Preventing Advanced Persistent Threats in Complex Control Networks

Juan E. Rubio, Cristina Alcaraz and Javier Lopez

Department of Computer Science, University of Malaga,
Campus de Teatinos s/n, 29071,Malaga, Spain
{rubio,alcaraz,jlm}@lcc.uma.es

Abstract. An Advanced Persistent Threat (APT) is an emerging attack against Industrial Control and Automation Systems, that is executed over a long period of time and is difficult to detect. In this context, graph theory can be applied to model the interaction among nodes and the complex attacks affecting them, as well as to design recovery techniques that ensure the survivability of the network. Accordingly, we leverage a decision model to study how a set of hierarchically selected nodes can collaborate to detect an APT within the network, concerning the presence of changes in its topology. Moreover, we implement a response service based on redundant links that dynamically uses a secret sharing scheme and applies a flexible routing protocol depending on the severity of the attack. The ultimate goal is twofold: ensuring the reachability between nodes despite the changes and preventing the path followed by messages from being discovered.

Keywords: advanced, persistent, threat, attack, detection, response, consensus, opinion, dynamics, secret, sharing, redundant, topology

1 Introduction

The interconnection of industrial environments with modern ICT technologies has increased the number of internal and external threats in this context, including those from traditional IT systems (e.g., malware, spyware, and botnets). Among these, the Advanced Persistent Threats (APT) are a new class of sophisticated attacks that are executed by well-resourced adversaries over a long period of time. They usually go undetected because they leverage zero-day vulnerabilities and stealthy and evasive techniques [1]. While APTs originally attacked military organizations, they are now targeting a wide range of industries and governments with multiple purposes: economic (espionage, intellectual property), technical (access to source code), military (revealing information) or political (destabilization of a company). Their goal is to get through the organization's network and take over the industrial control systems.

Stuxnet was the first attack of this kind, reported in 2010, which sabotaged the Iranian Nuclear Program by causing physical damage to the infrastructure and thereby slowed down the overall process. Ever since, the number of reported vulnerabilities concerning Industrial Control Systems has been dramatically increasing, as the research community has become more involved and new attacks have been revealed. Like Stuxnet, all APTs are tailored to the specific victim's network topology, and they count on a defined succession of steps: firstly, the attacker intrudes on the network by using social engineering (e.g., by means of fraudulent e-mails containing trojans); secondly, they install a backdoor from which the attackers connect to the target network. Then, several exploits and malware are used to compromise as many computers in the victim network as possible, to ultimately modify the productive process or exfiltrate information back to the attacker domain.

On the whole, an APT is a meticulously planned attack adapted to the target infrastructure, one whose complexity makes the use of traditional countermeasures (e.g., antivirus, firewalls) insufficient to tackle them. An additional effort is required to mitigate their effects, by involving the organization in security awareness training and introducing novel services in continuous evolution within the company [2]. For this reason, we propose the design of practical mechanisms to firstly detect and then effectively respond to these attacks, applied to a common network representation. We can summarize our contributions as:

- Modeling the evolution of an APT within the victim network topology.
- Implementation of a multi-agent system for the detection of an APT based on the topological changes suffered in selected parts of the network, observed by hierarchically chosen nodes in accordance with controllability criteria.
- Use of redundancy edges and random routing protocols to overcome the network deformation provoked by the APT and to avoid compromised systems, ensuring the reachability between nodes and the survivability of the network.

The remainder of this paper is organized as follows: Section 2 outlines preliminary concepts about dynamic control networks and describes the threat model used for the APT. In Section 3 the detection of these attacks is addressed by means of a network decision model. Based on this mechanism, response techniques are implemented in Section 4, which are theoretically and experimentally analyzed in Section 5.

2 Preliminaries

2.1 Structural Controllability

Considering the cost of the implementation of large control networks from a research point of view, it becomes mandatory to model and simulate the problem through graph theory, taking into account the network topology and the nature of its distribution. Specifically, we focus on topologies of the type power-law $y \propto x^{-\alpha}$ [3], since the vast majority of critical control systems follow these structures, which produce small sub-networks similar to current control substations.

With the purpose of helping the reader understand the underlying theoretical concepts of our model, topics related to structural controllability and power dominance are described here. The concept of structural controllability was introduced by Lin in 1974 [4], which associates the control to a subset of nodes with the maximum capacity of dominance.

Let $G = (V, E)$ be a *directed* graph that represents the network topology, given by its adjacency matrix, that is, a square binary matrix M with dimension $|V|$ where $M(i, j) = 1$ whenever $(v_i, v_j) \in E$ and zero otherwise. Through $G(V, E)$, it is possible to characterize dynamic control networks including loops and weighted edges that represent the interconnection of control devices with remote terminal units (e.g., sensors or actuators). These links contain the maximum capacity to conduct the main traffic between two points, which is defined as the *control load capacity* (CLC).

To represent this traffic, we use the edge betweeness centrality (EBC) [5]. It is an indicator that represents the sum of the fraction of the shortest paths that pass through a given edge, so that edges with the highest centrality participate in a large number of shortest paths. The result is a weighted matrix related to $G_w(V, E)$ whose weights are computed as follows:

$$E_{BC} = \sum_{s,t \in V} \frac{\delta(s, t|e)}{\delta(s, t)} \tag{1}$$

where $\delta(s, t)$ denotes the number of shortest (s,t)-paths and $\delta(s, t|e)$ the number of paths passing through the edge e. On the other hand, let the in-neighborhood N_i^{in} of a node i be the set of nodes v_j such that $(v_j, v_i) \in E$, while the out-neighborhood N_i^{out} is the set of nodes v_j such that $(v_i, v_j) \in E$. Consequently, let the in-degree $d_i^i n$ of a node v_i be the number of its incoming edges, i.e., $d_i^{in} = |N_i^{in}|$, while the out-degree d_i^{out} is the sum of its outgoing edges, i.e., $d_i = |N_i^{out}|$.

Taking these concepts and EBC into account, the Dominating Set (DS) of a graph G can be defined as the minimum subset of nodes $D \subseteq V$ such that for each vertex $v_i \notin D$ is adjacent to at least one member of D, that is $\exists v_k \in D|(v_k, v_i) \in E$. These nodes D with highest control capacity will be those with the highest edge betweeness centrality $E_{BC}(v)$ for all their outgoing edges. The creation of this set is explained in Algorithm 1. Related to this concept, the Power Dominating Set (PDS) consists in an extension of the DS by including new *driver nodes* (denoted by N_D), those with the maximum capacity of dominance. The original formulation of this set was given by Haynes *et al.* in [6], and was later simplified into two fundamental observation rules by Kneis *et al.* in [7]:

OR1 *A vertex in N_D observes itself and all its neighbors, complying with DS.*
OR2 *If an observed vertex v of degree $d^+ \geq 2$ is adjacent to $d1$ observed vertices, the remaining un-observed vertex becomes observed as well. This also implies that **OR1** $\subseteq$ **OR2** given that the subset of nodes that comply with **OR1** becomes part of the set of nodes that complies with **OR2**.*

For our purpose in this paper, the dominating nodes play the role of agents that detect topological changes in their surroundings that may be derived from an APT attack, and establish backup links that ensure the continuity of the network.

Algorithm 1 DS($G(V, E)$)

```
output ($DS = \{v_i, ..., v_k\}$ where $0 \leq i \leq |V|$)
local: $BC(V)$ representing betweeness centrality of $V$

Choose $v \in V$ with highest BC
$DS \leftarrow \{v\}$ and $N(DS) \leftarrow \{v_i, ..., v_k\}$ $\forall i \leq j \leq k \setminus (v, v_j) \in E$
while $V - (DS \cup N(DS)) \neq \emptyset$ do
    Choose vertex $w \in V - (DS \cup N(DS))$ with highest BC
    $DS \leftarrow DS \cup \{w\}$
    $N(DS) \leftarrow N(DS) \bigcup \{v_i, ..., v_k\}$ where $\forall i \leq j \leq k \setminus (w, v_j) \in E$
end while
```

2.2 Threat model: Representation of APT Attacks

Assuming a successful intrusion inside a network represented by a matrix M, we model an APT with a succession of attacks perpetrated on its topology. Specifically, just as an actual APT works, the attacker firstly selects one node and then makes several lateral movements in order to find new nodes to compromise. Since we want to provide realism in this model and consider a scenario of high criticality, we assume the attacker always seeks those nodes with more controllability, that is, those belonging to the DS and hence the ones with the highest betweeness centrality.

In each of the steps in its life cycle, the APT can commit individual attacks on the topology, i.e. changing the edges from the compromised node at a given time instant. This consequently generates a new matrix M'. The types of attacks can be:

* **Removal of an incoming edge:** given the vertex v_i that represents the compromised node such that v_j exists and $M(j, i) = 1$, it implies setting $M(j, i) = 0$.
* **Removal of an outgoing edge:** given the vertex v_i that represents the compromised node such that v_j exists and $M(i, j) = 1$, it implies setting $M'(i, j) = 0$.
* **Addition of an incoming edge:** given the vertex v_i that represents the compromised node such that v_j exists and $M(i, j) = 0$, it implies setting $M'(i, j) = 1$.
* **Addition of an outgoing edge:** given the vertex v_i that represents the compromised node such that v_j exists and $M(i, j) = 0$, it implies setting $M'(i, j) = 1$.

In a simple version of the APT, we suppose that the kind of the attack and the first node compromised within the network are chosen randomly. From that moment on, the attack migrates to the adjacent node with highest betweeness centrality, simulating the fact that the attacker can perform a reconnaissance of the network when looking for potential victims that deal with higher loads of control traffic. The resulting attacker behavior is described in Algorithm 2. An example of an APT with three attacks over a defined network topology is depicted in Fig. 1, where driver nodes are marked in black to show how the APT always migrates to vertices with higher controllability. Firstly, node 4 is selected and an outgoing edge is added towards node 2. Then, the attacker moves to node 6 and removes the edge coming from node 3 and then, since node 6 still has dominance, the attack stays there and removes the edge going to 7.

Algorithm 2 Advanced persistent threat life cycle

```
output: $M'$ representing the resulting matrix
local: $M$ representing $G(V,E), numOfAttacks$
$attackedNode \leftarrow random\ v_i \in E$
$M' \leftarrow M$

for i:=1 to $numOfAttacks$ step 1 do
    $attack \leftarrow randomAttack\ over\ attackedNode\ (edge\ addition\ or\ removal)$
    $update\ M'\ based\ on\ attack$
    $attackedNode \leftarrow$ SelectNewAttackedNode($M, attackedNode$)
    if $attackedNode == null$ then
        $attackedNode \leftarrow random\ v_i \in E$
    end if
end for

function SelectNewAttackedNode($M$,$node$)
    $childNodes \leftarrow vertexes\ v_j | M(node, v_j) = 1$
    $parentNodes \leftarrow vertexes\ v_k | M(v_k, node) = 1$
    $candidates \leftarrow childNodes \cup parentNodes$
    $maxCentrality := 0$
    $attackedNode \leftarrow null$
    for vertex v in candidates do
        $centrality \leftarrow$ CalculateBetweenessCentrality($v$)
        if $centrality > maxCentrality$ then
            $attackedNode \leftarrow v$
        end if
    end for
    return $attackedNode$
end function
```

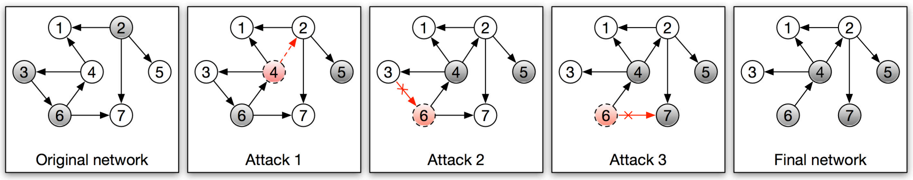


Figure 1. Example of APT with 3 attacks: 1st: Addition of edge from node 4 to node 2. 2nd: Removal of edge from node 3 to node 6. 3rd: Removal of edge from node 6 to node 7.

3 APT Detection through Opinion Dynamics

Now we have modeled the effect of an APT over the network topology, in this section we describe a feasible method to allow the network to locate subtle changes in certain parts, making it easier to accurately deploy response techniques to overcome the effect of one of these threats.

We start with the notion introduced before: let us suppose a *power law* distribution network defined by the *directed* graph $G = (V, E)$ and represented by the adjacency matrix M. Let us also suppose the presence of n agents deployed over that network (the DS nodes of the graph G), so each node $v_i \in V$ is connected to one or more agents. We further assume that the agents can also communicate with each other according to a communication network represented by the directed graph $G_c = (V, E_c)$, where for all $(v_i, v_j) \in E_c$, we have that $(v_i, v_j) \in E$. Our goal is to put into practice a distributed cooperative algorithm among these agents to detect precise topology attacks in their neighborhood by exchanging information on changes produced in their observable nodes. In this regard, various decision models can be imported from graph theory, among which we can highlight *consensus* and *opinion dynamics*.

In the consensus approach, a collection of agents cooperate to reach a common objective by sharing information about their state and other environmental conditions [8]. Such negotiation depends on the network topology, so it can be leveraged to collectively build a global indicator of the entire network health at a given moment. Compared to this algorithm, opinion dynamics proposes a model that admits the fragmentation of patterns, so the aforementioned agents may differ in their opinions during the negotiation process [9]. These network partitions will depend on the closeness to the opinion of each node, which is calculated based on the number of topological changes they detect. Therefore, it makes it easier to identify which areas of the network are more affected by the action of the APT and to what extent.

Opinion dynamics originally models the influence among individuals in a group or the entire society, where there is a wide spectrum of opinions. Each agent crafts its own opinion taking into consideration the ones from the rest of agents to a certain extent. Eventually, the opinions are distributed into several clusters. For our purpose, it implies fragmenting the network according to the multiple changes that could occur in separate areas, whose individual consensus value raises an indicator of the severity of the attacks over that particular portion of the topology.

Let us suppose that $x_i(t)$ represents the opinion of a fixed agent i at time t. The vector $x(t) = (x_1(t), ..., x_n(t))$ represents the opinion profile at time t for all the agents. Given an agent i, the weight given to the opinion of any other agent j is denoted by a_{ij}. For simplicity, we consider a_{ii} such that $\sum_{k=1}^{n} a_{ik} = 1$. Therefore, agent i also takes into account its own opinion during the opinion formation process, which can be described as follows:

$$x_i(t+1) = a_{i1}x_1(t) + a_{i2}x_2(t) + ... + a_{in}x_n(t)$$

In a matrix notation it can be written as:

$$x(t+1) = A(t, x(t))x(t)$$

where the matrix $A(t, x(t)) = [a_{ij}]$ is the square matrix that collects the weights, which summarize the relationships between the agents' opinions. These weights can change over time or by opinion, so finally an agent i adjusts its opinion in period $t+1$ by taking a weighted average of the opinion of agent j at time t. When t tends to infinity, the final behavior of the opinion profile may lead to a consensus among all or part of the agents, which can also be visualized graphically.

Returning to our domain, we execute this algorithm assuming that $x_i(0)$ will be calculated for each agent i as follows: let us suppose that $BC(v_i)$ represents the original *betweeness centrality* for each agent i that, as explained, works as an indicator of the controllability of that particular node. If $BC'(v_i)$ is the *betweeness centrality* of the same agent after being victim of a particular attack of those defined in Section 2.2 or another node in its neighborhood, we define the initial opinion $x_i(0)$ as

$$x_i(0) = \frac{|BC'(v_i) - BC(v_i)|}{BC(v_i)} \tag{2}$$

Consequently, $x_i(0)$ holds the ratio of change in the controllability of an agent i after an attack, compared to its initial state (due to an increase or decrease of adjacent edges). We assume that when the value was originally zero or the resulting ratio is greater than 1, the result is normalized to the value of 1.

Altogether, if we have the vector $x(0)$ concerning the initial opinion of all agents in the DS, we can run the opinion dynamics algorithm to obtain a value of the change ratio of the network after suffering an individual attack, making it possible to distinguish between different clusters of agents with similar opinion. In this case, the closeness among opinions, which is represented by the matrix A with the weights assigned for each agent, has been modeled according to the difference in the degree of change (the individual opinion each agent holds): for two given agents i and j, if the difference is below a determined epsilon value (e.g., 0.3), they increase the weight given to each other; this models the fact that agents that experiment a similar degree of change in their surrounding topology must agree on the presence of an anomaly in their respective area.

Figure 2 shows the opinion dynamics algorithm for a network of 30 nodes and 17 agents after suffering an APT comprising 10 attacks. The lines represent the evolution in the opinions for each agent, so finally there is multiple consensus between them: in particular, there are only two agents that indicate relatively large changes (more than 0.5 of fluctuation in their betweeness centrality). However, four agents agree on a change of about 0.25 points around their zone of influence, and many of them indicate a fault of approximately 0.1 in the zone governed by these nodes. As can be seen in the figure, a α value has been added to the plot, which holds the ratio of agents that find a consensus on the amount of degree experienced. This value, together with the opinion about the changes in the topology, serves as the criticality indicator that regulates how strong the response technique must be to mitigate the effects of the APT.

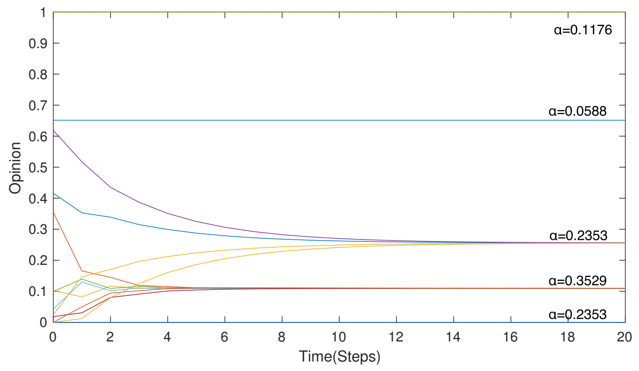


Figure 2. Calculus of the opinion dynamics for a set of agents

4 APT Response

Once we have obtained a measure of the extent to which the network topology is at risk due to the effect of an APT attack, we are in a position to adopt multiple response techniques. We set the goal of preserving the connectivity for all those nodes in charge of delivering control signals to the rest of nodes of the network. According to the different change ratios raised by the opinion dynamics algorithms, we can apply different techniques in separate nodes of the network.

Specifically, we suppose a scenario where we wish to ensure that one node i belonging to the DS wants to send control messages to another node j in the network. This is done in the presence of an APT that can remove certain edges that originally enabled both nodes to communicate over a defined path, traversing other points of the topology [10]. At the same time, we want to avoid hopping over compromised nodes that may be victims of the APT and hence intercept these sensitive packets, preserving confidentiality by this means. Moreover, it is desirable that the communication pattern (i.e., the paths described by the messages when being transmitted over the network) is as random as possible, so as to guarantee that the attacker cannot easily determine the topology of the network. As a result, we have a security service that ensures the continuity of the network until the APT has been successfully removed from the system. To sum up, we seek these three objectives when designing a response technique:

(a) Ensure the presence of a path between node i and j when possible.
(b) Define a routing protocol that prevents determining the path.
(c) Introduce a mechanism to avoid the interception of messages.

To satisfy objective (a), we propose building an edge-redundant network with hidden edges that are added to the original network topology, so these auxiliary links can be leveraged in the event a path between two given nodes is lost after an APT attack. To accomplish this, we create a parallel network from $G = (V, E)$, which we name $G' = (V, E')$, where E' contains the same edges as E and includes new ones from the DS nodes to recover the controllability of the network. Specifically, we define and compare three different strategies:

- **STG1:** addition of redundant edges to all nodes in the network.
- **STG2:** addition of redundant edges only to DS nodes.
- **STG3:** addition of redundant edges only to nodes that are not included in the DS.

Our aim is to compare their level of response in terms of message loss and the overhead they experience, as described in Section 5. Algorithm 3 describes the procedure by which redundancy is added depending on the strategy selected: for each vertex, a set of candidates is created that includes the DS and excludes its parents and the node itself. In the case it is empty, we simply select the DS with maximum out-degree as the new parent of the aforementioned vertex, creating a new edge by this means. It is important to note that during the process, it is ensured that the resulting network $G' = (V, E')$ fulfills OR1 and OR2 conditions, as stated in [11].

On the other hand, to address objectives (b) and (c), we leverage a secret sharing scheme [12]: a secret (i.e., control message) is divided into n shares that are distributed among the sender's neighborhood nodes and follow independent routes, so that the recipient cannot reconstruct the message until it collects, at least, a defined number k of them, where $1 \leq k \leq n$. In the case we have $k = 1$, it can be considered as the basic level of security, as the message in clear is sent over a determined path over the network. If we have $k = n$, then the recipient must collect all the shares to reconstruct the original message. At this point, since our aim is to provide a security mechanism that bases its robustness on the criticality of the attack detected, the election of n will depend on the number of DS agents whose opinion is similar, for which we make use of the α value defined in Section 3. Namely, the maximum number of shares to divide the original message into depends on the ratio of agents that have experienced the same severity in the attacks against their surrounding nodes: the greater the number of DS that experience the same criticality, the greater the number of shares. However, the k value can be random (ranging from 1 to n) in order to make the recovery method as stochastic as possible and thereby not leak any information about the topology when analyzing the stream of messages. The resulting methodology, to divide the messages into shares and send them over the network when it has been attacked, and opinion dynamics has been executed, is described in Algorithm 4. It is important to note that the respective shares are arbitrarily sent over the original and redundant links, in order to make the protocol as misleading for the attacker as possible. Figure 3 shows how shares are divided and distributed over the network leveraging a pathfinding algorithm

Algorithm 3 hiddenTopology$(G(V,E), DS, STG_x)$

```
output $(G' = (V, E'))$
local: $D_r \leftarrow \emptyset$, $E' \leftarrow E$

if $STG_x = 1$ then
    $D_r \leftarrow V$
else if $STG_x = 2$ then
    $D_r \leftarrow DS$
else if $STG_x = 3$ then
    $D_r \leftarrow V - DS$
end if

for vertex v in $D_r$ do
    $F \leftarrow Fathers^a(G(V,E), v)$
    $D_c \leftarrow DS - (F \cup v)$
    $Candidates \leftarrow \emptyset$
    for vertex $c$ in $D_c$ do
        $D \leftarrow Children^b(G(V,E), c) \cap DS$
        $O \leftarrow Children(G(V,E), c) - D$
        comment: checking of OR1 and OR2 fulfillment
        if $v \in DS$ and $((|O| \geq 2$ and $|D| \geq 0)$ or $(|O| = 0$ and $|D| \geq 1)$ or $(|D| = 0$ and $|O| = 0))$ then
            $Candidates \leftarrow Candidates \cup c$
        else if $v \notin DS$ and $((|D| \geq 0$ and $|O| \geq 1)$ or $(|D| = 0$ and $|O| = 0)$ then
            $Candidates \leftarrow Candidates \cup c$
        end if
    end for
    if $Candidates = \emptyset$ then
        $Candidates \leftarrow MaxOutDegree^c(G(V,E), DS)$
    end if
    Arbitrarily select vertex $c_1 \in C$
    $E' \leftarrow E' \cup (c_1, v)$
end for
```

[a] Selection of fathers of v, those belonging to its in-neighborhood
[b] Selection of sons of c, those belonging to its out-neighborhood
[c] Selection of the DS node with maximum out-degree

(e.g., *Dijkstra*, *Breadth-first search (BFS)*) [13][14]. In that example, the secret is divided into three shares with $k = 2$.

5 Theoretical and Practical Analysis

So far in the paper, we have modeled the behavior of an APT against a control network represented with a graph, over which we have applied structural controllability concepts to define a dominance set of nodes. These take the role of agents that make a distributed decision algorithm determine the health of the network based on topological changes detected in their neighbourhood. From this information, they can leverage a parallel hidden topology with redundant links, over which they can continue to deliver their messages with enhanced privacy in the presence of the APT.

In this section, we demonstrate the correctness of this mechanism, as well as offer experimental results to show how effective it is when ensuring the continuity of the network.

Algorithm 4 SecretSharing($G(V, E')$)

```
local: $M$ representing the set of messages to be sent.
for $message\ m$ in $M$ do
    $agent \leftarrow GetRecipient(m)$
    $alpha \leftarrow GetAlpha(agent)$
    $n \leftarrow alpha * |N^{out}_{agent}|$
    $k \leftarrow generate\ random\ from\ 1\ to\ n$
    $S \leftarrow divideSecret(m, n, k)$
    $send\ shares\ to\ n\ neighbours$
end for
```

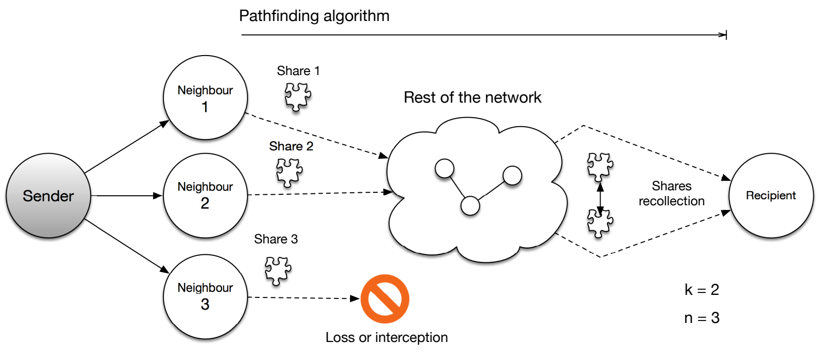


Figure 3. Secret sharing scheme and shares delivery

5.1 Theoretical Analysis

The correctness proof of the message recovery problem is solved when the following requirements are satisfied: (1) the ratio of lost messages when facing an APT attack decreases when using the redundant topology; (2) the algorithm that crafts the set of redundant edges and sends the messages along the network is able to properly finish in a finite time (termination).

We can show the termination of the algorithm through induction, where we first define the initial and final conditions, and the base cases.

Precondition We assume that the network described by $G(V, E)$ is threatened by one or more attacks, probably causing the removal of available routes from the sender to the destination. In other words, there exists a share s belonging to a message m from sender v_1 whose recipient is v_r for which it is not possible to find a sequence of vertices $v_1, v_2, ..., v_r$ such that $(v_i, v_{i+1}) \in E$, $\forall i \in 1, .., r$.

Postcondition given the aforementioned message share and redundant network $G'(V, E')$, there exists a sequence of vertices $v_1, v_2, ..., v_r$ such that $(v_i, v_{i+1}) \in E'$, $\forall i \in 1, .., r$. The availability of additional edges in E is subject to the redundancy strategy selected. Either way, the new route is located by a pathfinding algorithm like *BFS* or *Dijkstra*.

Case 1 We have a message m that is divided into n shares, such that $m = \{s_1, s_2, ..., s_n\}$. In the first step, share s_1 is sent to vertex v_2 through $(v_1, v_2) \in E$.

Case 2 In an intermediate step of the path from sender to destination, the share s_1 traverses the node v_l, and the pathfinding algorithm is evaluated to check the availability of a route. According to Algorithm 4, three scenarios can be distinguished in this point:

-Recovery solution: it takes place when the destination is reachable only through the redundant topology G', that is, there exists a route $v_l, v_{l+1}, ..., v_r$ where $(v_l, v_{l+1}) \in E'$ but $(v_l, v_{l+1}) \notin E$.

-Privacy solution: it occurs when multiple routes are available to reach the recipient of the share, using either the original or the redundant topology. Namely, there exists, at least, a route $v_l, v_m, ..., v_r$ where $(v_l, v_m) \in E$ and another one $v_l, v_{l+1}, ..., v_r$ such that $(v_l, v_{l+1}) \in E'$ and $(v_l, v_{l+1}) \notin E$. In this case, the share hops to v_m or v_l arbitrarily, with the aim of making the route as confusing as possible, thereby dodging potentially compromised nodes over which the attacker expects the traffic to flow. At this point, note that the network may experience some delays when delivering such shares (due to extra hops to reach the recipient), which could be the subject of further research. However, since we are considering a critical scenario, we prioritize availability rather than performance.

-Share loss: in the worst-case scenario, the redundant edges are not sufficient to find a path from v_l to v_r and the original path is no longer available due to the APT. In these circumstances, the share is lost and the algorithm terminates. Note, however, that the secret sharing scheme is resistant to share losses with a given threshold, so the rest of shares s_i with $i \neq 1$ can still rebuild the message m. This depends on the n and k parameters: specifically, the message m successfully reaches its recipient with a probability $\frac{k}{n}$. In this regard, we must stress that the choice of n is based on the severity indicator α, as explained in Section 4.

Induction finally, after a finite number of steps where the different subcases of Case 2 have been applied (except for a secret loss), the node v_j before the last in the sequence holds the share s_1 and there exists an edge (v_j, v_r). The share is finally delivered and Algorithm 4 terminates, satisfying the postcondition of saving a portion of the messages from getting lost, ensuring the validity of our algorithm.

We can also give a brief analysis of the computational complexity of the response algorithm, which must be performed in two ways: for the secret sharing scheme and for the subsequent delivery of shares over the network by using a pathfinding algorithm. As for the former, processing a given message takes n steps, as many as the number of shares it has been split into (determined by α), having $O(n)$ complexity. With respect to the communication mechanism, the complexity must firstly consider the overhead invested by the pathfinding method, which in the case of BFS is $O(n + e)$, where $n \approx |V|$ and $e \approx |E|$. Secondly, it also implies the complexity associated with the share delivery along the graph. Considering the worst-case scenario of the longest route, such a transmission has a cost of $O(n-1+e)$, since the share has to traverse all edges and every node in the network but the sender.

5.2 Experimental Analysis

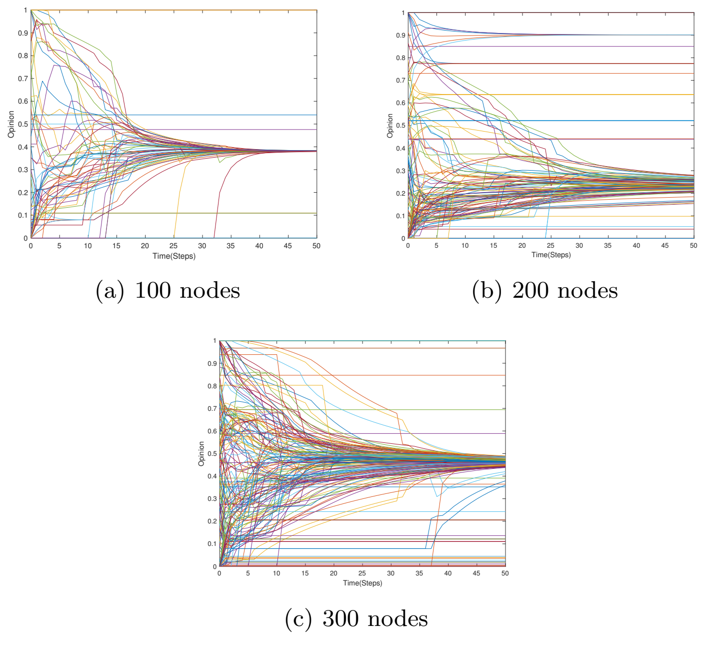
After successfully designing mechanisms to firstly detect topological changes by using distributed opinion algorithms and consequently deploying a response technique to ensure the continuity and preserve privacy in the network, our aim is to test these services in practice. We have conducted the implementation in MATLAB of an APT that follows the behavior described in Algorithm 2. After each attack of the sequence, opinion dynamics is executed on those agents belonging to the DS, which is calculated based on Algorithm 1. If we run different test cases, we can check how the opinion of agents evolve to reach a consensus with each other and form different clusters within the network. Figure 4 shows how the total number of DSs of three different networks (of 100, 200 and 300 nodes) is divided into substantial sets depending on the degree of change they experience after suffering a battery of 50 attacks. It is especially significant to note the presence of a big cluster in each of the three test cases, which indicates an important effect of the APT (of approximately 0.35, 0.25 and 0.45 ratio of change).

Opinion dynamics influences the α value that regulates the number of shares in which the secret messages are divided and distributed from each DS node to the rest of the network to their destination, as explained in Section 4.

To probe the effectiveness of our response technique that leverages a hidden topology comprising additional edges, we have generated a set of 100 messages whose sender belongs to the DS and the recipient is any other node within the network. Following the secret sharing scheme of Algorithm 4, each agent divides the message and gives each part to the corresponding neighbors, which are responsible for the delivery by leveraging the BFS algorithm. The path is calculated at each hop when traversing all nodes until the destination, since the topology can change over time, caused by the APT. In the event that the recipient is unreachable from a certain node at a given time, we consider that share to be lost. Consequently, taking into consideration the scheme, we deem a message to be lost if a number of its shares greater than k have been lost, since it is no longer possible for the recipient to construct the message.

Figure 5 shows the ratio of errors (i.e., message loss due to the unavailability of control paths) when using the normal and the hidden topology networks of STG1, STG2, and STG3. In more detail, we have run three test cases with a network of 100, 200 and 300 nodes, against which we perpetrate an APT of 50 attacks. Prior to executing it, we craft a set of 100 random messages for which we ensure the availability of paths from the sender's neighbors to the recipient. From that point on, the attacks take place and we try to send the original messages after each one. As a result, we can check how the loss ratio fluctuates as attacks occur. In this sense, based on the plots, the original network presents a higher quota of lost messages, whereas applying STG1 (i.e., a redundant edge for all nodes) experiments the lowest ratio, as expected in principle. However, we can see how redundancy in DS nodes (STG2) also achieves an acceptable degree of message reachability, comparable with STG1 and even better at certain points when running the same experiment in different topologies, as Table 1 indicates. This can be explained by the fact that attacks always move to nodes that deal with more traffic and hence have higher controllability (i.e. the DS nodes), as described in Section 2. Therefore, the addition of extra links to recover the connectivity between DSs results in a robust response that, on the other hand, does not introduce too much overhead because of the lower number of additional edges added.

The supremacy and higher connectivity of STG2 and especially STG1 are visible when analyzing the network global efficiency [15]. This measure indicates the efficiency of the information exchange in the network and how resistant it is to failures. If the distance $d(i, j)$ between any two vertices i and j in the graph is defined as the number of edges in the shortest path between i and j such that $i \neq j$, the efficiency is expressed as $1/d(i, j)$. From this definition, the global efficiency of a graph is the average efficiency over all $i \neq j$. Figure 6 shows the evolution in this indicator when performing an APT attack over the original and redundant topologies, for the three test cases of 100, 200 and 300 nodes.

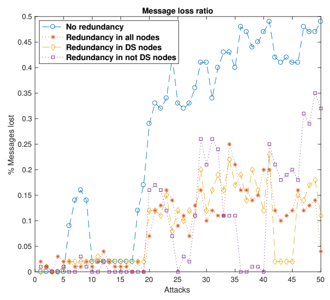


(a) 100 nodes

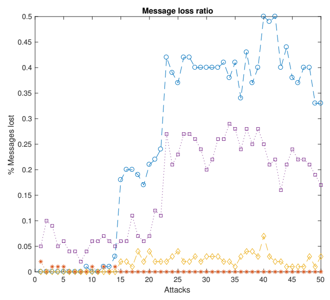
(b) 200 nodes

(c) 300 nodes

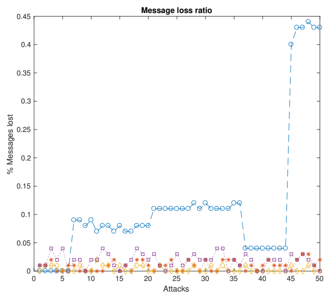
Figure 4. Opinion dynamics after 50 attacks



(a) 100 nodes



(b) 200 nodes

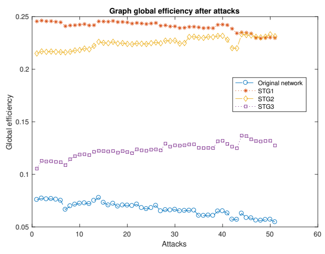


(c) 300 nodes

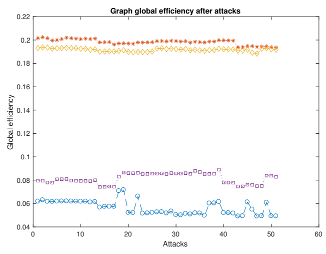
Figure 5. Message loss ratio with the different strategies, 100 messages and 50 attacks over a network of 100, 200 and 300 nodes

Table 1. Message loss ratio after 50 attacks, 100 messages and multiple topologies

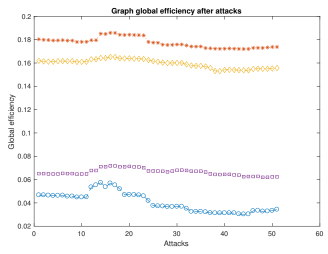
Nodes \Strategy	Original network	STG1	STG2	STG3
10	0.81	0.64	0.64	0.58
50	0.076	0.25	0.3	0.36
100	0.62	0	0.03	0.01
200	0.24	0.04	0.02	0.14
300	0.71	0.02	0	0.21
400	0.07	0.04	0.02	0.03
500	0.39	0.1	0.07	0.19
600	0.4	0.05	0.02	0.03
700	0.32	0.03	0.07	0.1



(a) 100 nodes



(b) 200 nodes



(c) 300 nodes

Figure 6. Global efficiency with different strategies after 50 attacks

6 Conclusions

APT attacks must be addressed with innovative techniques that supplement traditional detection and response techniques. As for the former, we have put into practice a dynamic decision mechanism by making use of graph theory and structural controllability concepts and defining a framework of attacks. That allows us to accurately identify topological anomalies in the network with different degrees of criticality. Accordingly, we have proposed the design of a redundant topology that allows the continuity of the network and also preserves privacy by making the routing process as uncertain as possible for an external attacker. Future work will involve the creation of a richer taxonomy of attacks that not only focus on topological changes, but also on the stealthy compromise of selected nodes within the network. The integration of other distributed decision mechanisms will be also studied, together with restoration and recovery services that ensure a better resilience against APT attacks.

Acknowledgements

The first author is supported by the Spanish Ministry of Education through the National F.P.U. Program under Grant Agreement No. FPU15/03213. In addition, this work has been partially supported by the the Andalusian Government Research Program through the FISICCO project (P11-TIC-07223) and by the Spanish Ministry of Economy and Competitiveness through the PRECISE project (TIN2014-54427-JIN).

References

1. Ping Chen, Lieven Desmet, and Christophe Huygens. A study on advanced persistent threats. In *IFIP International Conference on Communications and Multimedia Security*, pages 63–72. Springer, 2014.
2. Nikos Virvilis, Dimitris Gritzalis, and Theodoros Apostolopoulos. Trusted computing vs. advanced persistent threats: Can a defender win this game? In *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)*, pages 396–403. IEEE, 2013.
3. Giuliano Andrea Pagani and Marco Aiello. The power grid as a complex network: a survey. *Physica A: Statistical Mechanics and its Applications*, 392(11):2688–2700, 2013.
4. Ching-Tai Lin. Structural controllability. *IEEE Transactions on Automatic Control*, 19(3):201–208, 1974.
5. Sen Nie, Xuwen Wang, Haifeng Zhang, Qilang Li, and Binghong Wang. Robustness of controllability for networks based on edge-attack. *PloS one*, 9(2):e89066, 2014.
6. Teresa W Haynes, Sandra M Hedetniemi, Stephen T Hedetniemi, and Michael A Henning. Domination in graphs applied to electric power networks. *SIAM Journal on Discrete Mathematics*, 15(4):519–529, 2002.
7. Joachim Kneis, Daniel Mölle, Stefan Richter, and Peter Rossmanith. Parameterized power domination complexity. *Information Processing Letters*, 98(4):145–149, 2006.
8. Vincent D Blondel, Julien M Hendrickx, Alex Olshevsky, and John N Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 2996–3000. IEEE, 2005.
9. Rainer Hegselmann, Ulrich Krause, et al. Opinion dynamics and bounded confidence models, analysis, and simulation. *Journal of Artificial Societies and Social Simulation*, 5(3), 2002.
10. Cristina Alcaraz and Javier Lopez. Safeguarding structural controllability in cyber-physical control systems. In *European Symposium on Research in Computer Security*, pages 471–489. Springer, 2016.
11. Cristina Alcaraz and Stephen Wolthusen. Recovery of structural controllability for control systems. In *International Conference on Critical Infrastructure Protection*, pages 47–63. Springer, 2014.
12. Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.
13. Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
14. Chin Yang Lee. An algorithm for path connections and its applications. *IRE transactions on electronic computers*, (3):346–365, 1961.
15. Bryan Ek, Caitlin VerSchneider, and Darren A Narayan. Global efficiency of graphs. *AKCE International Journal of Graphs and Combinatorics*, 12(1):1–13, 2015.