In cloning attacks, an adversary captures a sensor node, reprograms it, makes multiple copies, and inserts these copies into the network. Cloned nodes subvert sensor network processing from within. In a companion paper [2], we show how to detect and remove clones from sensor networks using random key predistribution security measures. Keys that are present on the cloned nodes are detected by using authentication statistics based on key usage frequency. For consistency with existing random key predistribution literature, and ease of explanation, the network in that paper used an Erdos-Renyi topology. In the Erdos-Renyi topology, the probability of connection between any two nodes in the network is uniform. Since the communications ranges of sensor nodes are limited, this topology is flawed. This article applies the clone detection approach from [2] to more realistic network topologies. Grid and ad hoc topologies reflect the node connectivity patterns of networks of nodes with range limits. We provide analytical methods for choosing detection thresholds that accurately detect clones. We use simulations to verify our method. In particular we find the limitations of this approach, such as the number of nodes that can be inserted without being detected.

Keywords Sensor Networks; Clone Detection

1. Introduction

Sensor networks deployed in hostile environments are vulnerable to physical attacks; including node cloning. In a cloning attack, an adversary captures a sensor node and copies its state. She reverse engineers the node, reprograms it, makes multiple copies of it, and inserts these copies back into the network. Chan and Perrig [1] explain how clones can falsify sensor data; extract data from the network; and/or stage denial of service attacks. A typical sensor network will contain hundreds or thousands of nodes in a hostile environment. It will not be possible to constantly monitor nodes to detect potential tampering. Therefore it is necessary to develop a security approach for detecting and removing cloned nodes.

In [2] we present a statistical approach for detecting cloning attacks in networks whose security guarantees are based on random key predistribution. In random key predistribution, first introduced in [3], before deployment each node is equipped with a key-ring of \( k \) keys chosen at random from the same large pool of \( P \) keys. Given \( k \) and \( P \), there is a well defined probability that any two nodes will share at least one cryptographic key and use it.
symmetric key cryptography to communicate. After deployment, nodes find their common keys and can communicate securely. A full discussion of random key predistribution, cloning issues and motivation for this work is in [2] and references therein.

Each node uses its keys as an authentication token. We analyze statistics on how often keys are used for authentication to determine which keys are being used by cloned nodes. Note that node authentication occurs only during network initialization. In this paper key usage refers only to the node authentication process, it does not refer to the number of times a key is used to encrypt or decrypt individual packets in the sensor network.

Each node constructs a Bloom filter [4] of the keys it has in common with its neighbors (i.e., the neighbor’s authentication credentials). This Bloom filter is securely transmitted to a server that records key usage data. The use of Bloom filters significantly reduces the traffic volume and provides greater security since keys are not sent over the network. The server creates a key usage histogram from the Bloom filters. An extension of this approach that does not require a centralized collection point is in [5]. We do not present that approach, since it contains complications that are not relevant to determining the clone detection thresholds discussed here. In [2], we derive the probability density function for key usage, and show how the false positive rate \( \alpha \) determines the key usage threshold. It also determines at what point the network fractures and becomes unusable.

Keys whose use exceeds the threshold value are thought to be used by clones and are ostracized from the network. The key detection protocol succeeds in removing all cloned keys from the network when a high false positive rate \( \alpha \) is allowed. In [2], we show that using large values for \( \alpha \) does not fracture networks with Erdos-Renyi topologies. The security analysis in [2] shows how our protocol tolerates many common attacks.

Unfortunately, the Erdos-Renyi model is not realistic for wireless sensor networks. It posits that a connection between any two nodes is equally likely, which does not match wireless communication with limited communications ranges. The Erdos-Renyi model is used in the literature [3, 6] because of its simplicity and because it has been widely studied in Mathematics literature [7].

This paper adapts the approach from [2] to grid and ad hoc network topologies [8] that better reflects the range constraints of wireless sensor networks. Grid networks are highly structured; each node can communicate only with a fixed number of immediate neighbors. Ad hoc networks are less structured; nodes are placed at random. They can only communicate with the other nodes within communication range \( r \). Each node’s neighborhood size is defined by a random process.

The remainder of this paper is organized as follows. Section 2 discusses the network models under consideration. Section 3 presents the clone detection protocol. Section 4 analyzes the proposed method and explains how we determine a threshold value for the number of times a key may be used. Simulation results verifying our analysis are presented in Section 5. Section 6 gives conclusions and possible future extensions to our research.

2. Sensor Network Topologies

This section presents two sensor network topologies, which we will use to analyze clone detection for realistic sensor networks using random key predistribution security.

2.1. Grid Network Topology

Grid network topologies are regular tessellations of the sensor field. The use of this topology for analyzing sensor networks originates in [8]. When users control node
placement, nodes are often distributed in a regular pattern. Grid topology analysis relies on results from percolation theory [9].

In this paper we analyze a two-dimensional rectangular grid. The same approach can easily be adapted to other tessellations, like those in [9]. Every node in the network (ignoring edge effects) has exactly four nodes in its neighborhood. Communications between a node and its neighbors occur only when they share a common key.

Figure 1a shows a 2-dimensional grid of nodes in a 100 by 100 meter square region. The 100 sensor nodes are placed at regular intervals with 10 meters between neighboring nodes. Figure 1b shows the communications links that are established between nodes that use a random key predistribution scheme.

2.2. Ad hoc Network Topology

In *ad hoc* networks, there is no control over node placement. A model of their topologies was proposed by [10]: the sensor network has $N$ nodes. These $N$ nodes are uniformly randomly distributed in a square region (i.e., the $x$ and $y$ coordinates are random variables in the range $[0..\text{max}]$.) They can communicate only with nodes that are within their communications range $r$.

[10]’s ad hoc model assumes that two nodes within range of each other communicate with probability 1. We allow nodes within range of each other to communicate with probability $p_c$ [11]. For modeling random key predistribution (with key pool size $P$ and $k$ keys in the keyring), $p_c$ is the probability that two nodes share a key [3]:

$$p_c = 1 - \left(1 - \frac{k}{P}\right)^{2(P-k+1/2)} \left(1 - \frac{2k}{P}\right)^{(P-k+1/2)}$$  \hspace{1cm} (1)

Figure 2a shows an example distribution of nodes in a square region. Again, 100 sensor nodes are placed in a 100 by 100 meter region. Figure 1b shows the communication links enabled by random key predistribution.

Unlike the Erdos-Renyi topology, nodes can not communicate if they are separated by a distance greater than the radio range $r$. Unlike grid networks, little is assumed about node placement. The *ad hoc* model better captures the topology of networks deployed in hostile environments.

![Figure 1.](image)

(a) (b)
3. Clone Detection Protocol

As shown in [2], cloning attacks can be detected by analyzing key usage statistics. We use the fact that each node randomly selects \( k \) keys from the pool without replacement to predict the distribution of the number of nodes possessing a given key. The distribution of the number of times a key is used is a direct consequence of the number of nodes possessing the key.

When clones are inserted into the network this key usage distribution is changed, since the cloned keys are present on a greater number of nodes than normal. The keys are therefore used more frequently than keys that have not been cloned. By collecting key usage statistics we can determine which keys have been cloned. Note that we consider only the use of keys to create a communications link between two nodes during network initialization.

The sensor network has a pool of \( P \) keys for a set of \( N \) authorized nodes. Each of the \( N \) nodes has a key ring of \( k \) keys taken randomly from the pool. \( C \) cloned nodes are added to the network. The total number of nodes is \( T = N + C \). A base station \( B \) is outside the network with a copy of the entire key pool. Every node has \( B \)’s public key that is used to communicate securely with the base station. \( B \) collects key usage statistics from the network and detects the keys that are used by cloned nodes.

We use counting Bloom filters to collect key usage statistics. Each node makes a counting Bloom filter of the keys it uses to communicate with neighboring nodes. It appends a random number (nonce) to the Bloom filter and encrypts the result using \( B \)’s public key. This encrypted data structure is forwarded to \( B \). \( B \) decrypts the Bloom filters it receives. The nonces are used to foil replay attacks; messages with duplicate nonces are discarded. \( B \) performs membership queries on the counting Bloom filters it receives and calculates how many times each of the keys are used in the network.

Keys used too often are considered cloned. \( B \) makes a Bloom filter of the cloned keys. For authentication purposes, the filter is encrypted using \( B \)’s secret key. The Bloom filter is broadcast through the sensor network using a gossip protocol [12]. Each node checks the keys in its key ring for membership in the Bloom filter of cloned keys. Cloned keys are removed from the local key ring and all connections using compromised keys are terminated. More details are in [2].

4. Threshold Detection

A sensor network has \( N \) nodes active in a region. The key pool contains \( P \) keys. Each node has a key ring of \( k \) keys chosen at random from the pool. Also let \( n \) denote the neighborhood
The expected number of times a key is used for making connections in the network is:

$$\mu_k = \frac{\sum_{j=1}^{n} M_j \cdot \left( \frac{N}{M_j} \right)}{p}$$

(2)

with variance:

$$V_k = \frac{\sum_{j=1}^{n} M_j \cdot \left( \frac{N_j}{M_j} - \mu_k \right)^2}{p}$$

(3)

(Note that $M_j$ is not canceled in (2), since it is required to compute the variance in (3.).

Where $M_j$ is the probability a key is on exactly $j$ nodes, and $N_j$ is the expected number of times a key on exactly $j$ nodes is used. If $P_d(j)$ the probability a key is on exactly $j$ nodes and $C_j$ denotes the number of times that key, $k_j$ on $j$ nodes is used for making connections in the network, then

$$N_j = P \cdot P_d(j) \cdot C_j \cdot \sum_{i=0}^{k-1} \frac{P_e(i)}{i+1}$$

(4)

In the case of Erdos-Renyi networks, $C_j$ (the maximum number of times that key, $k_j$ on $j$ nodes can be used for making connections in the network) is given by $\frac{\binom{j}{i-1}}{2}$ as indicated in [2]. This is not true for grid and ad hoc networks.

**Corollary 1.** The maximum number of times that key $k_j$ on $j$ nodes is used for making connections in the grid network, provided that no two of those $j$ nodes have any other key in common is:

$$C_j = j \cdot \sum_{i=1}^{4} i \cdot \binom{4}{i} \cdot \left( \frac{j-1}{N-1} \right)^i \cdot \left( \frac{N-j}{N-1} \right)^{4-j}$$

(5)

**Proof.** The number of times a key is used by a node for making connections is dependent on the number of nodes in the neighborhood with that key. For a key $k_j$ is located on exactly $j$ nodes in the sensor network and a node with $k_j$, the probability $p_i$ that the key is on $i$ other nodes in the neighborhood is:

$$p_i = \binom{n}{i} \cdot \left( \frac{j-1}{N-1} \right)^i \cdot \left( \frac{N-j}{N-1} \right)^{n-j}$$

(6)

The expected number of times a key that is on exactly $i$ other nodes in the neighborhood is used to make connections is:

$$i \cdot \binom{n}{i} \cdot \left( \frac{j-1}{N-1} \right)^i \cdot \left( \frac{N-j}{N-1} \right)^{n-j}$$

(7)
Since the neighborhood size is \( n \), \( i \) can vary from 1 to \( n \); making the expected number of times that the key is used by the node for making connections in its neighborhood:

\[
\sum_{i=1}^{n} i \times \binom{n}{i} \times \left( \frac{j-1}{N-1} \right)^{i} \left( \frac{N-j}{N-1} \right)^{n-j}
\]  

(8)

Because key \( k_j \) is on exactly \( j \) other nodes, the expected number of times it is used to make connections is:

\[
C_j = j \times \sum_{i=1}^{n} i \times \binom{n}{i} \times \left( \frac{j-1}{N-1} \right)^{i} \left( \frac{N-j}{N-1} \right)^{n-j}
\]  

(9)

For the grid network \( n = 4 \), making the number of times that a key on \( j \) nodes can be used to create connections in the network, provided that no two of those \( j \) nodes have any other key in common is:

\[
C_j = j \times \sum_{i=1}^{4} i \times \binom{4}{i} \times \left( \frac{j-1}{N-1} \right)^{i} \left( \frac{N-j}{N-1} \right)^{4-j}
\]  

(10)

Figure 3 shows simulation results and predicted values for mean key usage and variance as network size increases for grid networks.

**Corollary 2.** Maximum number of times a key, \( k_j \) on \( j \) nodes can be used to initialize communications links for an ad hoc network provided that no two of those \( j \) nodes have any other key in common is:

\[
C_j = j \times \sum_{i=1}^{n} i \times \binom{n}{i} \times \left( \frac{j-1}{N-1} \right)^{i} \left( \frac{N-j}{N-1} \right)^{n-j}
\]  

(11)

Where \( n = \sum_{i=0}^{N-1} i.p_i.p_c \), \( p_i \) is the probability \( i \) nodes are in range of the node and \( p_c \) is given by (1).

**Proof.** The expected number of times that the key \( k_j \) (on exactly \( j \) nodes) is used to initialize connections is given by equation (9). For an ad hoc network, there are \( N \) sensor nodes with communications range \( r \) randomly deployed in a square region of area \( A \). For a node has range \( r \), the area covered by its neighborhood is \( \pi r^2 \). Since \( A \) is the total area, the probability any node lies within the range of a given node is:

\[
p_r = \pi r^2 / A
\]  

(12)

Since all nodes have the same probability \( p_r \) of landing in the neighborhood, the number of nodes in the neighborhood will follow a binomial distribution. The probability that any node has \( i \) nodes in its neighborhood becomes:

\[
p_i = \binom{n}{i} p_r^i (1 - p_r)^{n-i}
\]  

(13)

The expected degree of a node in the network is therefore (where \( p_c \) is given by (1)):

\[
n = \sum_{i=0}^{N-1} i.p_i.p_c
\]  

(14)
In practice, nodes at a distance of r or less from the edge of the surveillance domain have a smaller neighborhood. Since the effective area where nodes can be placed is less than \( \pi r^2 \). These edge effects are a simulation artifact. We do not consider them in our analytical estimations. Luckily, they do not lead to any appreciable divergence between our analytical predictions, and the simulation verification. Figure 4 shows simulation results and predicted values for the mean and variance of (11) as network size increases.

We determine whether or not a key is cloned by comparing \( U_i \) (the number of times key \( i \) is used to establish connections indicated by the Bloom filter data) to \( \mu_k \) (from Equations (4), (5), and (11)). If \( U_i \) is significantly higher than \( \mu_k \), the key is probably being used by cloned nodes. A key is thought to be cloned if the key fits either

(i) a normal curve or

(ii) the distribution from Theorem with likelihood less than false positive rate \( \alpha \).

The value \( U_i \) which matches \( \alpha \) becomes the threshold \( T \). Keys used more than \( T \) times are marked as clones. A detailed discussion of hypothesis testing for threshold determination is in [2].
For an ad hoc sensor network of 900 nodes, key pool size $P = 1000$, key ring size $k = 50$, and range $r = 10$ m in a 100 m by 100 m region, Table 1 gives threshold values for various settings of $\alpha$ obtained analytically and using simulations. Normal distribution threshold values are obtained using statistical tables. Theorem 2 values are based on setting the $\alpha$

![Figure 4. Mean number of times a key is used to create communications links in an ad hoc sensor network: simulation (a) and analytical prediction (b). The associated variance: simulation (c) and analytical prediction (d). As with Fig. 3, $P = 1000$, $k = 50$, x axis is the number of nodes, 35 repetitions, and 95% confidence intervals.](image)

For an ad hoc sensor network of 900 nodes, key pool size $P = 1000$, key ring size $k = 50$, and range $r = 10$ m in a 100 m by 100 m region, Table 1 gives threshold values for various settings of $\alpha$ obtained analytically and using simulations. Normal distribution threshold values are obtained using statistical tables. Theorem 2 values are based on setting the $\alpha$

| Threshold ($U_i$) | Predictions |
|------------------|-------------|
|                  | Normal Distribution | Theorem 2 | Simulation |
| $\alpha$         |               |            |            |
| 0.75             | 14           | 15         | 14         |
| 0.50             | 19           | 20         | 19         |
| 0.20             | 24           | 26         | 25         |
| 0.05             | 29           | 33         | 32         |
P most used keys to being clones. Simulation values are the mean of 35 repetitions. Note the similarity of the results.

Figure 5 compares analytical predictions of the number of keys (out of 1000 keys) detected by the algorithm as clones for different values of false positive rate $\alpha$ and different number of clones inserted into the system for ad hoc networks. Figure 6 includes both true positives and false positives.

Table 2 gives threshold key use values for varying settings of the accepted false positive rate $\alpha$, for a grid sensor network of 1600 nodes with key pool size $P$ 1000 and key ring size $k$ 50. The threshold values are obtained using the same prediction and simulation approaches as for ad hoc networks.

Figure 6 gives analytical predictions of the number of keys (out of 1000 keys) detected by the algorithm as clones for different values of false positive rate $\alpha$ and different number of clones inserted into the grid network.

| $\alpha$ | Cloned Keys | $\alpha$ | Cloned Keys |
|----------|-------------|----------|-------------|
| 0.80     | 801         | 0.80     | 808         |
| 0.60     | 585         | 0.60     | 602         |
| 0.38     | 398         | 0.38     | 419         |
| 0.15     | 160         | 0.15     | 188         |
| 0.05     | 59          | 0.05     | 77          |

Figure 5. Number of keys detected as cloned in an ad hoc network as the number of clones inserted into the network varies.

| $\alpha$ | Cloned Keys  | $\alpha$ | Cloned Keys  |
|----------|--------------|----------|--------------|
| 0.80     | 808          | 0.80     | 808          |
| 0.60     | 603          | 0.60     | 603          |
| 0.38     | 423          | 0.38     | 423          |
| 0.15     | 201          | 0.15     | 201          |
| 0.05     | 102          | 0.05     | 102          |

| $\alpha$ | Cloned Keys  | $\alpha$ | Cloned Keys  |
|----------|--------------|----------|--------------|
| 0.77     | 607-860      | 0.77     | 618-866      |
| 0.48     | 301-607      | 0.48     | 319-618      |
| 0.23     | 113-301      | 0.23     | 132-319      |
| 0.05     | 37-113       | 0.05     | 49-132       |

| $\alpha$ | Cloned Keys  | $\alpha$ | Cloned Keys  |
|----------|--------------|----------|--------------|
| 0.77     | 622-866      | 0.77     | 622-866      |
| 0.48     | 328-622      | 0.48     | 330-622      |
| 0.23     | 148-328      | 0.23     | 155-330      |
| 0.05     | 54-148       | 0.05     | 63-151       |

Figure 6. Number of keys detected as cloned as the number of clones inserted into the grid network varies.
of clones inserted into the system for grid networks. Figure 6 includes both true positives and false positives.

5. Simulations

This section gives simulation results to verify our clone detection approach. We present results for both grid and ad hoc network topologies. The sensor networks have a key pool size \( P \) of 1000 keys. Each node has 50 keys in its key ring \( k \). The ad hoc network has 900 nodes. The grid network has 1600 nodes.

Figure 7 plots the maximum component size of the network versus the false positive rate \( \alpha \) for both grid and ad hoc networks. The figure shows an inflection point for component size when the value of \( \alpha \) is somewhere between 0.3 and 0.5. In ad hoc networks, the inflection point occurs when \( \alpha \) is approximately 0.8. False positive rates above those values will fracture the network and disable the sensor network. The prediction model in [2], predicts that the phase change in the ad hoc network occurs at \( \alpha = 0.62 \). Also a known result from percolation theory is that the inflection point for grid networks of 4 neighbors occurs when the

![Figure 7. Maximum component size as the false positive rate \( \alpha \) varies for (a) grid and (b) ad hoc topologies.](image)

| Threshold \( U_i \) | Predictions          | Normal Distribution | Theorem 2 | Simulation |
|---------------------|-----------------------|---------------------|-----------|------------|
| \( \alpha \)       |                       |                     |           |            |
| 0.77                | 5                     | 6                   | 4         |
| 0.48                | 6                     | 7                   | 6         |
| 0.23                | 7                     | 8                   | 8         |
| 0.05                | 8                     | 10                  | 11        |
probability of connection between any two nodes is 0.5. The corresponding value of $\alpha$ for grid networks is 0.47. These predictions are consistent with our simulation results.

Figure 8 shows simulation results for the number of keys (out of 1000 keys) that are detected as clones for different values of $\alpha$ and different number of clones inserted into the system. The values presented include both true and false positives.

Figure 9 shows the number of cloned keys detected for different $\alpha$ settings and different numbers of clones for both ad hoc and grid networks. These values do not include false positives. The keys that are detected are the keys that are most highly used in the network. As can be seen from the figures, larger values of $\alpha$ result in a larger number of cloned keys being detected, but this comes at the cost of a higher number of legitimate keys being falsely detected as clones. This is an acceptable tradeoff, since this result in very few legitimate nodes being rejected from the network.

Figures 10 and 11 present the results of running our clone detection protocol for different values of the false positive rate $\alpha$ and for a different number of cloned nodes in the network for grid and ad hoc networks respectively. A value of $\alpha = 0.48$ for Grid networks

Figure 8. Number of keys detected as cloned as the number of clones inserted into the network varies, for (a) ad hoc and (b) grid networks.
results in most cloned nodes being removed from the network. The value of $\alpha$ should be slightly less than 0.48, since 0.47 is the critical point where the network fractures. The protocol is more effective for the ad hoc topology, when $\alpha = 0.60$ all clones are eliminated when 90 or more clones are introduced. Even when a smaller number of clones are inserted in to the system, very few clones remain connected to the network. The clones’ ability to cause harm is severely restricted.

Figure 9. Number of cloned keys detected versus $\alpha$. The key ring size $k$ is 50. (a) Grid Networks and (b) ad hoc networks.

![Graphs](image)

Figure 10. Number of cloned nodes still connected to the grid network after running the protocol as $\alpha$ varies.

| $\alpha$ | Cloned Nodes still connected |
|---------|-------------------------------|
| 0.77    | 2                             |
| 0.48    | 5                             |
| 0.23    | 6                             |
| 0.05    | 7                             |

(a) 40 Clones

| $\alpha$ | Cloned Nodes still connected |
|---------|-------------------------------|
| 0.77    | 3                             |
| 0.48    | 8                             |
| 0.23    | 12                            |
| 0.05    | 13                            |

(b) 80 Clones

| $\alpha$ | Cloned Nodes still connected |
|---------|-------------------------------|
| 0.77    | 4                             |
| 0.48    | 11                            |
| 0.23    | 16                            |
| 0.05    | 18                            |

(c) 120 Clones

| $\alpha$ | Cloned Nodes still connected |
|---------|-------------------------------|
| 0.77    | 3                             |
| 0.48    | 10                            |
| 0.23    | 19                            |
| 0.05    | 24                            |

(d) 160 Clones
6. Conclusion

This paper focuses on sensor network cloning attacks where an adversary captures a sensor node, reprograms it, and adds multiple modified copies of the node to the network. We examine the feasibility of our clone detection protocol, first presented in [2] for ad hoc and grid network topologies. These topologies better represent realistic sensor network implementations, since they address communications range limitations.

We detect cloned keys by looking at how often they are used to establish connections between nodes; this is proportional to the number of nodes that possess the key. Inserting clones into the network skews this distribution. Cloned keys are present on more nodes and are used more often than keys that have not been cloned. The number of cloned keys detected increases with the false positive rate $\alpha$. We quantify the maximum value of $\alpha$, where the network will fracture for both grid and ad hoc networks. This is done using the method in [13].

Our ability to detect and remove clones increases with the number of clones in the network. Our approach does not require any cooperation from the cloned nodes. In [2], we show that the best strategy for the clones is to ignore the protocol. Active tampering with the protocol can only increase the likelihood that the clones will be detected.

We show how to quantify the number of clones that will not be detected for various number of clones inserted into the network. This is useful, in that it indicates a number of clones that may be present even if this protocol is used. Sensor network applications and security measures should be designed in a manner that can tolerate that many malicious participants.

One drawback of the presentation here and in [2] is the use of a centralized authentication mechanism. Centralized authentication using a base station is undesirable. It introduces a single point of failure into the network and can become a significant system bottleneck. It also

| $\alpha$ | Cloned Nodes still connected |
|---------|-------------------------------|
| 0.80    | 6                             |
| 0.60    | 13                            |
| 0.38    | 19                            |
| 0.15    | 20                            |
| 0.05    | 20                            |

Figure 11. Number of cloned nodes still connected to the ad hoc network after running the protocol as $\alpha$ varies.
violates random key predistribution’s support of network self-organization, which is a very appealing aspect of the approach. Using multiple key servers in [5] eliminates the need for having a centralized authentication mechanism. That approach is orthogonal to the issues addressed in this paper.

About the Authors

Richard Brooks (M’97–SM’04) received the B.A. degree in mathematical sciences from The Johns Hopkins University, Baltimore, MD, and the Ph.D. degree in computer science from Louisiana State University, Baton Rouge. He is currently an Associate Professor in the Holcombe Department of Electrical and Computer engineering, Clemson University, Clemson, SC. He was the Project Investigator (PI) of the Mobile Ubiquitous Security Environment Project sponsored by the Office of Naval Research (ONR) as a Critical Infrastructure Protection University Research Initiative. He was the Co-PI of a National Institute of Standards and Technology Project defining security standards for networked building control systems. He has had other research projects funded by the Army Research Office, by the ONR, and by the Defense Advanced Research Projects Agency. He has a broad professional background with computer systems and networks. He has been a Technical Director of Radio Free Europe’s computer network for many years. His consulting clients include the French Stock Exchange Authority and the World Bank. He is the author of Disruptive Security Technologies with Mobile Code and Peer-to-Peer Networking (CRC, 2004). His current research interests include distributed strategic systems for network security and national defense. Dr. Brooks received an Exemplary Achievement Certificate from the Louisiana State University Graduate School for his Ph.D. dissertation.

Vijaykrishnan Narayanan is a Professor of Computer Science and Engineering at The Pennsylvania State University. His research interests are in Computer Architecture, Embedded Systems and Nanoarchitectures. He is a senior member of IEEE and a member of ACM.

Mahmut Kandemir is an associate professor in the Computer Science and Engineering Department at the Pennsylvania State University. He is a member of the Microsystems Design Lab. Dr. Kandemir’s research interests are in optimizing compilers, runtime systems, embedded systems, I/O and high performance storage, and power-aware computing. He is the author of more than 80 journal publications and over 300 conference/workshop papers in these areas. His research is funded by NSF, DARPA, and SRC. He is a recipient of NSF Career Award and the Penn State Engineering Society Outstanding Research Award.

P. Y. Govindaraju received the MS degree in computer engineering from Clemson University, Clemson, SC in 2005. He currently works for Value Momentum in Minneapolis, Minnesota. His research interest is developing secure sensor networks.

Matthew Pirretti received his Ph.D. in Computer Science and Engineering from the Pennsylvania State University in 2007. He currently works for Motorola Corporation.

References

1. Haowen Chan and Adrian Perrig, “Security and privacy in sensor networks,” IEEE Computer Magazine, pp. 103–105, 2003.
2. R. R. Brooks, P. Y. Govindaraju, M. Pirretti, N. Vijaykrishnan, and M. Kandemir, “On the detection of clones in sensor networks using random key predistribution” IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, vol. 37, no. 6, pp. 1246–1258, 2007.
3. Laurent Eschenauer and Virgil D. Gligor, “A key management scheme for distributed sensor networks,” in Proceedings of the 9th ACM Conference on Computer and Communications Security, pp. 41–47, 2002.

4. B. Bloom, “Space/time trade-offs in hash coding with allowable errors,” Communications of the ACM, vol. 13, no. 7, pp. 422–426, July 1970.

5. R. R. Brooks, B. Pillai, M. C. Weigele, and M. G. Pirretti, “Multicast Encryption Infrastructure for Security in Sensor Networks,” International Journal of Distributed Sensor Networks, vol. 5, no. 2, pp. 139–157, 2009.

6. H. Chan, A. Perrig, and D. Song, “Random key predistribution schemes for sensor networks,” in Proceedings of the 2003 IEEE Symposium on Security and Privacy, 2003.

7. B. Bollobás, Random Graphs, Cambridge University Press, Cambridge, UK, 2001.

8. S. S. Iyengar and R. R. Brooks, eds, Distributed Sensor Networks, Chapman & Hall, Boca Raton, 2005.

9. D. Stauffer, and A. Aharony, Introduction to Percolation Theory, Taylor & Francis, London, 1992.

10. Bhaskar Krishnamachari, Stephen B. Wicker, and Ramon Bejar, “Phase transition phenomena in wireless ad-hoc networks,” Symposium on Ad-Hoc Wireless Networks, GlobeCom2001, San Antonio, Texas, November 2001.

11. R. R. Brooks, D. Friedlander, J. Koch, and S. Phoha, “Tracking multiple targets with self-organizing distributed ground sensors,” Journal of Parallel and Distributed Computing Special Issue on Sensor Networks, vol. 64, no. 7, pp. 874–884, August 2004.

12. R. R. Brooks, Disruptive Security Technologies with Mobile Code and Peer-to-Peer Networks, CRC Press, Boca Raton, 2005.

13. R. R. Brooks, B. Pillai, S. Racunas, and S. Rai, “Mobile network analysis using probabilistic connectivity matrices,” IEEE Transactions on SMC, Part C, In Press, January, 2006.
Submit your manuscripts at http://www.hindawi.com