possession of their corresponding key rings. Although the probability an adversary will gain the common keys it targets from the compromise of a single sensor is small, the odds increase as it accumulates key rings from multiple sensor compromises. Sensor networks can mitigate this threat by making the key ring size small in relation to the pool size so that each sensor node compromise only endangers a small portion of the overall key pool. Conversely, the key ring size should be large so that targeted pairs have many common keys, forcing the adversary to obtain every one to compromise the link. Chan, Perrig, and Song offer a similar mitigation strategy by requiring that a minimum number of keys be shared when establishing secure links [18].

The major advantage of random-key predistribution for sensor network key management is that computationally expensive public-key operations need not be performed. Key setup is not expensive, because sensor nodes discover common keys by simply comparing a received key identifier to their internally held list. Similarly, hashing the set of common keys to derive the pair-wise key is a relatively trivial operation compared to public-key algorithms. Another advantage is that the security of the scheme is probabilistically determinable unlike public-key-based schemes, where security is based on a mathematical problem perceived to be computationally difficult.

The major disadvantage of this approach is the large amount of communications required during key setup. Sensor nodes must broadcast their list of key identifiers.

Because we wish this list to be large for security reasons, many large messages must be exchanged. This stage of random-key predistribution incurs significant time to establish the security association and consumes communications bandwidth and energy. A second disadvantage is the general notion that a given link's security can be broken through the compromise of other nodes. Unlike public-key cryptography, where node compromise only reveals the security associations in which that node participates, node compromise in networks that use random-key predistribution can result in an adversary gaining access to other security associations for which the compromised node is not a party.

6. Identity-Based Random-Key Predistribution

Two separate sets of authors [19, 20] applied the concept of identity-based keying to the random-key predistribution schemes of Eschenauer and Gligor and Chan, Perrig, and Song to create IBRKP schemes. Whereas previous random-key predistribution schemes focused on reducing sensor computations at the expense of communications, our main focus is on reducing communications.

IBRKP is different from random-key predistribution in two important ways. First, the IBRKP key authority predistributes keys from the key pool to each node based on a pseudo-random function of the node identity, rather than just randomly. Second, only node identities are exchanged during the IBRKP key setup phase, not individual key identifiers, as in random-key predistribution. IBRKP eliminates the need to exchange key identifiers by deriving them from the exchanged node identifiers. The remainder of this section details the four stages of IBRKP as outlined in Fig. 1.

IBRKP's first stage is establishment of a key authority and initialization of the system key pool. The key authority is an entity that must securely generate and store cryptographic keys. It establishes a key pool size, S, based on desired connectivity and security characteristics and generates the required number of keys. Each key is a randomly generated value and is of sufficient size for the desired security of the system. The key authority

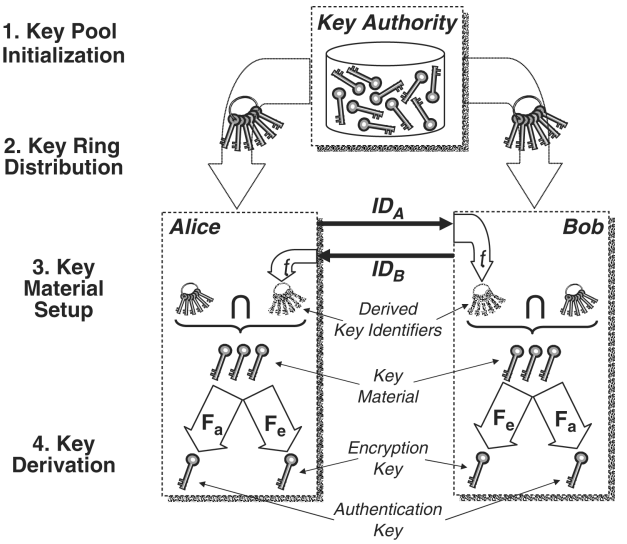

FIGURE 1 Identity-based random-key predistribution stages.

also establishes the size of the distributed key rings, *m*, and the functions that derive the key identifiers, encryption keys, and authentication keys.

In the second stage, the key authority derives and distributes key rings to the sensor nodes. Key rings are created for each node by performing a function, *f*, on the node's identifier to generate the *m* key identifiers. A system-wide-shared secret value may also be combined with the node identifier to prevent an adversary from computing a node's key identifier list without first compromising the system-wide-shared secret value. The benefit of such a strategy is that if an adversary does not know the node key identifier lists of two communicating nodes, it will not know which keys are needed to compromise a link a priori, and thus, which nodes to compromise to obtain these keys.

Any pseudorandom function that produces unique and uniformly distributed outputs can be used for the function *f*. For instance, AES can be used for this function by hashing the node identifier to create an AES key. AES is then executed in counter-mode [21], with the output being used to create the key identifiers. For sensor networks where the computation speed of *f* may be a design consideration, faster pseudorandom functions such as the Mersenne Twister [22] may be appropriate. Because as many unique key identifiers are needed as the key ring size, any duplicate generated values by the *f* function are discarded.

For each node, the key authority derives the key identifier list using the pseudorandom function and retrieves each corresponding key from its key pool to construct the sensor node's key ring. The key ring containing the *m* keys is transferred from the key authority to the sensor node via a traditional communications security (COMSEC) key management system [e.g., Electronic Key Management Systems (EKMS)].

The third stage occurs whenever a sensor node wishes to securely communicate with another sensor node in the deployed sensor field. We assume all pair-wise communications take place between identified sensor nodes, that is, no anonymous pair-wise communications occur within the distributed sensor network. This assumption also implies that sensor nodes learn of destination node identities via routing messages or some other practical means. Note, although node identities must be obtained before two nodes communicate, identities do not have to be learned prior to deployment. Retrieval and use of sensor node identities can be "ad hoc" in these networks such that new sensor nodes can be readily identified and communicated with when encountered in the deployed sensor field.

To derive the protection keys for a pair-wise link, the sending node first derives the set of *m* key identifiers corresponding to a destination node using the same function *f* that was established and used in the first two IBRKP stages. The sending node then determines the *i* common keys it shares with the destination node. The *i* common keys are then concatenated in a canonical order to form the unique pair-wise key material for the sender–destination pair.

The probability a pair of nodes share exactly *i* keys when each possesses *m* keys is given by Chan et al.[18]. However, in a heterogeneous network where nodes may have varying resistance to physical compromise and key storage limitations, there may be considerable advantages to varying the key ring size. For instance, a manned sensor gateway node may be less vulnerable to physical compromise and contain much greater memory storage than a low-cost disposable sensor node dropped behind enemy lines.

To compute the probability that two heterogeneous nodes share *i* keys, we first note that the key authority distributes key rings of size m_a and m_b to two nodes we designate Alice and Bob, respectively. The pseudorandom function *f* used by the key authority has $\binom{S}{m_a}$ ways of selecting Alice's keys and $\binom{S}{m_b}$ ways of selecting Bob's keys, for a total

of $\binom{S}{m_a}\binom{S}{m_b}$. There are $\binom{S}{i}$ ways to select the i keys Alice and Bob have in common. Alice has $\binom{S-i}{m_a-i}$ ways of selecting her remaining keys, whereas Bob has $\binom{S-m_a}{m_b-i}$ ways of selecting his remaining keys. Hence, the probability that Alice and Bob share exactly i keys is as follows:

$$p(i) = \frac{\binom{S}{i}\binom{S-i}{m_a-i}\binom{S-m_a}{m_b-i}}{\binom{S}{m_a}\binom{S}{m_b}}$$

To provide context for the above equation, we posit an IBRKP system with pool size S = 2,097,152 (= 2^{21}), key ring size $m_a = m_b$ = 16,384 (= 2^{14}), and the average number of common keys i = 128 (= 2^7). For these parameters, the probability that two nodes do not share any keys is exceedingly rare, less than 10^{-50}. Even the chance that two nodes share less than half the average number of common keys, 64, is about 10^{-10}.

The fourth and final stage uses the common pair-wise key material and potentially some common nonce information to derive the encryption and message authentication keys used to provide link and network layer messages. Conventional key derivation methods such as those described in PKCS #5 [23] and IKE [9] may be applied to securely derive these protection keys.

6.1 IBRKP Security

The IBRKP system contains two main avenues for attack: compromise of the key authority, and compromise of sensor nodes.

1. *Compromise of key authority*: Because it contains all the keys used to derive keying material, a compromise of the key authority results in a total loss of security for all derived encryption and authentication keys used throughout the network. A traditional approach would locate it in a physically secure facility far away from potential enemies. However, there may be scenarios where a distributed sensor network's autonomy and isolation dictate that it possess its own key authority. With compromise of the key authority being a greater concern when it can be physically compromised, a simple mitigation strategy is to distribute this functionality among multiple key authority shareholder nodes. In addition to distributing the storage of the key shares, the key generation function can also be distributed for added protection. Traditional threshold schemes such as Shamir's secret sharing [24] may be applied to allow distribution while also preserving the availability of the key authority in the event of damage or destruction of one or more of the key authority shareholders.
2. *Compromise of developed sensor nodes*: Loss of security due to the physical compromise of deployed sensor nodes is also of concern. The probability that an adversary that obtains a single node's key ring can break the protection keys used by a specific sender–destination pair is $\left(\frac{m}{S}\right)^i$. To add some context to this equation,

we assume that in the distributed sensor environment, we can support a pool size S = 2,097,152 (= 2^{21}), key ring size m = 16,384 (= 2^{14}), and the average number of common keys i = 128 (= 2^7). Therefore, the probability of a single-node compromise breaking a given pair-wise link is exceedingly small: 2^{-896}. Clearly, a single-node compromise is little to worry about, unless it is of the node you wish to protect.

More powerful attacks are possible by an adversary against an IBRKP-protected sensor network by compromising multiple nodes and aggregating the obtained keys. The probability of a given pair-wise link being compromised by x randomly selected nodes is the same as for random-key predistribution schemes and was documented by Chan et al. [18] as $\left(1-\left(1-\frac{m}{S}\right)^x\right)^i$. For the IBRKP system parameters we propose above, we show in Fig. 2 the security of a pair-wise link consisting of an average number of common keys for various numbers of sensor nodes compromised, as well as the security for the rare case (1 in 10 billion) of a pair-wise link consisting of only half the average number of common keys.

The security shown in Fig. 2 may be sufficient for many applications, but the per-node key ring storage required, 524,288 bytes, may be too large for some sensor node platforms. Using the same key pool size of S = 2,097,152, we examine the security provided by smaller key ring sizes in Fig. 3.

6.2 IBRKP Implementation

We implemented IBRKP on an IBM Thinkpad Pentium 41.6 GHz laptop running Windows XP. The implemented code performs the three main node functions: (a) determines the key identifiers from the other node's identity, (b) finds the common set of key identifiers, and (c) hashes the common set of keys in a canonical order and derives the encryption and message authentication keys. The total timing of the three functions using various key ring sizes is shown in Table 1.

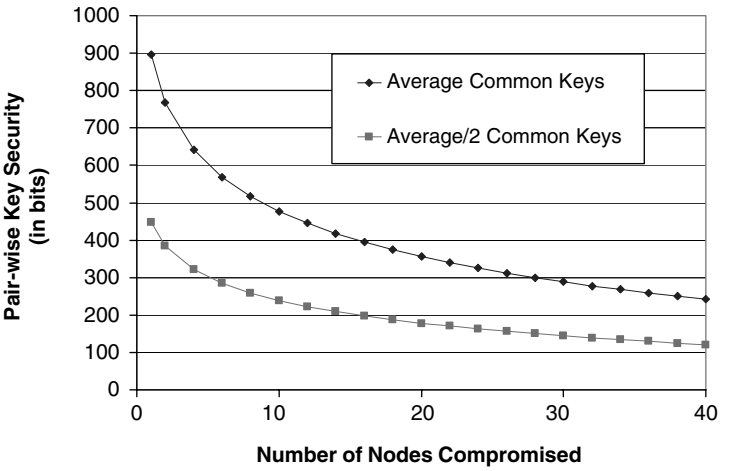

FIGURE 2 IBRKP security versus nodes compromised.

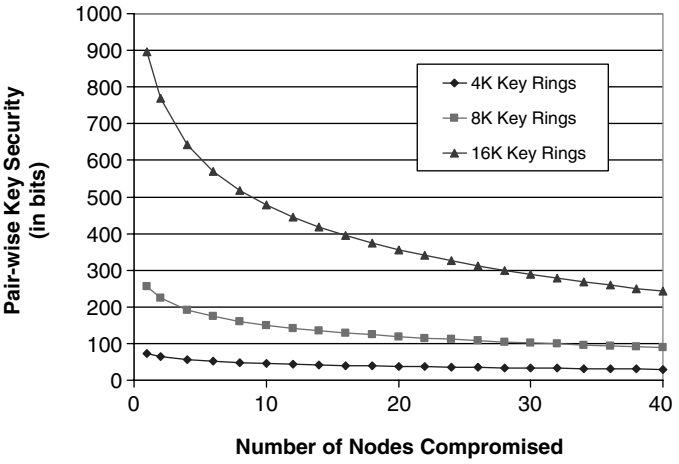

FIGURE 3 IBRKP security for different key ring sizes.

TABLE 1 IBRKP Computation Time for Various Key Identifier Sizes

Key ring size	Total time (milliseconds)
4096	1.4
8192	3.0
16384	6.2
32768	12.8

As a comparison, the Maurer–Yacobi identity-based public-key scheme with a 2048-bit modular exponentiation and 2048-bit exponent takes approximately 440 milliseconds to complete on the same platform. Although a Pentium-based laptop processor is much faster than virtually any current sensor processor, the performance advantage of IBRKP versus modular exponentiation is striking independent of processor speed.

For the 16,384 key ring size case, the timing breakdown of the individual functions is shown in Table 2. We derive the key identifiers by using AES in counter-mode with the node identifier as the key. Surprisingly, this conceptually simple process of deriving pseudorandom key identifiers consumes the majority of the overall computation time. Because we do not need a function with the cryptographic strength of AES to generate key identifiers, systems that wish to reduce computation time can chose faster pseudorandom functions such as the Mersenne Twister.

TABLE 2 IBRKP Computation Time per Function

Function	Time (milliseconds)	Percentage of Total Time
Derive key IDs	3.55	57%
Find common set of key IDs	2.45	40%
Hash common set	0.2	3%

TABLE 3 Table Lookups per Key Identifier for Various Key Ring Sizes

Key ring size (number of keys)	Table lookups per key identifier
4,096	3.67
8,192	5.23
16,384	6.16
32,768	6.31

To find the common set of key IDs, we process each derived key identifier one by one and search our list of own-key identifiers to determine if a match exists. Instead of performing a traditional binary search, we leverage the fact that our key identifiers should be uniformly distributed, and so, when computing the location in our own list, we would expect to find a match for the derived key identifier if the list was perfectly uniformly distributed. We perform a linear search if the first lookup is not a match. The average number of lookups into the table to determine a match or nonmatch per key identifier using our method and for various key identifier sizes per node is shown in Table 3.

We note that this method scales well for larger key ring sizes that might be considered for platforms with sufficient key storage.

We use the fairly plodding PBKDF2 function specified in PKCS #5 [23] to derive the encryption and message authentication keys from the common set of keys. Despite the relative slowness of the function, it consumes only a short amount of the overall key computation time.

6.3 Flexibility

A major advantage of the IBRKP approach is the flexibility it provides sensor network designers. A given IBRKP implementation can be tailored for its network security requirements, computational capabilities, and key storage constraints. By eliminating the energy necessary for communicating key management information and reducing computational energy consumption, the IBRKP is sensitive to the energy constraints of many sensor applications. The key storage required for IBRKP is relatively large, especially for resistance to large numbers of compromises, but this storage is infrequently accessed and thus need not consume much energy. Moreover, like Moore's law demonstrates for computational capability, storage capacity continues to increase while becoming more inexpensive. For instance, Infineon now offers a 512-Mbit flash memory chip for $8.50, while for even larger storage needs, Toshiba recently introduced a 0.85-inch-diameter 4 gigabyte hard disk drive weighing under 10 grams and about the size of a postage stamp. How much more will we store in 2024?

7. Conclusions

We described how sensor networks require efficient, low latency key management techniques that enable strong security and tolerance of node compromise. However, conventional interactive approaches using public-key certificate-based key management techniques are not communications efficient and are very time-consuming. Protocols that leverage elliptic curve cryptography reduce communications but still require considerable

interactive exchange. Noninteractive techniques that leverage identity-based public-key cryptography show considerable promise, but these techniques are relatively immature and require considerable computations. Conversely, random-key predistribution techniques reduce computations, but at the expense of many interactions. Recent work in the cryptographic community combines the benefits of identity-based cryptography and random-key predistribution into a framework called identity-based random-key predistribution (IBRKP). IBRKP establishes pair-wise keys with virtually no extra communications, making it an attractive alternative in sensor networks, where communications are relatively expensive. We provided early implementation results and security versus node memory trade-offs showing that this approach can be an attractive alternative for many distributed sensor network applications.

Acknowledgment

The author wishes to thank Greg Cirincione's review of an earlier draft of this article.

References

1. D. Culler, D. Estrin and M. Srivastava, "Overview of sensor networks," *Computer, IEEE Computer Society*, (August) 41–49 (2004).
2. K. Martinez, J. Hart, and R. Ong, "Sensor network applications," *Computer, IEEE Computer Society*, (August), 50–56 (2004).
3. Federal Information Processing Standard Publication 197, Advanced Encryption Standard (AES). National Institute of Standards and Technology (NIST). November (2001).
4. D. Balenson, D. Carman, P. Dinsmore, and P. Kruus, "Security requirements and constraints for Army sensor networks," NAI Labs Technical Report #00-015, May 15 (2000).
5. D. Balenson, D. Carman, P. Dinsmore, and P. Kruus, "Communications security architecture for Army sensor networks," NAI Labs Technical Report #00-016, September 30 (2000).
6. C. Madson, and R. Glenn, "The use of HMAC-SHA-1-96 within ESP and AH," RFC 2404, IETF, November (1998).
7. S. Frankel, and H. Herbert, "The AES-XCBC-MAC-96 algorithm and its use with IPsec," RFC 3566, IETF, September (2003).
8. D. Carman, P. Kruus, and B. Matt, "Constraints and approaches for distributed sensor network security", NAI Labs Technical Report #00-010, June 1 (2000).
9. D. Harkins, and D. Carrel, "The Internet Key Exchange (IKE)," RFC 2409, IETF, November (1998).
10. D. Carman, B. Matt, and G. Cirincione, "Energy-efficient and low-latency key management for sensor networks," *Proceedings of the 23rd Army Science Conference*, Paper # OO-03, December (2002).
11. J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *Architectural Support for Programming Languages and Operating Systems* (ASPLOS-00) (ACM Press, 2000) pp. 93–104.
12. National Institute of Standards and Technology (NIST), "Key management guideline," http://csrc.nist.gov/CryptoToolkit/kms/key-management-guideline-(workshop).pdf, November (2001).
13. Q. Huang, J. Cukier, H. Kobayashi, B. Liu, and J. Zhang, "Fast authenticated key establishment protocols for self-organizing sensor networks," TR-2003-102, Mitsubishi Electric Research Laboratories, Inc. (2004).
14. A. Shamir, "Identity-based cryptosystems and signature schemes," *Advances in Cryptology—Crypto 84, LNCS 196* (Springer-Verlag, Heidelberg, 1984) pp. 47–53.
15. D. Boneh, and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM J. of Computing*, **32**(3), 586–615 (2003).
16. U. Maurer, and Y. Yacobi, "Non-interactive public-key cryptography," *Advances in Cryptology—EUROCRYPT 91, LNCS 547* (Springer-Verlag, Heidelberg, 1991) pp. 498–507.

17. L. Eschenauer, and V. Gligor, "A key-management scheme for distributed sensor networks," *Proceedings of the Ninth ACM Conference on Computer and Communication Security*, Ed. Vijay Atluri, November (ACM Press, 2002) pp. 41–47.
18. H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," *IEEE Symposium on Security and Privacy*, May (2003).
19. S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach," *Proceedings of the 11th IEEE International Conference on Network Protocols*, November (2003) p. 326.
20. R. Di Pietro, L. Mancini, and A. Mei, "Random key-assignment for secure wireless sensor networks." *Proceedings of the First ACM workshop on Security of ad hoc and sensor networks* (ACM Press, NY, NY, 2003) pp. 62–71.
21. M. Dworkin, SP 800-38A, "Recommendation for block cipher modes of operation," National Institute of Standards and Technology (NIST), Technology Administration, Department of Commerce, December (2001).
22. M. Matsumoto, and T. Nishimura, "Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator," *ACM Trans. on Modeling and Computer Simulation*, **8**(1), 3–30 (1998).
23. RSA Laboratories. "PKCS #5: Password-based encryption standard." Version 2.0, March (1999).
24. A. Shamir, "How to share a secret," *Communications of the ACM*, **22**(11), November (1979).
25. R. Min, and A. Chandrakasan, "Energy-efficient communications for ad-hoc wireless sensor networks," *Proceeding of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers* (2001) pp. 139–143.

David W. Carman is a Principal Research Scientist at McAfee Research and has been performing information assurance research and development since 1986. He has been the principal investigator for several DARPA, Army, and commercial network security projects. Mr. Carman is currently the Tactical Information Protection industry technical area lead for the Army Research Laboratory's Communication and Networking Collaborative Technology Alliance (CTA). He has performed research and development into low-latency and energy-efficient key management for U.S. Army sensor networks. Mr. Carman holds a B.S.E.E. from The Pennsylvania State University and an M.S.E.E. from The Johns Hopkins University.