# Addressing Unique Control System Concerns in Wireless Networks: Processor Based Denial of Service Attacks and Recovering from Key Compromise

Kun Sun
Intelligent Automation, Inc.
Rockville, Maryland
ncsusunkun@gmail.com

**Abstract:** The use of wireless networks in control systems offers some different challenges from a traditional corporate wireless network, and this paper will address two of those challenges. The first is the relatively low capability CPU's found in controllers and instruments can be vulnerable to denial of service attacks if computation intensive public key functions are repeatedly required by attackers. To counteract this threat, the author proposes a polynomial-based weak authentication scheme to mitigate potential denial of service attacks against public key protocols. A secret key established by the polynomial-based weak authentication is used to authenticate the exchanged messages of public key protocols. Each node verifies the messages before running the expensive public key operations.

The second challenge addressed in this paper is recovering from a loss of a crypto key, which could occur if a security enabled controller or instrument is stolen. In many schemes, a key manager must update the group key to revoke the compromised nodes from the whole network. This paper proposes and describes the use of stateless group key update schemes to update the group key in wireless networks, especially when the wireless communication channel is unreliable. Stateless group key update schemes have two nice properties. First, a legitimate node can get the more recent group key as long as the node receives the corresponding key update message, even if the node is offline for a while, or misses several previous rounds of key update messages. Second, no coalition of revoked nodes, of an arbitrary size, can decrypt the new group key.

**Keywords:** Denial of Service, Key Management, Wireless Networks

## 1 Introduction

Industrial control systems (ICS's) are crucial to the operation of the U.S. critical infrastructures, such as energy, water treatment, agriculture and food, and transportation systems. Recent advances in wireless networks potentially allow wireless to enhance the capabilities of the existing monitoring and control infrastructure in the ICS's. Wireless networks can drastically reduce the cost of accessing information per point and provide unprecedented monitoring capabilities.

The use of wireless networks in control systems offers some different challenges from a traditional corporate wireless network, and this paper will address two of those challenges. The first is the relatively low capability CPU's found in controllers and instruments can be vulnerable to a denial of service attack if computation intensive public key functions are repeatedly required by attackers.

In traditional networks such as the Internet, Public Key Cryptography (PKC) has been the enabling technology underlying many security services and protocols (e.g., SSL [1] and IPsec [2]). There have been a few recent attempts to use PKC in wireless sensor networks [3][4][5][6], which demonstrate that it is feasible to perform PKC operations on the current sensor platforms such as MICAz motes [9] and Imote2 motes [10]. However, public key protocols are vulnerable to denial of service (DoS) attacks [11]. Because the PKC operations (e.g., modular exponentiation in RSA [7] or scalar multiplication in ECC [8]) are expensive in computation, the attacker can keep sending fake ID's and fake public keys to make legitimate nodes busy performing authentication or calculating secret keys. The underlying reason this attack can be successful is that there is no authentication on exchanged messages in public key protocols.

To defeat DoS attacks against public key cryptography in wireless networks, we propose a polynomial-based weak authentication scheme to help two nodes establish a secret key using a pre-distributed bivariate polynomial, and then use the secret key to authenticate the exchanged messages in public key protocols. The polynomial-based pairwise key establishment is hundreds of time faster than key establishment using public key protocols. However, this basic scheme can only provide weak authentication. Because when the degree of the secret polynomial is $t$, once $t+1$ nodes are compromised, the secret polynomial will be disclosed. To increase the security of the basic scheme, we propose a polynomial pool based authentication scheme that no adversary will be able to compromise any polynomial in the polynomial pool, even though she can compromise a large number of legitimate nodes.

The second challenge addressed in the paper is recovering from a loss of a crypto key, which could occur if a security enabled controller or instrument is stolen. In many schemes, a key manager must update the group key to revoke the compromised nodes from the whole network. Based on the interdependency of key update messages, group key update schemes can be classified into two categories, stateful [12][13] and stateless [14][15]. A stateful group key update scheme uses keys sent in a given rekeying instance to encrypt the key to be sent in the next rekeying instances. It has the disadvantage that a member that goes offline or misses a key update message during a key update instance cannot recover any future keys. In stateless group key update schemes, group key update messages are independent of each other. A member that goes offline misses only the data and the keys sent when it was offline. Once online, the member can decrypt future group key. Thus, stateless group key update schemes have two nice properties. First, a legitimate node can get the more recent group key as long as the node receives the corresponding key update message, even if the node is offline for a while, or misses several previous rounds of key update messages. Second, no coalition of revoked nodes, of an arbitrary size, can decrypt the new group key.

In wireless networks, the communication failures and collusions make it difficult to achieve an immediate group key update in many cases. In the worst case, the key update message may be totally lost. Two legitimate nodes cannot communicate if a more recent key update message only reaches one of them. Industrial control systems require us to minimize the outage for updating the global session (group) key and the delay for message transmission. However, no existing group key update scheme can guarantee that all nodes receive the latest group key after a key update. Maintaining a synchronized group key therefore becomes a non-trivial task, given the dynamic nature of wireless networks and the possibility of intermittent communication failures.

We present several security mechanisms to solve the second challenge. First, we adopt a stateless group key update scheme based on Subset Difference Rekeying (SDR) [14], which is the most efficient stateless rekeying algorithm proposed in the literature. Given $r$ revoked nodes in total $n$ nodes, the key manager only needs to send at most $2r-1$ (or $1.25r$ on average) key update messages. Next, we present a packet retransmission scheme in case of unsynchronized keys among neighbor nodes. Our method has the nice property that a legitimate node can get the updated group key from a received key update message, even if that node has missed several previous rounds of key updates. It also guarantees that whenever legitimate nodes communicate with each other, they will synchronize their group keys and agree on the latest one needed to authenticate and verify data packets.

The rest of the paper is organized as follows. In Section 2, we present the polynomial-based weak authentication schemes to mitigate DoS attacks on public key protocols. In Section 3, we describe the stateless group key update schemes with packet retransmission method. We conclude the paper in Section 4.

# 2 Polynomial-based Weak Authentication

In this section, we first present a basic scheme of using polynomial-based weak authentication to mitigate potential DoS attacks against public key protocols. Then, we study the security of the basic scheme, and point out that it cannot mitigate DoS attacks when a certain number of nodes are compromised. Next, to increase the security of polynomial based weak authentication, we propose a polynomial pool based weak authentication scheme that works well no matter how many nodes have been compromised.

## 2.1 Basic Scheme

Suppose we have a Certificate Authority (CA) or a key manager that generate and distribute private keys and related parameters to all nodes in the network. Each node in the network must first be initialized by communicating with the CA and receiving the keying materials. The basic idea of polynomial-based weak authentication is shown in Figure 1. The CA first generates a bivariate t-degree polynomial $f$ over a finite field $GF(q)$, where $q$ is a large prime number.

$$f(x, y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j \quad \text{where x, y, } a_{i,j} \in GF(q)$$

Function f satisfy the following property: *f(x,y) = f(y,x)*.

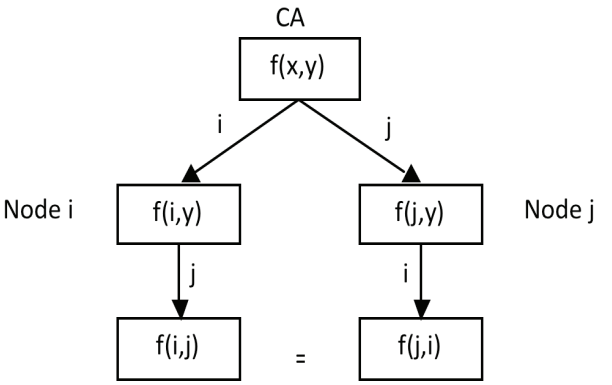


*Figure 1 – Polynomial-based weak authentication*

As shown in Figure 1, for node *i*, CA evaluates *x* in the bivariate polynomial *f(x,y)* by node ID *i*, and deploys f(i,y) onto *i*. For node *j*, CA evaluates *x* in *f(x,y)* by node ID *j*, and deploys *f(j,y)* onto *j*. When node *i* and node *j* want to communicate with each other, they can establish a pairwise secret key based on each other's ID since *f(i,j) = fj,i)*. Then they can use key *f(i,j)* to authenticate the exchanged messages for public key protocols. Because each node verifies the messages before running public key operations, DoS attack against public key protocols cannot succeed. We need to point out that this basic scheme is *t*-collision resistant, where *t* is the degree of polynomial. That means once *t+1* nodes are compromised, the secret polynomial *f* is disclosed.

Because the evaluation of a polynomial is much faster than the expensive public key functions, polynomial-based weak authentication scheme can be used to mitigate DoS attacks. We implement both polynomial-based weak authentication scheme and Elliptic Curve public key cryptography on Imote2 motes. Then, we compare the execution times of polynomial-based scheme and Elliptic Curve Diffie Hellman (ECDH) protocol as shown in Figure 2 and Figure 3 [17]. We found that polynomial-based pairwise key establishment is much faster than the key establishment using public key protocols. It only needs around 2 *ms* to establish a pair-wise key using the polynomial-based key establishment. It is 425 times faster than fixed key establishment and 700 times faster

than ephemeral key establishment in ECDH. It explains why polynomial-based weak authentication can defeat DoS attacks against public key protocols.

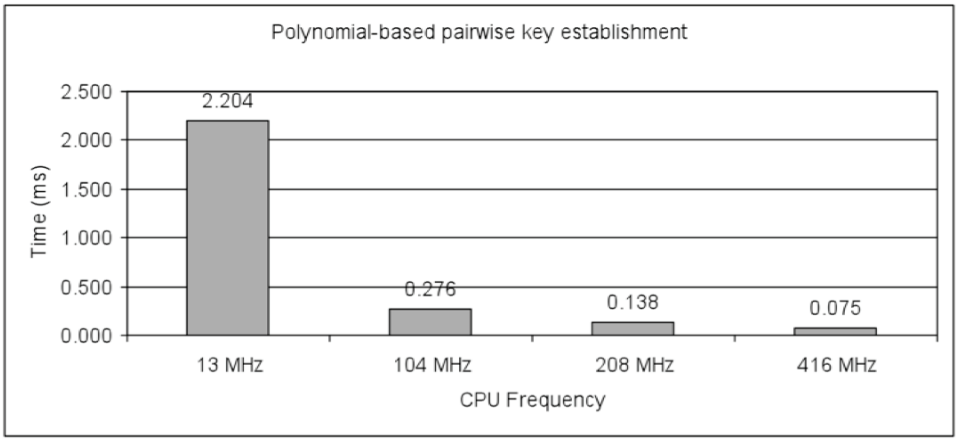


*Figure 2 – Computation overhead in polynomial-based scheme*

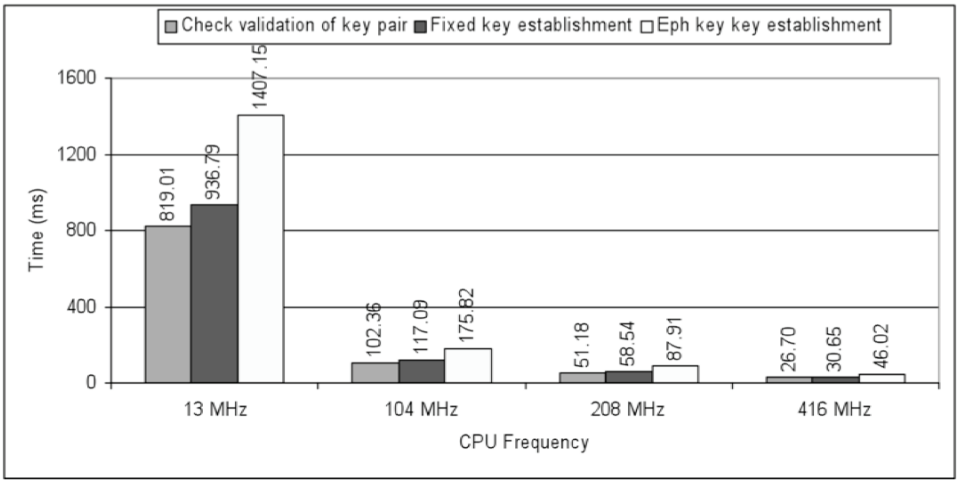


*Figure 3 – Computation overhead in Elliptic Curve Diffie Hellman*

## 2.2 Detection of Compromised Nodes and Polynomials

In order to bypass weak authentication and launch DoS attacks, the adversary needs to compromise regular nodes to obtain their pre-distributed polynomial key shares. We present a detection method to identify compromised nodes and polynomials.

**Detection of Compromised Nodes:** We use a simple threshold-based approach to detect compromise of regular nodes and polynomials. Specifically, each controller node keeps a counter $\mu_i$ for each regular node $i$ with which it has communicated with to

record the number of key establishment attempts (either complete or incomplete key establishment). The counter $\mu_i$ records the number of key establishment attempts in the past $w$ time units. When $\mu_i > \tau$, a predefined threshold, the controller node identifies node $i$ as a suspicious node. The adversary has the ability to complete key exchange using the private keys of compromised nodes together with their polynomial key shares. Thus, the counter $\mu_i$ should take into consideration both complete and incomplete key exchange attempts coming from regular nodes.

**Detection of Compromised Polynomials:** When a node is captured, the adversary not only obtains its private key, but also its polynomial key share. Using this polynomial, the adversary can bypass weak authentication for all other nodes that use the same polynomial. To deal with this problem, the key manager keeps track of the unique ID's of suspected nodes, and maintains them in terms of the ID's of the polynomials they are using. Specifically, for each polynomial with ID $f$, the key manager maintains a set $C_f$ of IDs of the suspicious nodes that have shares on the polynomial $f$. When a polynomial $f$ has $| C_f | > t$, where $t$ is the degree of the polynomial, the controller node marks $f$ as suspicious. As a result, all nodes that claim to have a share of polynomial $f$ are considered suspicious.

To facilitate the analysis of the detection method, we divide node compromise attacks into the following two cases. We assume the number of compromised nodes sharing the same polynomial $f$ is $n_f$.

- In **Case 1**, we assume that the number of compromised nodes sharing the same polynomial $f$ is fewer than $t$ (i.e., $n_f<t$). The adversary can launch limited DoS attacks only directly using the polynomial key shares of these compromised nodes.
- In **Case 2**, the adversary achieves to capture more than $t$ nodes per polynomial for up to $r$ different polynomials. Thus, the adversary can recover weak authentication keys on non-compromised, or even non-existing, nodes sharing the same polynomial $f$, which makes it possible to launch more powerful DoS attacks.

The threshold based detection method is effective and efficient in Case 1. In this case, the adversary can only use a few compromised nodes in order to bypass weak authentication to a key manager. Thus, after using the keying materials of a compromised node for more than $\tau$ times with $w$ time units, the key manager can identify the compromised node correctly based on the ID's used in weak authentication and add this node into the suspicious node list. The adversary may try to avoid detection by using the keying materials from each compromised node for no more than $\tau$ times with $w$ time units. However, this has successfully reduced the DoS attacks. In any case, this method can reduce the number of public key requests to no more than $t \cdot (\tau + 1)$ times within $w$ time units per polynomial, since the adversary can compromise at most $t$ nodes per polynomial.

In Case 2, the adversary can compromise more than $t$ regular nodes sharing the same polynomial, and as a result, achieves to recover polynomial $f$. This allows her to obtain pairwise keys of non-compromised or even non-existing nodes sharing the same

polynomial, and launch DoS attacks by sending in authentication requests. Suppose the adversary continues to use node *i*'s ID and keying materials to send in authentication requests. After $\tau$ attempts with $w$ time units, the controller node will adds node *i* into the suspicious node list and stop accepting any requests with node *i*'s ID, even if node *i* is not compromised. The adversary will thus be forced to switch to a different ID, using the keying materials recovered from the compromised polynomial. After using the $t$ different node ID's, the key manager will add the corresponding polynomial into the suspicious polynomial list. As a result, the adversary cannot use the same polynomial any more. Similar to Case 1, the detection method can reduce the number of public key requests to no more than $r \cdot (t+1) \cdot (\tau+1)$ times with $w$ time units using the compromised $r$ polynomials.

The bad news in Case 2 is that even if a node is not compromised, if the adversary has used its keying materials or exhausted $(t+1)$ nodes for a polynomial that it also shares, the node will be denied access to the network. This could be another form of DoS attacks against certain regular nodes.

In conclusion, our detection method is effective and efficient in Case 1, where the adversary has compromised a limited number of nodes but not any polynomial. However, when the adversary is able to compromise more nodes, as indicated in the analysis of Case 2 the detection method does not provide sufficient protection, even though it can still identify compromised nodes and polynomials. Nevertheless, Case 2 represents severe node compromise attacks. The adversary already owns the majority of the network. Detecting compromised nodes and polynomials does not make any sense any more.

## 2.3 Polynomial Pool based Authentication via Constrained Deployment

Our analysis in the above section demonstrates the limitation of the detection method. In this section, we seek an alternative method for efficient authentication that can handle node compromise attacks. The proposed scheme is adapted from the polynomial pool based key pre-distribution scheme [16] . To deal with potential node compromise attacks, we constrain the use of each polynomial to at most $t$ times, where $t$ is the degree of each polynomial. As a result, no adversary will be able to compromise any polynomial, even though she can compromise a large number of regular nodes.

### 2.3.1 Setup

The CA randomly generates a pool $\boldsymbol{P}$ of $m$ random $t$-degree bivariate polynomials. Each $f(x,y)$ in $\boldsymbol{P}$ is symmetric, that is, $f_i(x,y)=f_i(y,x)$. The CA pre-determines a range of possible ID's for regular nodes. Without loss of generality, assume the ID's are from 1 to $N$. In other words, this scheme can support up to $N$ regular nodes throughout the lifetime of the network. For simplicity, we assume $N=t \cdot m$.

We partition the range of regular IDs (i.e., $1 \ldots N$) into multiple equal-size intervals, each of which has $t$ IDs. Let $S_r = \{(r-1)t+1, (r-1)t+2, \ldots, (r-1)t+t\}$, $r = 1, \ldots, m$. That

is, $S_r$ is the set of $t$ node ID's in the $r$-th interval. Each interval $S_r$ is assigned the $r$-th polynomial $f_r(x,y)$ in $\boldsymbol{P}$.

We assume the ID's of key managers start from N+1. There is no limit on the number of key managers. Furthermore, we assume each key manager has tamper resistant hardware, and thus cannot be compromised to leak keying materials.

For each key manager with ID $c$, the CA computes a share for $c$ on each polynomial $f$ in $\boldsymbol{P}$, $f(c, y)$, and stores $f(c, y)$ on node $c$. As a result, the key manager c has $m$ polynomial key shares. For each regular node with ID $i$, the CA first identifies its corresponding interval index $r = \left\lfloor \frac{i+t-1}{t} \right\rfloor$. The CA then selects the $r$-th polynomial $f_r(x,y)$ in $\boldsymbol{P}$, generates a polynomial key share as $f_r(c, y)$, and stores $f_r(c, y)$ on the node.

Note that the generation and assignment of polynomial key shares for controller and regular nodes can be interleaved. Nodes can be deployed incrementally without any particular order.

### 2.3.2 Authentication

After deployment, each key manager periodically advertises its node ID through wireless broadcast. When a node $i$ needs to join the network, it first waits for an advertisement packet from a key manager, from which it extracts the key manager's ID, denoted $c$. Node $i$ then computes a pairwise key $K_{ci}=f_r(i,c)$ with its polynomial key share $f_r(i,y)$, and sends to the key manager $c$ an authentication request.

The key manager, after receiving the authentication request and finding out the requesting node's ID $i$, first extracts the interval index $r = \left\lfloor \frac{i+t-1}{t} \right\rfloor$, then selects the corresponding polynomial key share $f_r(c,y)$, and finally computes the pairwise key $K_{ci} = f_r(c,i)= f_r(i,c)$, which is shared between nodes $c$ and $i$. This key will be used by the key manager to authenticate node $i$. The actual authentication may follow any standard authentication protocol; we do not discuss it in detail.

## 2.4 Analysis

The security proof in [16] ensures that the use of $t$-degree bivarite polynomial is unconditionally secure and $t$-collusion resistant. That is, the coalition of no more than $t$ compromised nodes knows nothing about the pairwise key between any two non-compromised nodes. In the proposed scheme, all key managers are assumed to have tamper-resistant hardware, and thus cannot be compromised through node capture. Each polynomial provides polynomial key shares to at most $t$ regular nodes. As a result, even if all $t$ nodes are compromised, the adversary will not be able to learn anything about the polynomial. Thus, the proposed scheme is resistant to any node compromise attacks. That is, the compromise of any node does not disclose keying materials belonging to any other nodes.

The performance overhead on regular nodes is very light. The computation of a pairwise key involves the evaluation of a t-degree polynomial. The storage overhead involves

(t+1) numbers in the finite field $F_q$ ,i.e. $(t + 1)\lceil \log q \rceil$. The communication overhead follows the adopted authentication protocol.

The performance overhead on a key manager is similar to a regular node except that the controller node has to store *m* polynomial key shares. The storage overhead is thus:

$$m(t+1)\lceil \log q \rceil = \frac{N}{t}(t+1)\lceil \log q \rceil \approx N\lceil \log q \rceil$$

Assume the network to be deployed is expected to have up to 100,000 regular nodes and $\lceil \log q \rceil$ = 64 bits = 8 bytes. Each manager node has to store about 800 Kbytes. Thus, the proposed scheme does have medium storage requirement on the key managers. However, given today's high-end wireless platforms, the storage requirement can be satisfied.

# 3 Group Key Management

In wireless networks, secure network access is enforced by employing a network-wide (group) key, which is only known by approved nodes, to authenticate all the packets transmitted in the network. Due to the need to add new nodes and remove compromised or stolen nodes, this group key has to be updated from time to time. Therefore, a group key update system is required to guarantee all legitimate nodes agree on the same key at the time they communicate.

In the following, we first present a stateless group key update scheme based on Subset Difference Rekeying (SDR) [14]. Next, we use the one-way key chain mechanism and packet retransmission scheme to reduce out-of-sync time due to unsynchronized group keys in legitimate nodes.

## 3.1 Stateless Group Key Update Scheme

In a basic stateless group key update scheme, the key manager *c* uses the pair-wise secret key $K_{c,i}$ shared with each non-revoked node *i* to encrypt the new group key. This scheme works well for small groups but not well for large groups, due to the high communication overhead and high computation overhead on the key manager. It requires *n-r* symmetric key encryption and sending *n-r* encrypted group keys on the manager side, where *n* is the total number of group members and *r* is the number of the revoked nodes.

SDR is the most efficient stateless rekeying algorithm proposed in the literature. Given *r* revoked nodes in total *n* nodes, in one round of group key update, the key manager only needs to send at most *2r-1* (or *1.25r* on average) key update messages. It requires each node to store $(0.5*\log^2 N+0.5*\log N+1)$ keys, while in the basic scheme, each node only needs to store one key. SDR consists of two major parts: subset finding and key assignment. In subset finding, non-revoked nodes are partitioned into disjointed groups. Whenever we want to revoke compromised nodes, we can always find subset covers that only cover eligible nodes in the group. The key assignment can guarantee only the non-revoked nodes can derive the key to decrypt the new group key. In the following, we give a brief introduction on these two major parts.

**Subset Finding in SDR:** We first construct a binary tree with leaf nodes representing all nodes in the network, as shown in Figure 4. The nodes have unique IDs from *1* to *n*, aligned in an increasing order from left to right, as shown at the bottom of Figure 4.

Whenever we want to revoke compromised nodes, we can always find subset covers that only cover eligible nodes in the group. For example, we want to revoke two compromised nodes (node 4 and node 6) in Figure 4. The labels of node 4 and node 6 in the tree are 11 and 13, respectively. A leaf u is in $S_{i,j}$ if and only if it is in the subtree rooted at $v_i$ but not in the subtree rooted at $v_j$, where $v_i$ is an ancestor of $v_j$. For example, subset $S_{2,11}$ covers nodes 1, 2, 3, but not node 4; subset $S_{3,13}$ covers nodes 5, 7, 8, but not node 6. Therefore, we encrypt new group key using subset key of $S_{2,11}$ and $S_{3,13}$ and flood this message through the network. Only non-revoked nodes can decrypt the message and obtain the new group key.

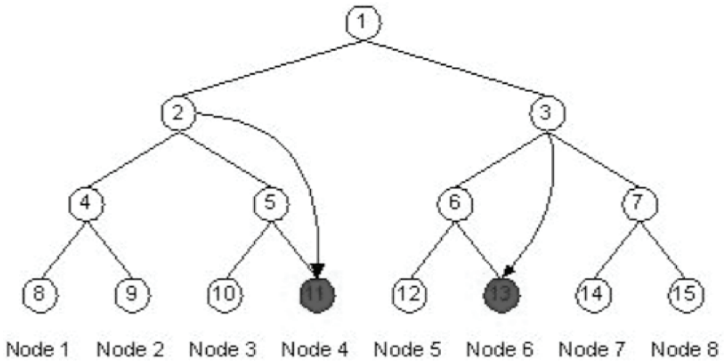


*Figure 4 – SDR tree*

**Key Assignment in SDR:** Each node $V_i$ has a random label $S$, as shown in Figure 5. $G_L(S)$ is the label for node $V_i$'s left child, and $G_R(S)$ is the label for its right child. The key for subset $S_{i,j}$ is $G_M(label\ of\ V_j)$. $G$ is a pseudo-random function. SHA-1 function can be used as the pseudo-random function G, for example, $G_L(x) = sha1(1||x)$, $G_M(x) = sha1(2||x)$, $G_R(x) = sha1(3||x)$. In Figure 5, $L_{i,j}$ is the subset key for subset $S_{i,j}$.

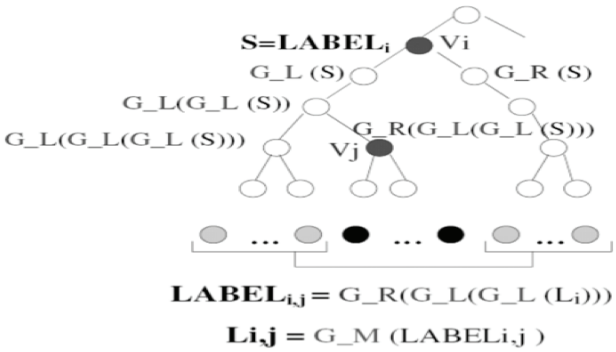


*Figure 5 – Key assignment in SDR*

## 3.2 Group Key in One-way Key Chain

An attacker may send bogus group key update messages, and this could lead to a resource consumption attack if the verification of a key update message involves expensive operations, such as public key cryptographic operations. We observe that group keys are intended for authenticating wireless packets only. As a result, backward secrecy is not a concern. In other words, once group keys are updated, the secrecy of group keys is not necessary any more. Based on this observation, we organize the group keys in a one-way key chain to facilitate the authentication of future keys based on previous ones. Therefore, a normal node can verify a future group key with a limited number of hash operations that are much cheaper than an asymmetric cryptographic operation.

A one-way key chain is a chain of keys generated through repeatedly applying a one-way hash function $H$ on a random number key seed. For instance, $k_{n-1} = H(k_n)$, ..., $k_0 = H(k_1)$. The property of one-way means, given a latest released key $k_i$ from a one way key chain, it is computationally infeasible for an adversary to find any unreleased key $k_j$ such that $H^{-i}(k_j)$ equals $k_i$. However, it allows a receiver to easily verify that a later key $k_x$ really belongs to the key chain by checking that $H^{x-i}(k_x)$ equals $k_i$. Using the one-way key chain to organize the network access control keys, authentication of later keys based on a previous one can be trivially performed at each node by hashing a newly distributed key a few times and verify if it matches a previously known one. Figure 6 illustrates this approach.


*Figure 6 - Organize network access control keys into one-way key chain*

## 3.3 Packet Retransmission Scheme

In wireless networks, a node may not receive the group key update messages due to the limited transmission range and topology changes. As a result, two legitimate nodes may simultaneously hold different group keys. We refer to such a scenario as key un-synchronization. We call these nodes using old group key as out-of-sync nodes. When a node is out-of-sync with the group key it cannot access the network until the next round of group key update, since all their packets won't have the right message authentication code.

When key un-synchronization happens, a legitimate node may receive some packets authenticated with a different group key than the one it holds. For convenience, we refer to such received packets as unverified packets, since they cannot be immediately verified by the receiver. Simply accepting or forwarding an unverified packet is not acceptable. Otherwise, an attacker can easily defeat the proposed network access control system by using spurious messages.

Considering (a) how a receiver synchronizes the group key with the corresponding sender and (b) whether a receiver buffers an unverified packet, a legitimate node has four options to avoid key un-synchronization or to handle an unverified packet. These options are:

1. Synchronizes the keys with all nodes along the communication path first, to avoid the key un-synchronization in future data transmission
2. Buffers the unverified packets, synchronizes the group key with the sender, and finally verifies these buffered packets;
3. Simply drops the unverified packets in case of key un-synchronization
4. Drops the unverified packets and asks the sender to retransmit the packets, while a key synchronization is triggered during the retransmission.

The first three options each have serious drawbacks. The first option cannot exclude the possibility that a transmission cannot complete because of frequent key updates. The second option exposes legitimate nodes to memory consumption attacks. That is, the victim nodes are enforced to buffer the bogus packets authenticated with a ``newer'' group key, while waiting to receive the necessary group key update message, and it may suffer from DoS attacks. The third option prevents two legitimate nodes from establishing or continuing communication when they are key un-synchronized.

We adopt the fourth option because it can assist the communicating nodes to reestablish communication in the case they become key un-synchronized, constrain the propagation of any unverified packet to a single hop, and avoid the memory consumption attack since the receiver is not required to buffer any unverified packets.

To bring the out-of-sync node back to the network, we exploit the stateless feature of the SDR scheme we are using. Each node buffers the most recent key update message it has received. It can transmit this buffered message to other nodes that have not been updated to the newest group key. These receiving nodes can then extract the new group key from the received message, since a stateless group key update scheme allows a legitimate user to obtain the updated group key as long as the user has the corresponding key update message, even if the user has been offline for a while, or missed several previous rounds of key updates. In this way, two key un-synchronized nodes that want to communicate can synchronize their group keys without relying upon the key manager.

# 4 Conclusion

In this paper, we attempt to solve two challenges of securing wireless networks in control systems. First, we present a polynomial-based weak authentication scheme to mitigate potential DoS attacks against public key protocols. This scheme provides one layer of authentication to mitigate DoS attacks. Second, we propose to use stateless group key update schemes to update group keys in wireless networks. A stateless group key update scheme is an ideal candidate when the communication channel is unreliable. We also present a packet retransmission scheme to update the group key on out-of-sync

nodes, so that those nodes can access the network before the next round of group key update.

---

**About the Author** – Dr. Kun Sun is a senior Research Scientist in Intelligent Automation Inc. at Rockville, Maryland. He leads the research on network and systems security. He has won a number of research awards (1.5 Million in total) from government agencies like DoD, DHS, and NIST. He serves as technical lead for projects including secure network access in industrial control system, security analysis using security metrics, secure database system in ad-hoc networks, trusted query and trusted routing in wireless sensor networks, and DoS mitigation in wireless networks.

## References:

[1] SSL 3.0 specification. http://wp.netscape.com/eng/ssl3/.

[2] S. Kent and R. Atkinson. IP authentication header. IETF RFC 2402, November 1998.

[3] An Liu, Peng Ning, "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks," in Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, pages 245--256, April 2008.

[4] N. Gura, A. Patel, and A.Wander. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), pages 119–132, August 2004.

[5] D. Malan, M. Welsh, and M. Smith. A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In Proceedings of IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON), pages 71–80, 2004.

[6] H. Wang and Q. Li. Efficient Implementation of Public Key Cryptosystems on Mote Sensors. In Proceedings of International Conference on Information and Communication Security (ICICS), Dec. 2006.

[7] Rivest, R.; A. Shamir; L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM 21 (2): 120–126. 1978.

[8] D. Hankerson, A. Menezes, and S.A. Vanstone, Guide to Elliptic Curve Cryptography, Springer-Verlag, 2004.

[9] MICAz motes. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf.

[10] Imote2 motes. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2_Datasheet.pdf

[11] University of Technology Seminar on Network Security, in TML laboratory report series, December 2000.

[12] D. Wallner, E. Harder, and R. Agee. Key Management for Multicast: Issues and Architectures. IETF Request For Comments, RFC2627, 1999.

[13] C. Wong, M. Gouda, and S. Lam. Secure Group Communications Using Key Graphs. In Proceedings of the ACM SIGCOMM '98, pages 68–79, Vancouver, B.C, September 1998.

[14] D. Naor,M. Naor, and J. Lotspiech. Revocation and Tracing Schemes for Stateless Receivers. Advances in Cryptology-CRYPTO'01, LNCS 2139:41–62, 2001.

[15] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Dean. Self-Healing Key Distribution with Revocation. In Proceedings of 2002 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 2002.

[16] Liu, D., & Ning, P. (2003). Establishing Pairwise Keys in Distributed Sensor Networks. Proceedings of ACM Conference on Computer and Communications Security (CCS).

[17] Kun Sun, An Liu, Peng Ning, Rogue Xu and Douglas Maughan. "Securing Network Access in Wireless Sensor Networks", In Proceedings of the Second ACM Conference on Wireless Network Security (WiSec 09), March, Zurich, Switzerland