INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Research in Information Security

Dr. Ashok Kumar Das

IEEE Senior Member
Associate Professor
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

E-mail: *ashok.das@iiit.ac.in*
URL: `http://www.iiit.ac.in/people/faculty/ashokkdas`
`https://sites.google.com/view/iitkgpakdas/`

The key prioritization technique using post deployment knowledge

- By using memory for sensing applications, a sensor node can keep a large number of key units during key predistribution phase.
- After deployment, once a sensor node determines its post-deployment location, it can prioritize these key units based on this post-deployment knowledge.
- A sensor node can then give up the key units that are less likely to be used for pairwise key establishment to thwart against node capture attacks by an adversary and return the memory to the sensing applications.
- Thus, it has a higher probability to keep those key units that may be required for secure communications with its neighbor nodes.
- Hence, this phase reuses the memory for sensing applications to keep more key units during key predistribution phase, keeps the higher priority key units that are most likely to be used after the post-deployment location is known, and finally discards the low priority key units to thwart against node capture attacks and returns the memory to applications.

Key distribution using pre-deployment and post-deployment knowledge: a combined approach

In order to do that the closest pairwise keys scheme (CPKS) can be combined with the key prioritization technique applied under the EG scheme to provide better network connectivity and resilience against node capture compared to those for the existing schemes, called as the **Enhanced CPKS (ECPKS)**.

Key pre-distribution:

This phase is performed by the key setup server in offline before deployment of the sensor nodes in some target field. It consists of the following steps:

- *Step-1:* Let n be the size of the sensor network. For each sensor node u, the key setup server assigns a unique identifier id_u.
- *Step-2:* The key setup server selects a large key pool $\mathcal{K}$ of M key units. Each key unit consists a random number (i.e., a symmetric cryptographic key) generated by the setup server and a unique location (x, y) in the deployment field associated with it.

Combined approach

- For example, a key unit for a sensor node i is of the form $ku_i = (k, (x_i, y_i))$ where k is a randomly generated symmetric key by the (key) setup server and (x_i, y_i) is the location attached with k. For convenience we refer this location (x_i, y_i) as the id of the key k.
- *Step-3:* For each sensor node u to be deployed in the sensor network, the setup server selects a set S_1 of c sensor nodes whose expected locations are closest to sensor node u. For each sensor node v in S_1, for which a pairwise key between u and v has not already been generated, a new random key k_{uv} is generated. The key-plus-id combination (k_{uv}, id_v) is loaded to u's key ring, whereas the pair (k_{uv}, id_u) is loaded to v's key ring.
- *Step-4:* To further improve performances of this scheme, more key materials can be loaded in each sensor node initially. Since the low priority key units after the key prioritization phase will be deleted from memory, so the returned memory will be used for application part by the sensor node. To achieve this goal, for each sensor node u to be deployed in the network, the setup server selects another set S_2 of m key units randomly from the key pool $\mathcal{K}$ and loads this set to u's key ring.

Direct key establishment

In order to establish a secret key between two neighbor nodes, say, u and v, they need to exchange the following messages:

- Node u generates a random nonce RN_u. It then sends its own id id_u and RN_u to its neighbor v.
  $u \rightarrow v : id_u || RN_u$.
- Node v also generates a random nonce RN_v and then sends its own id id_v as well as RN_v to its neighbor u.
  $v \rightarrow u : id_v || RN_v$.
- Assume that the id of v is found from first c key units of u's key ring. Let k_{uv} be the key shared with node v corresponding to that id. It sends its own id id_u, the id id_v of v as well as the random nonce RN_v of v and a message authentication code (MAC) on these fields under the key k_{uv} to node v.
  $u \rightarrow v : T = (id_u || id_v || RN_v) || MAC_{k_{uv}}(T)$.

- In a similar fashion, v verifies the id of u in first c key units of its key ring and assume that v finds a secret key k_{uv} shared with node u corresponding to that id. It then sends its own id id_v, the id id_u of u as well as the random nonce RN_u of u and a message authentication code (MAC) on these fields under the key k_{uv} to node u.
  $v \to u : \{T' = (id_u || id_v || RN_u)\} || MAC_{k_{uv}}(T')$.
- After receiving the last message by the nodes u and v, they perform one symmetric cryptographic MAC verification on that message, under the key k_{uv}. If the MAC verification is successful, then only they store the key k_{uv} for their future communications. We note that by sending the random nonces of each other by the nodes u and v, the transaction between them is determined uniquely.

Key prioritization

This phase uses the last remaining m key units from the key rings of sensor nodes. This phase consists of the following steps:

- *Step-1:* Sensor node u securely determines its post-deployment location, say $PDL_u = (u_x, u_y)$. u computes the distances between PDL_u and the locations attached to these m key units. Then u ranks these m key units in its key ring KR_u in increasing order according to the computed distances. The shorter distance has the higher priority. Node u chooses c' highest priority key units from these key units and deletes the remaining $(m - c')$ key units from its key ring which are less likely to be used for pairwise key establishment in order to thwart against node capture attacks. The returned memory is used for application by the sensor node u.
- *Step-2:* Similarly, a physical neighbor v of sensor node u performs the above step.
- *Step-3:* It is easy to observe that if two nodes are close to each other, there is a high probability to establish a direct key between them. To establish a direct key between them, nodes u and v exchange only the locations (ids) of the c' highest priority key units from their key rings.

Communication steps in key prioritization

- Node u sends its own id, a randomly generated nonce RN_u and a list of c' highest priority key ids to the node v:
  $u \rightarrow v : id_u || RN_u || \{$ list of c' highest priority key ids $\}$.
- Similarly, node v also sends its own id, a randomly generated nonce RN_v and a list of c' highest priority key ids to the node u:
  $v \rightarrow u : id_v || RN_v || \{$ list of c' highest priority key ids $\}$.
- Assume that u finds q common keys, say, $k_1, k_2, \ldots, k_q$ $(q \geq 1)$ from the c' highest priority key units. It computes a secret key shared with node v as $k_{uv} = k_1 \oplus k_2 \oplus \ldots \oplus k_q$ and then sends the following message with a MAC under the computed key k_{uv} to node v:
  $u \rightarrow v : T_1 = (id_u || id_v || RN_v) || MAC_{k_{uv}}(T_1)$.
- Similarly, node v also computes the secret key k_{uv} shared with the node u. It then sends its own identifier id_v, node u's identifier id_u as well as the random nonce RN_u of node u along with a message authentication code (MAC) of these fields under the key k_{uv} to node u:
  $v \rightarrow u : T_2 = (id_u || id_v || RN_u) || MAC_{k_{uv}}(T_2)$.

Network connectivity in combined approach

We use the following notations for deriving the probability of establishing a direct key between two neighbor nodes.

- n : the network size.
- d : the average number of neighbor sensor nodes of each sensor node.
- c : the number of pre-distributed key units given to each node for the direct key establishment phase of this scheme.
- M : the key pool size.
- m : the number of pre-distributed key units given to each node for the key prioritization phase of this scheme.
- ρ : the communication range.
- e : the maximum deployment error.

Network connectivity in combined approach

- E_1 : the event that a sensor node can establish a key during the direct key establishment phase of this scheme with one of its d physical neighbors using only the first c key units from its key ring.
- p_1 : $P(E_1)$, the probability that two neighbors establish a direct key for the event E_1.
- E_2 : the event that a sensor node can establish a direct key during the key prioritization phase of this scheme with one of its d' physical neighbors, where $d' = \lfloor (1 - p_1) \cdot d \rfloor$, using the last m key units from its key ring.
- p_2 : $P(E_2)$, the probability that two neighbors establish a direct key for the event E_2.
- p : the overall probability that two neighbors establish a direct key which is same as the probability that at least one of the events E_1 and E_2 will occur.

Network connectivity in combined approach

- It is easy to observe that the events E_1 and E_2 are stochastically independent.
- Hence, we have $P(E_1 \cap E_2) = P(E_1) \cdot P(E_2)$.
- Now, $P(\bar{E}_1 \cap \bar{E}_2) = 1 - P(E_1 \cup E_2) = 1 - [P(E_1) + P(E_2) - P(E_1 \cap E_2)] = (1 - P(E_1))(1 - P(E_2)) = P(\bar{E}_1)P(\bar{E}_2)$, where $P(\bar{E})$ represents the probability of the complement of the event E.
- Thus, both the events $\bar{E}_1$ and $\bar{E}_2$ are also stochastically independent.
- As a result, the required probability that at least one of the events E_1 and E_2 will occur is $1-$ (probability that none of the events E_1 and E_2 will occur) $= 1 - P(\bar{E}_1 \cap \bar{E}_2)$ $= 1 - (1 - p_1)(1 - p_2) = p_1 + p_2 - p_1 p_2$.
- Therefore, we obtain the overall probability of establishing a direct pairwise key between two neighbor sensor nodes as

$$p = p_1 + p_2 - p_1 p_2. \tag{1}$$

Network connectivity in combined approach

For the analysis of network connectivity, the following parameter values are considered:

- The communication range is $\rho = 30$ meters.
- The number of nodes in the network is $n = 10000$.
- The average number of neighbor nodes for each node is $d \leq 40$.
- The number of pre-distributed keys given for the direct key establishment phase is $c = 200$.
- The key pool size is $M = 40000$.
- The number of pre-distributed keys given for the key prioritization phase is $m = 100$.
- The number of highest priority keys after the key prioritization phase is $c' = 80$.
- The size of the deployment area A is chosen so that the maximum network size becomes $n = \lfloor \frac{A \times (d+1)}{\pi \times \rho^2} \rfloor$.

Network connectivity in combined approach

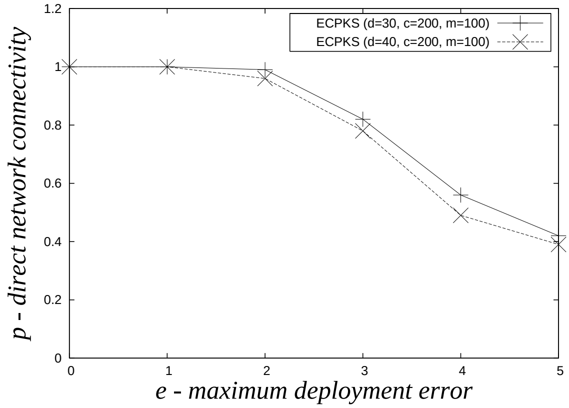

Figure: Direct network connectivity of the combined scheme.

Resilience against node capture attack

- Assume that there are n sensor nodes in the sensor network.
- This network is considered as an undirected graph with n vertexes each having the same degree d, where d is the average number of neighbor nodes of each node.
- The total number of edges of the undirected graph with n vertexes is $\frac{\sum_{i=1}^{n} d}{2} = \frac{n\,d}{2}$.
- Hence, the total direct communication links in the sensor network of size n is $\frac{n\,d}{2}$.
- Let us consider the direct key establishment phase of this combined scheme.
- In this phase, each pairwise key between two neighbor nodes was generated by the key setup server randomly and thus, those keys are different for each pair of neighbor nodes in the network. Therefore, no matter how many nodes are compromised secret communications between non-compromised nodes are still secure.

Resilience against node capture attack

- As a result, we have $\frac{n\,d}{2} \cdot p_1$ secure links so far even after capturing N_c nodes by the adversary.
- Now consider the key prioritization phase of this combined scheme. Since a sensor node keeps only c' highest priority key units after key prioritization phase in order to establish secret keys with its remaining $(1 - p_1)d$ neighbor nodes, $\frac{nd}{2}(1 - p_1)p_2 \times [1 - (1 - \frac{c'}{M})^{N_c}]$ links will be compromised by the adversary from the total $[\frac{nd}{2}p_1 + \frac{nd}{2}(1 - p_1)p_2]$ secure links in the sensor network.
- Hence, the required fraction of compromised secure communication links (i.e., direct keys) between non-compromised sensor nodes can be estimated as

$$\begin{aligned} P_e(N_c) &= \frac{(1 - p_1)p_2}{p_1 + (1 - p_1)p_2} \times \left[1 - \left(1 - \frac{c'}{M}\right)^{N_c}\right] \\ &= \left(1 - \frac{p_1}{p}\right) \times \left[1 - \left(1 - \frac{c'}{M}\right)^{N_c}\right], \end{aligned} \qquad (2)$$

where M is the key pool size.

Resilience against node capture attack

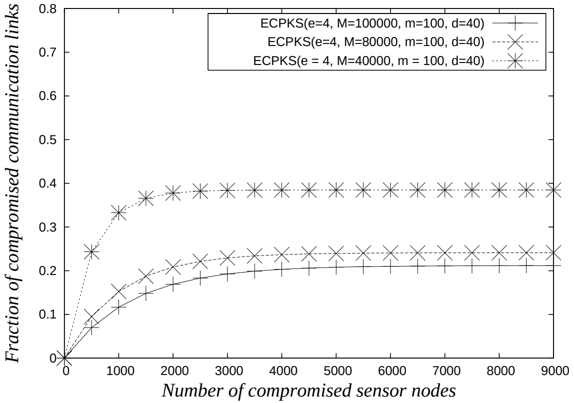

Figure: Resilience against node capture of the combined scheme.

Resilience against node capture attack (Important observations)

- CPKS provides unconditional security, because compromise of any number of nodes do not reveal any secret between non-compromised nodes in the network.
- The security of CPPS shows that it performs well as long as number of captured nodes in the network is small. However, when the number of captured nodes exceeds a certain threshold, CPPS leads to compromise of a large fraction of total secure communication between non-compromised nodes.
- In ECPKS, it follows that even if an adversary captures large number of nodes in the network, it leads to compromise of only a certain fraction of communication links between non-compromised nodes.
- Since the key prioritization phase is applied only for d' neighbors of a sensor node where $d' = \lfloor (1 - p_1) \cdot d \rfloor$, so the security degrades only due to pairwise keys established with those d' nodes.
- In CPPS, if the length L of cell side is chosen larger, CPPS leads to a larger number of sensor nodes sharing the same bivariate polynomial, which in turn degrades the security performance. CPKS is unconditionally secure against node compromise, whereas CPPS is less secure than CPKS. On the other hand, ECPKS provides better security than CPPS.

Performance comparison among ECPKS, CPKS and CPPS

Schemes $\Rightarrow$ *Items* $\Downarrow$	*CPKS*	*CPPS*	*ECPKS*
Storage requirement	c key units	5 polynomial shares	c' key units
Communication overhead	node's own id	node's home cell id	node's own id + highest priority key ids
Computational overhead	$2T_{MAC}$	evaluation of a t-degree univariate polynomial + $2T_{MAC}$	$2T_{MAC}$ (in direct key establishment) + $2T_{MAC}$ (in key prioritization)
Network connectivity	poorer if e is larger	poorer if e is larger	significantly better even if e is larger
Resilience against node capture	unconditionally secure	unconditionally secure and t collusion resistant	better than CPPS

Important References

- Donggang Liu, Peng Ning, “Improving key predistribution with deployment knowledge in static sensor networks,” ACM Transactions on Sensor Networks (TOSN), vol. 1, no. 2, pp. 204-239, 2005.
- Ashok Kumar Das, “ECPKS: An Improved Location-Aware Key Management Scheme in Static Sensor Networks,” International Journal of Network Security, vol. 7, no. 3, pp. 358-369, 2008.