Energy-Memory-Security Tradeoffs in Distributed Sensor Networks

David D. Hwang[1,2], Bo-Cheng Charles Lai[1], and Ingrid Verbauwhede[1,2]

[1] University of California—Los Angeles, Electrical Engineering Dept., Los Angeles CA 90095, USA
dhwang,bclai,ingrid@ee.ucla.edu
[2] Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC, Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium

Abstract. Security for sensor networks is challenging due to the resource-constrained nature of individual nodes, particularly their energy limitations. However, designing merely for energy savings may not result in a suitable security architecture. This paper investigates the inherent tradeoffs involved between energy, memory, and security robustness in distributed sensor networks. As a driver for the investigation, we introduce an energy-scalable key establishment protocol called cluster key grouping, which takes into account resource limitations in sensor nodes. We then define a metric (the security leakage factor) to quantify security robustness in a system. Finally, a framework called the security-memory-energy (SME) curve is presented that is used to evaluate and quantify the multi-metric tradeoffs involved in security design.

1 Introduction

Distributed sensor networks (DSNs) are a particular class of ad hoc networks that consist of (potentially) thousands of individual sensor nodes, each communicating via low-power wireless channels. The individual nodes are resource-constrained devices comprised of a microcontroller, memory, environmental sensors, and a radio transceiver. Potential application areas include military battlefield scenarios, habitat monitoring, building environmental control, and factory reliability sensing. Security plays an important role in sensor network architecture, as sensors may be deployed in enemy territory, contain private monitoring information, relay trade secrets, or possess other forms of sensitive data. Due their unique nature, DSN security is challenging in many ways: security must be scalable, be maintained without a powerful server, adapt to changes in network topology, and take into account the physical exposure of the nodes to an adversary.

However, perhaps the most challenging aspect of security in sensor nodes is coping with their severe resource constraints. Each individual node is limited in processing ability, memory, and energy. Energy in particular is the most valuable resource a node possesses; in most cases, unless energy-scavenging techniques are used [1], once energy is depleted in a node, the node is permanently offline. This is

I. Nikolaidis et al. (Eds.): ADHOC-NOW 2004, LNCS 3158, pp. 70–81, 2004.

why energy-based denial of service attacks [10] such as sleep deprivation torture [11] are particularly effective against DSNs. Hence, unlike security protocols for workstations (or even PDAs and notebook computers), energy expenditures for the security architecture of sensor networks must be kept to a minimum. By examining energy costs of security in a sensor network, it is clear the largest energy consumer is radio transmission. In the symmetric-key protocol SPINS [2], the radio transmission aspect of the protocol uses over 97% of the total security energy, while the actual encryption requires less than 3%. Studies in [12] show that for sensor networks, the energy/bit expended in radio transmission is three orders of magnitude greater than the energy/bit expended in AES symmetric-key encryption. Hence, minimizing transmission cost in a security architecture is a key design goal. However, merely reducing security transmission energy may require a tradeoff with other resources (such as memory) or even affect the robustness of the security mechanism itself. The purpose of this paper is to investigate this relationship between energy, memory, and security in sensor networks. We propose a framework called the SME (security-memory-energy) curve to quantify such tradeoffs.

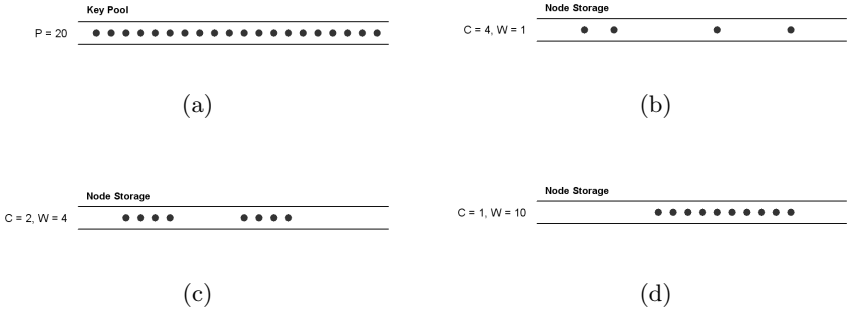

Fig. 1. Cluster Key Grouping Pre-Deployment Scenarios.

2 Cluster Key Grouping

As a driver for our investigation into energy-memory-security tradeoffs, we first present an energy-scalable protocol called cluster key grouping. In cluster key grouping, a key pool of P keys is generated off-line as shown in Fig. 1a. Prior to deployment, each node is randomly programmed with C clusters of keys, each cluster having a width of W keys per cluster; the total number of keys stored in each node is $K = C \cdot W$. All nodes are programmed with the same number of clusters C, with each cluster having the same width W. The key ring model presented in [4] is the specific case where $W = 1$ and thus $K = C$, as shown in

Fig. 1b. In the scenario in Fig. 1c, each node is programmed with $C = 2$ clusters, each of $W = 4$ keys; in the scenario in Fig. 1d, each node is programmed with only one large cluster of width $W = 10$.

Upon deployment, each node broadcasts the starting address of each of its C clusters. The remaining $W - 1$ addresses of each cluster are not broadcast since they are implicitly known from the starting address. If two nodes share at least one key between them, then a connection can be established based on the shared key and a secure link is said to be formed between them. The probability that the entire network is securely connected is related to P_c, the graph connectivity. The graph connectivity is a function of p, the probability two nodes share at least one key between them, also called the overlap probability. Given N nodes in a network, the desired graph connectivity can be calculated using the equation [4]:

$$P_c = \lim_{N \to \infty} Pr[G(N, p) \text{ is connected}] = e^{e^{-c}} \tag{1}$$

where c is a real constant, and $p = \ln(N)/N + c/N$. For an N-node network, a p can be specified to meet the required graph connectivity.

2.1 Effects of C and W on p

For a specified overlap probability p, we now investigate how C and W are used to generate p. As shown in the Appendix, p is derived as a function of the number of clusters C as:

$$p = 1 - \left(\frac{P - 2CW + 1}{P - CW + 1}\right)^{2C} \cdot \frac{(P - CW + 1)^2}{P \cdot (P - 2CW + 1)}. \tag{2}$$

This equation can be used as follows: a desired probability of overlap p is given as a specification. For a certain cluster size C, this equation determines the cluster width W required to obtain the desired p.

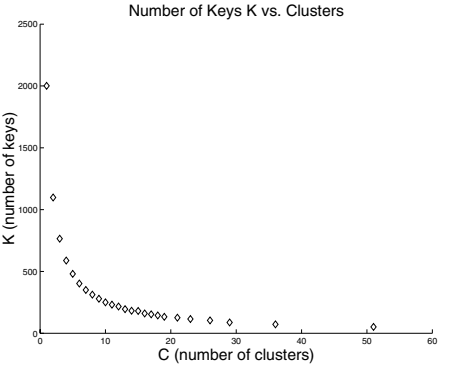

Fig. 2. Total Number of Keys in Cluster Key Grouping (to achieve $p = 0.4$).

Fig. 2 shows the total number of keys required ($K = C \cdot W$) for different C values, given $P = 10,000$ and the specified $p = 0.40$. At one extreme is the case presented in [4], where $W = 1$ and hence $K = C \cdot W = C$, as in Fig. 1b. In this case, by spreading the keys evenly throughout the entire key space, the lowest $K = 51$ total number of keys is required to obtain the specified p. At the other extreme is the case where only one wide cluster is stored, hence $C = 1$ and $K = C \cdot W = W$, as in Fig. 1d. Since all the keys are forced into one cluster and are not spread over the key pool, this case requires the highest $K = 2001$ total number of keys to obtain the specified p.

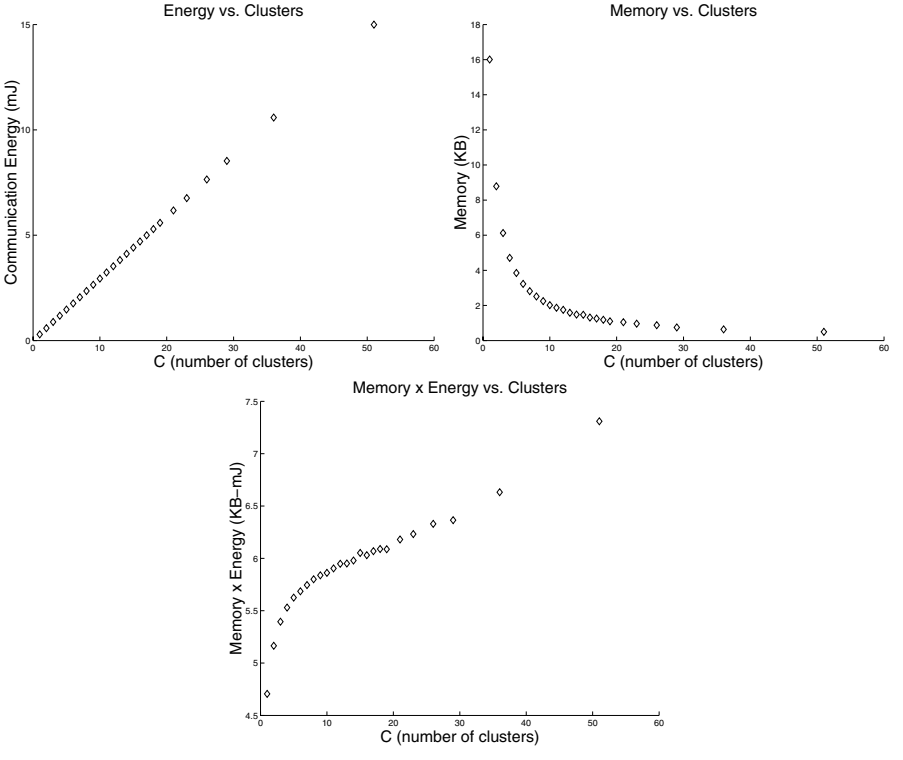

Fig. 3. Memory and Energy Requirements in Cluster Key Grouping (to achieve $p = 0.4$).

3 Energy-Memory Tradeoffs

With a driver protocol in place, we now examine the tradeoffs in security design. As stated earlier, the cluster-key grouping protocol is energy-scalable. By this we mean that the transmission energy expended to complete key agreement varies

based upon the value of C chosen. Recall that upon deployment each node must broadcast C starting addresses to its neighbors at a cost of E_b Joules per bit, requiring a total transmission energy of:

$$\text{energy} = C \cdot E_b \cdot \lceil \log_2 P \rceil \ \ [\text{Joules}]. \tag{3}$$

In the following simulations we assume the transmission energy of a Sensoria WINS NG node broadcasting at 10 mW RF over 900 meters, with a value of $E_b = 21$ mJ/bit [12]. (It is important to note that the energy factor does not take into account the reception energy or the energy required for re-broadcasting due to collisions in a CSMA scheme.) It is clear that the required transmission energy increases linearly as C increases, as seen in Fig. 3 for $P = 10,000$, $keysize = 64$, and the desired $p = 4$.

However, by choosing different values of C, the memory requirements of the security architecture are also affected, but with an opposite trend. In this sense, the protocol can also be considered memory-scalable. Each node requires memory to store $K = C \cdot W$ total keys, each of $keysize$ bits (i.e. 64, 128, etc.). Each node also must store the starting address of each of C clusters, with each address requiring $\lceil \log_2 P \rceil$ bits of memory. Hence the total memory requirement of each node is:

$$\text{memory} = K \cdot keysize + C \cdot \lceil \log_2 P \rceil \ \ [\text{bits}]. \tag{4}$$

These memory requirements fall quickly as C increases, as seen in Fig. 3. Thus for a specified overlap probability p, though energy requirements increase as C increases, memory requirements decrease as C increases. This leads to a tradeoff between the two physical metrics called the weighted memory-energy curve, which is the memory multiplied by the energy. (A weighing factor is introduced to compensate for the effects of varying overlap probabilities.) This is also shown in Fig. 3.

Table 1. Corner Cases of Cluster Key Grouping.

W	C	K	p_{actual}	Memory	Energy	Memory·Energy
1	51	51	0.40787	497 bytes	15 mJ	7.31 Kbytes-mJ
2001	1	2001	0.40010	16,000 bytes	0.294 mJ	4.71 Kbytes-mJ

To consider these tradeoffs in more detail, the two corner cases are examined in Table 1. The corner cases are defined as the extreme scenarios of C: the $W = 1$ ($C = K$) case, where each cluster is only one key, and the $C = 1$ case, where only one wide cluster is stored. From a pure communication energy standpoint, the $C = 1$ case requires 51 times less energy than the $W = 1$ case, since each node broadcasts only one starting address rather than 51 starting addresses to its neighbors. If energy were the only design factor, it is obvious this is the optimal solution. However, from a memory standpoint the $C = 1$ case also requires 32

times more memory than the $W = 1$ case, since a greater number of keys are required to obtain the desired overlap probability p. Thus, a tradeoff between memory and energy—the memory-energy curve—is quite apparent. The values of the memory-energy curve for the $W = 1$ and $C = 1$ cases are 7.31 and 4.71 Kbytes-mJ, respectively, which are on the same order of magnitude. Thus designing for the minimum of the memory-energy function would lead to using the $C = 1$ scenario.

Between the two corner cases lies the wide range of energy and memory utilization seen in Fig. 3. By varying the factor C, the cluster key grouping protocol can be energy-scaled (or conversely, memory-scaled) to meet the resource constraints of the network, with all permutations meeting the overlap requirement p.

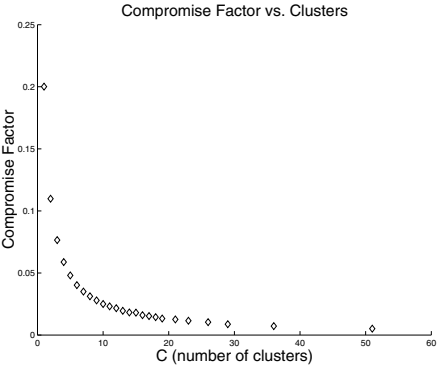

Fig. 4. Compromise Factor in Cluster Key Grouping (to achieve $p = 0.4$).

4 Security Tradeoffs

The above tradeoffs involve only the physical metrics of energy and memory. The question that remains is how security robustness is affected by such tradeoffs. Since security is an abstract concept, we must first formulate a "metric" for security that can be then traded off with the physical metrics. This metric will be called the security leakage factor, which quantifies the significance of a security breach to a system.

We begin by introducing the notion of the compromise factor, which is the number of keys compromised if a single node is compromised divided by the key pool, hence $compromise\ factor = K/P$. The compromise factor is plotted for different values of C in Fig. 4. Clearly, the greater the key spread over the key pool (i.e. the larger the number of clusters C), the fewer total keys K that are required, creating a lower total compromise factor. In order to quantify the effects of such a compromise on the security architecture, the security leakage factor is defined. The security leakage factor (SLF) is a function of the compromise factor:

$$security\ leakage\ factor = 1 + s_w \cdot compromise\ factor \tag{5}$$

where s_w is the security weight, which is any natural number. Security weight can be any positive real number, but for the sake of illustration natural numbers are used in this paper. As stated earlier, the security leakage factor—and the security weight in particular—attempts to roughly quantify the importance a security compromise has to a network. For example, if a network for whatever reason is not affected by a compromise or assumes that one cannot be made, then $s_w = 0$ would be assigned. If a network is extremely sensitive to key compromise, then a higher security weight (e.g. $s_w = 5$) would be assigned.

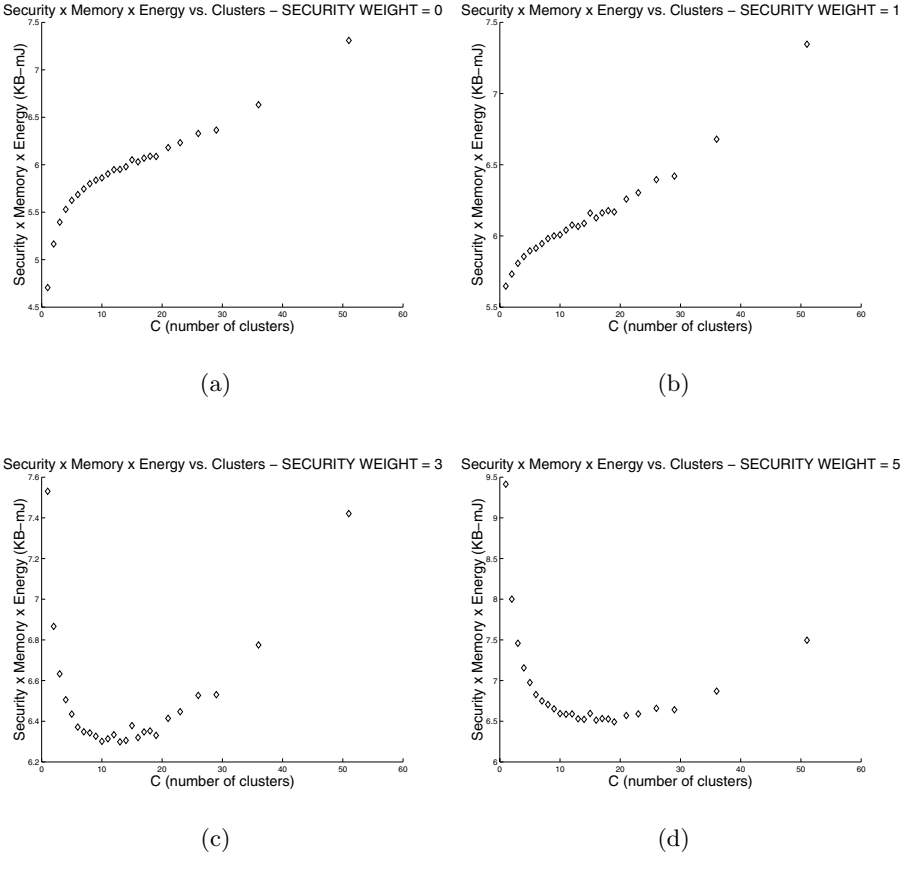

Fig. 5. Security-Memory-Energy Tradeoffs in Cluster Key Grouping (to achieve $p = 0.4$).

With a security metric in place, a comparison framework can be defined which quantifies the tradeoffs between security, memory, and energy. We call

this framework the SME curve, which is defined as *security leakage factor* · *memory* · *energy*. Fig. 5 shows SME curves for different security weights. As can be seen, depending on the security weight, there is a minimum of the curve along a particular number of clusters C. The case of $s_w = 0$ (no security effects) is the memory-energy curve mentioned earlier whose minimum is at $C = 1$. When security has a low priority ($s_w = 1$), the SME follows a curve similar to the weighted memory-energy function. However, when security leakage is more important ($s_w = 3$), then the curve alters into a reverse bell with a minimum at $C = 13$ clusters. This indicates that a network can have a minimal loss of security coupled with minimal energy consumption and memory requirements (according to our definitions) by choosing this cluster size. The sudden increase in the metric at smaller values of C is due to the increased importance of the compromise factor, which is large for small cluster numbers. (This characteristic increase begins at the security weight of approximately 1.8). At a security weight of $s_w = 5$ the minimum of the curve shifts to the right (towards a lower compromise factor) and hence a minimum is achieved at $C = 19$ clusters. As s_w increases to even larger values, the "tail" of the curve occurring at larger values of C continues to decrease until eventually the curve mimics the compromise factor itself; for $s_w = 65$ and greater, the minimum SME is always at $C = 51$ clusters. Table 2 summarizes these results.

Table 2. Security Tradeoffs in Cluster Key Grouping.

s_w	C (minimum)	Security-Memory-Energy
0	1	4.71 SLF-Kbytes-mJ
1	1	5.65 SLF-Kbytes-mJ
3	13	6.40 SLF-Kbytes-mJ
5	19	6.49 SLF-Kbytes-mJ
65	51	9.73 SLF-Kbytes-mJ

Therefore, by designing for the minimum of the SME curve instead of the minimum of energy curve, tradeoffs between energy, memory, and security can be taken into account. Though the security leakage factor is specific to the cluster key grouping model, it demonstrates the notion that security can be quantified as a metric, which can then be used to perform tradeoffs with traditional metrics such as energy, memory, processing latency, etc. In security architecture design, such a metric and such a framework are necessary to measure the sum effect of dimensionally-different metrics and to allow for fair comparison between similar protocols.

5 Related Work

In this section of the paper, we evaluate prior art sensor network security models. There are two simple key management schemes that can be used in sensor

network security. The first scheme is the pre-deployment programming of a single universal key into each of the N nodes of the network. In this scheme the key distribution problem is solved and nodes can communicate with one another securely by encryption via this universal key. However, security compromise is catastrophic in that if one of the N nodes is compromised, the security of the entire network is compromised. The other simple key management scheme is the pre-storage of pair-wise node keys, where each node stores a separate key for communication with each of the other $N-1$ nodes in the network, with the total number of keys in the network being $N \cdot (N-1)/2$. Clearly if $N = 10,000$, this mechanism is impractical in terms of memory constraints. Renewability is also an issue in this scheme, in that it is difficult to add new nodes to the network, unless extra renewal keys are pre-stored in each node before initial network deployment. Looking past these cases, there have been other proposals to address sensor network security.

[2] introduces the protocol suite SPINS (Security Protocols for Sensor Networks). SPINS includes two symmetric-key cryptographic protocols: μTESLA for authenticated broadcast from basestations (powerful nodes) to nodes, and SNEP for data confidentiality, authentication, integrity, and freshness. The SPINS suite uses counters and rough time stamps to ensure freshness. In SNEP, each basestation shares a secret key with each node in the network. In order for two nodes to communicate securely with one another, they must each go through a protocol with the basestation in order to obtain a node-to-node encryption key. Hence, SPINS has a potential problem with scalability, as the basestation may become a computational bottleneck if the number of nodes requesting keys overloads its bandwidth or computational capacity. SPINS also requires all nodes to be in the radio range of the basestation.

The scheme presented in [3] introduces another symmetric-key scheme in which all nodes share an initial universal key K_{GI}. This key is used as a root to generate other keys and allows for groups of nodes to elect local leaders (clusterheads) to control key management. A universal traffic encryption key (TEK) is generated from the universal key K_{GI} via hash functions and allows for all the nodes to communicate with one another in the network. This TEK is periodically updated to ensure freshness. One assumption mentioned in the paper is that tamper-proofing is assumed for each of the nodes. In some deployment scenarios (i.e. high security applications) tamper-proofing every node is possible; in other scenarios tamper-proofing may not be as feasible.

[4] presents a key distribution and establishment scheme based on the principle of a key pool and key ring. First, a large pool of P random keys is generated off-line. Prior to deployment, each node is programmed with K keys selected from the key pool using the sampling without replacement probability model. The set of K keys is called the key ring. Upon deployment, a discovery protocol begins in which each node broadcasts, in cleartext, its K key identifiers to its neighbors. If two neighboring nodes discover that a key is shared between them, then a secure connection can be established based on that shared key. This model is what was referred to as the basic key ring model.

[5] expands on the principles of [4] with three new techniques: 1) the q-composite scheme requires q keys to overlap, rather than a single key, to establish a secure connection, 2) the multipath-reinforcement scheme allows for a key update that requires correct communication from multiple paths to regenerate the key, and 3) the random-pair-wise scheme allows for node-to-node authentication. [6] presents a Blom pre-distribution scheme intended to improve resiliency of the network to node capture.

For general ad hoc network security, [8] proposes a public-key cryptography solution, in which threshold cryptography is used. Threshold cryptography is a technique in which a group of nodes form a certificate authority, which essentially verifies the link between an entity and his public key. [7] also presents an ad hoc security model using threshold cryptography and public-key encryption and decryption. The protocol mentioned in [9] uses a modified PGP (pretty good privacy) model to implement key management. However, public-key techniques, while possibly applicable for general ad hoc networks, may not be suitable for sensor networks based on energy considerations. [12] shows that public-key encryption energy requirements are orders of magnitude greater than symmetric-key encryption requirements. For example, on a MC68328 DragonBall processor, a 1024-b encryption for the symmetric-key AES algorithm requires 0.104 mJ. In contrast, a corresponding public-key encryption using RSA requires 42 mJ, which is 400 times more than AES. [13] also demonstrates that the energy cost of elliptic curve public-key cryptography is two to three orders of magnitude greater than Rijndael symmetric-key cryptography.

6 Conclusion

Energy-driven design is important in the architecture of secure sensor networks. However, energy is not the only criterion that must be factored into the development of a security architecture. This paper has demonstrated—via a protocol called cluster key grouping—that there are inherent tradeoffs involved between energy, memory, and security. By quantifying security as a metric and creating a comparison framework (the SME curve), an optimal architecture can be designed which factors in all such metrics.

Acknowledgements. The authors wish to acknowledge the support of the NSF (CCR-0098361), the Fannie and John Hertz Foundation (DH), and the Belgian National Science Foundation (FWO-G.0450.04).

References

1. A. Kansal and M. B. Srivastava. An environmental energy harvesting framework for sensor networks. *Proc. Int. Symposium on Low power Electronics and Design (ISLPED 2003)*, pp. 481-486, Aug. 2003.

2. A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. *Proc. 7th ACM Mobile Computing and Networks (MobiCom '01)*, pp. 189-199, July 2001.
3. S. Basagni, K. Herrin, E. Rosti, and D. Bruschi. Secure pebblenets. *Proc. 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '01)*, pp. 156-163, Oct. 2001.
4. L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. *Proc. 9th ACM Conference on Computer and Communications security (CCS '02)*, pp. 41-47, Nov. 2002.
5. H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. *Proc. 2003 IEEE Symposium on Research in Security and Privacy*, pp. 197-213, May 2003.
6. W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. *Proc. 10th ACM Conference on Computer and Communications security (CCS '03)*, pp. 42-51, Oct. 2003.
7. J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad hoc networks. *Proc. 9th IEEE International Conference on Network Protocols (ICNP '01)*, pp. 251-260, Nov. 2001.
8. L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network*, pp. 24-30, Nov./Dec. 1999.
9. J.-P. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. *Proc. 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc '01)*, pp. 156-163, Oct. 2001.
10. A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *IEEE Computer*, pp. 54-62, Oct. 2002.
11. F. Stajano and R. Anderson. The resurrecting duckling: security issues for ad-hoc wireless networks. *Proc. 7th Int. Workshop on Security Protocols*, Springer-Verlag, 1999.
12. D. W. Carman, P. S. Kruss, and B. J. Matt. Constraints and approaches for distributed sensor network security. NAI Labs Technical Report #00-010, Sept. 2000.
13. A. Hodjat and I. Verbauwhede. The energy cost of secrets in ad-hoc networks. *Proc. IEEE CAS Workshop on Wireless Communication and Networking*, Sept. 2002.

Appendix

To derive the overlap probability p, one must first consider the number of ways C non-overlapping clusters of width W can be arranged along a key pool of size P. It is clear that one cluster can have its starting point at any of P possible positions, giving P possible arrangements for $C = 1$. For two clusters, the first cluster may start in any of P ways, and the second cluster can start in any of $P - W - (W - 1) = P - 2W + 1$ positions in modulo P arithmetic. The $P - W$ term indicates the number of starting positions available after the first cluster is placed. The $W - 1$ term indicates the positions unavailable at the end of the key space (because a W width cluster cannot fit in a space of $W - 1$ or less, lest the clusters overlap). Thus the total number of ways for two clusters to be arranged is $P \cdot (P - 2W + 1)$. Continuing this principle and performing nested

calculations to larger cluster sizes, the total number of ways C clusters of width W can be arranged given a key pool of size P is

$$P \cdot \sum_{A_3=1}^{P-CW+1} \sum_{A_4=1}^{A_3} \cdots \sum_{A_C=1}^{A_{C-1}} A_c \tag{6}$$

for $C \geq 3$. For large values of P, we use the equation

$$\sum_{x=1}^{n} x^r \approx \frac{n^{r+1}}{r+1} \tag{7}$$

to obtain an approximate number of arrangements as:

$$P \cdot \frac{(P - CW + 1)^{C-1}}{(C-1)!}. \tag{8}$$

The number must later be multiplied by $1/C$ to account for circular shifting possibilities. The probability p that two neighboring nodes share at least one key is $1 - \Pr[\text{no key shared}]$. The probability that no key is shared is the number of ways $2C$ non-overlapping clusters can be arranged, divided by the number of ways C non-overlapping clusters can be arranged in the first node and second node, respectively, multiplied by the ways $2C$ clusters can be put into two partitions of C clusters each:

$$\Pr[\text{no key shared}] = \frac{\binom{2C}{C} \cdot [\#\text{ ways to arrange } 2C \text{ nodes}]}{[\#\text{ways for node 1}] \cdot [\#\text{ways for node 2}]}. \tag{9}$$

Using the approximation earlier, p reduces to:

$$p \approx 1 - \frac{\binom{2C}{C} [P \cdot \frac{(P-2CW+1)^{2C-1}}{(2C-1)!} \cdot \frac{1}{2C}]}{(P \cdot \frac{(P-CW+1)^{C-1}}{(C-1)!} \cdot \frac{1}{C})(P \cdot \frac{(P-CW+1)^{C-1}}{(C-1)!} \cdot \frac{1}{C})} \tag{10}$$

which simplifies to

$$p \approx 1 - (\frac{P - 2CW + 1}{P - CW + 1})^{2C} \cdot \frac{(P - CW + 1)^2}{P \cdot (P - 2CW + 1)}. \tag{11}$$