Defensive Resource Allocations with Security Chokepoints in IPv6 Networks

Assane Gueye[1(✉)], Peter Mell[2], Richard Harang[3], and Richard J. La[1]

[1] University of Maryland, College Park, MD, USA
{agueye,hyongla}@umd.edu
[2] National Institute of Standards and Technology, Gaithersburg, USA
peter.mell@nist.gov
[3] U.S. Army Research Laboratory, Adelphi, MD, USA
richard.e.harang.civ@mail.mil

Abstract. Securely configured Internet Protocol version 6 networks can be made resistant to network scanning, forcing attackers to propagate following existing benign communication paths. We exploit this attacker limitation in a defensive approach in which heightened security measures are deployed onto a select group of chokepoint hosts to enhance detection or deter penetration. Chokepoints are chosen such that, together, they connect small isolated clusters of the communication graph. Hence, attackers attempting to propagate are limited to a small set of targets or have to penetrate one or more chokepoints. Optimal placement of chokepoints requires solving an NP-hard problem and, hence, we approximate optimal solutions via a suite of heuristics. We test our algorithms on data from a large operational network and discover that heightened security measures are only needed on 0.65 % of the nodes to restrict unimpeded attacker propagation to no more than 15 % of the network.

Keywords: Chokepoints · IPv6 · Moving target · Vertex partitioning · Security

1 Introduction

Consider an attacker stealthily compromising an internal host on an Internet Protocol version 6 (IPv6) network. Leveraging this vantage point, the attacker then attacks other internal hosts, using some of them as launching points for additional attacks. The attacker's goal is to either find a particular resource or to simply gain control of as many internal hosts as possible. We further assume that the attacker will execute attacks at the transport layer and above because the vast majority of network-based attacks operate at these layers [10].

A securely configured IPv6 network can be made resistant to network scanning [7] and other forms of target acquisition [4], limiting the attacker's ability to discover new hosts and subnets. As a consequence, attacker propagation is mostly limited to existing benign communication pathways. We exploit this limited attacker movement in a new defensive model by augmenting a set of

© IFIP International Federation for Information Processing 2015
P. Samarati (Ed.): DBSec 2015, LNCS 9149, pp. 261–276, 2015.
DOI: 10.1007/978-3-319-20810-7_19

hosts with heightened security measures such that they form internal chokepoints for attacker propagation. The attacker will then be required to penetrate these chokepoints or else be limited to a small set of targets. We thus enable a novel defense-in-depth layer that complements traditional security approaches.

The heightened security measures at the chokepoints may take many forms, be detective or preventative, and include a combination of different approaches. The nodes may be hardened to a similar degree as network perimeter devices, may have increased monitoring (human or automated), or may have an enhanced deployment of security software. They may be special purpose servers with limited functionality (thus reducing the overall attack surface). These chokepoints then represent a more difficult target than standard network nodes and/or ones with increased detection capabilities.

We assume that there is budget to place heightened security measures on up k nodes. The question of interest to us then becomes: how do we assign these k nodes in order for them to be effective chokepoints? In this paper, we consider two objective functions: *minimizing the number of hosts an attacker could penetrate without compromising a chokepoint node* and *maximizing the number of chokepoints between the attacker and any random goal host.*

Optimal assignment of chokepoints requires solving the NP-complete [5] vertex separator problem; for this reason, we developed a suite of four greedy heuristic solutions. Aiding our identification of chokepoints is that our empirical evaluation indicates that transport layer communication graphs have two properties: (i) there are many nodes of low degree and few nodes of high degree, while (ii) low degree nodes tend to be connected to high degree nodes and vice versa (i.e., the graph of the network is *disassortative* [18]). The result is that there are many communication paths that tend to overlap onto a subset of hosts, which helps in our design of heuristics for choosing the set of k chokepoints.

To evaluate the effectiveness of our approach and proposed algorithms, we collected transport layer communication data throughout a large operational network consisting of 15509 nodes. We used the empirical degree sequence of this data to generate a suite of uniformly distributed random graphs conformant with the degree distribution within the network. We were thus able to create a variety of test graphs to broaden our test cases beyond the single monitored network. Finally, we ran our algorithms on each random graph and evaluated the performance according to two metrics: (a) the size of the largest cluster and (b) the average number of chokepoints between a pair of nodes.

Once the chokepoints are chosen, periodic monitoring of the communication patterns is necessary to detect ongoing changes and their impact on current chokepoint placement. While the link connectivity remains stable over time (from empirical observation), operational networks change slowly but continually as new users and hosts are added. This dynamic aspect to the communication graph can be accommodated by a mobile defensive technologies (those that can be repositioned rapidly and with little cost). Without mobile defensive measures being available, cluster sizes can always be minimized by adding new chokepoint at an increased cost. With mobile defensive measures, the chokepoints can be

repositioned to near-optimally account for changing communication conditions. Even if the communication graph remains stable, a moving defense regimen may be implemented where the chokepoints migrate over time (e.g., between locally optimal configurations) to prevent attackers from mapping out the defensive topology or learning of chokepoint placement using out-of-band methods.

The remainder of this paper is structured as follows. Section 2 discusses our threat model while Sect. 3 describes network configuration requirements. Section 4 provides our chokepoint approach. Section 5 shows how to minimize the maximal cluster size. Section 6 illustrates how to maximize the mean of the minimum number of chokepoints that must be traversed between pairs of nodes. Section 7 describes our experimental data and Sect. 8 provides empirical results. Section 9 details related work and Sect. 10 concludes.

2 Threat Model

We assume that an attacker has broken through a network's, often porous [14], perimeter security and penetrated an internal host. We also assume that this penetrated host is equally likely to be any node of the network. From this vantage point, the attacker attempts to reach the rest of the network. This includes attacking nodes in order to perform malicious actions directly on those nodes as well as penetrating the nodes to use them as launching points for other malicious activities. Because of the security network configuration discussed in the next section, we model attackers as being limited to propagating through existing or previous transport layer communication channels (using only network knowledge available from the set of compromised hosts). With respect to attacks, we assume that all attacks take place at the ISO transport layer or above. This is where the vast majority of attacks take place [10] since network devices are usually not directly penetrable through the Internet Protocol layer. The presumed goal of the attacker is to either propagate through the network to reach a particular goal host or simply to gain control of as many hosts as possible.

3 Network Configuration Requirements

We model the defended network as IPv6 only. The actual network may contain IPv4 hosts and dual stack IPv4/IPv6 hosts, but our defensive approach applies solely to the IPv6-only subset. This means that as a network slowly migrates to IPv6, our approach will provide monotonically increasing benefits and may thus provide further motivation for a full transition.

We further assume that an attacker is only able to discover new hosts to attack by using the knowledge resident on previously compromised hosts (i.e., the neighbor discovery table). We can do this because we will discuss how it is possible to configure an IPv6 network to mitigate other target acquisition methodologies.

A primary method for target acquisition in IPv4 is network scanning (sending out probes to subnet addresses looking for active hosts). However, in IPv6 the size

of a standard subnet is the square of the number of addresses in the entire IPv4 address space. In a securely configured IPv6 subnet, this makes network scanning infeasible since the number of addresses to be probed is insurmountable [7]. Other methods for target acquisition exist [4] but can be mitigated through a combination of perimeter security and secure configurations.

A method not easily addressed is that of an attacker monitoring layer 2 multicast traffic to "sniff" the addresses of active hosts on a subnet. The primary hurdle is the native IPv6 Neighbor Discovery (ND) protocol that serves as a replacement for the IPv4 Address Resolution Protocol (ARP) (and that has many of the same security weaknesses). However, other layer 2 multicasting protocols exist such as Apple's Bonjour [3] that present similar challenges.

The address disclosures inherent in layer 2 multicasting protocols can be accounted for in our chokepoint approach by incorporating the layer 2 intra-subnet communication along with the set of monitored transport layer flows. This has the effect of making each IPv6 subnet a complete sub-graph in our communication graph because ND will cause all hosts to multicast to all other hosts in a subnet. This is not a desirable situation with respect to our defensive solution.

Thus to limit this effect, we propose making the subnets as small as possible in order to limit the size of these complete sub-graphs. This results in a security versus functionality tradeoff. If the subnets are too small, then automated service discovery protocols (e.g., ND and Apple's Bonjour) will be unable to automatically find services, thus requiring a manual setup. If they are too big, then attackers will gain additional knowledge of network targets.

In evaluating our approach, we assume a high security implementation for the network where administrators are willing to sacrifice automated service discovery to achieve enhanced security. Hence, we model the enterprise taking the security/functionality tradeoff to its limit, using an IPv6/126 subnet for each host. This approach eliminates all layer 2 information leakage by restricting subnets to covering a single host and its related switch port. We then randomly assign these mini-subnets within the larger network address space to preserve network scanning resistance. By doing this, we essentially force layer 2 routing up to layer 3 and gain security advantages by doing so (while giving up some functionality in the form of automatic neighbor discovery and service configuration).

Given this security configuration, an attacker is limited to propagating using only the target address knowledge resident on the previously compromised hosts (i.e., existing benign communication paths). Our work then is to counter an attacker using this knowledge to expand their influence within a network.

4 Chokepoint Approach and Metrics

Our approach is to leverage limited attacker propagation in securely configured IPv6 enterprise networks to inhibit attacker movement or enhance detection by setting up security 'chokepoints'. A chokepoint is a preexisting host that is given heightened security measures where the security measures can be protective,

detective, or a combination of both. The nodes not chosen to be chokepoints are referred to as 'ordinary'. The primary thrust of our work is then in determining how to best choose the chokepoints.

Our goal is to place the chokepoints so that attackers are maximally restricted in their ability to propagate through the network. We do this by modeling the layer 4 network communication as a graph with the nodes being hosts. We then choose the chokepoints in such a way that the graph is broken into isolated clusters when the chokepoints are removed. Each cluster represents an island of available targets to an attacker. To reach nodes in other clusters, however, the attacker needs to penetrate one or more chokepoint nodes. The heightened security at a checkpoint may impede attacker propagation, increase the chance of detecting an attacker, or both (depending upon the type of heightened security deployed).

We consider two metrics to optimize with our choice of chokepoints: (i) minimizing the maximal cluster size and (ii) maximizing the mean of the minimum number of chokepoints that lie between pairs of nodes. The former metric aims to constrain the attacker to as small a cluster as possible, and the second seeks to maximize the number of chokepoints that an attacker must penetrate on average while seeking a particular network target. The next two sections describe algorithms designed to achieve these goals.

Notations: We assume that the network is represented as an un-weighted undirected graph $G = (V, E)$, where V is the set of nodes and E is the set of edges, with $|V| = n$ and $|E| = m$. E is the set of observed benign communication links in the network (not the permissible links). We further assume that there is a budget to deploy enhanced security at up to k nodes. We use S to denote the set of (at most) k designated chokepoint nodes. With a little abuse of notation, we let $G \backslash S$ represent G with the nodes in S removed along with edges incident to them. This graph may be disconnected into a set of isolated clusters or components.

5 Minimizing the Maximum Cluster Size

Our primary goal is to choose the chokepoints in such a way that they maximally bound attacker propagation (either providing impedance points or enhanced detection points). To do this, the communication graph should break into clusters upon removal of the chokepoints. Note that the removal of chokepoint nodes and creation of isolated clusters is solely to model the optimization problem. It does not reflect actual removal of hosts or imply that the chokepoints themselves are invulnerable.

Within the same cluster, any node can attack any other node without encountering a chokepoint. The size of the maximum cluster is hence the largest set of nodes that could potentially be accessed by an attacker without being impeded or detected by a chokepoint. The first goal of chokepoint placement is to minimize the size of this maximal cluster. Unfortunately, this problem is an instance of the vertex separator problem which is known to be NP-hard [5] and has only

received little attention in the literature. In this paper, we consider four heuristic algorithms to find approximate solutions to the problem. The algorithms are iterative in nature and exploit well-known features of the network.

1. Iterative removal of maximal degree node (DEG): This approach is motivated by the observation that engineered networks, such as the Internet, are very sensitive to attacks that target nodes with largest degrees [9]. In our DEG algorithm, we iteratively remove (i.e., choose as chokepoint) the node with the maximum degree in the remaining graph as well as all edges that are incident to it. After each iteration, we re-compute the degree of all nodes and recall the routine until k nodes are removed.
2. Iterative removal of maximal betweenness node (BET): This approach is inspired by the observation that in communication networks there are many low degree nodes that tend to talk to a few high degree 'hub' nodes while the few hub nodes talk to the many low degree nodes. As a consequence, most paths go through the hubs nodes. The fraction of shortest paths from all vertices to all others that pass through a given node is defined as its "*betweenness centrality*". Our BET algorithm is similar to DEG, but here we remove nodes with largest betweenness centrality.
3. Iterative greedy removal (GRD): In this approach, as in previous heuristics, nodes are removed one-by-one. However, instead of removing the node with the largest degree/betweenness, at each step we remove the node that minimizes the size of the current largest cluster. After each iteration, we update the graph and iterate until k nodes are removed.
4. Iterative Vertex Bisection (IVB): In this approach, we attempt to iteratively bisect the largest remaining cluster until all k nodes are removed. For each bisection attempt, we randomly choose pairs of nodes and iteratively grow two non-overlapping trees until we can no longer do so. We take the pair of trees that best optimizes the objective and then compare them against the output of the DEG algorithm (limited to the number of nodes removed in the tree bisection), with the best result being chosen.

In general, we have observed that IVB and GRD provided slightly better results, but not uniformly. While not immediately apparent, DEG can be implemented in $O(n + m)$ linear time and is reasonably effective overall. BET is the slowest among all algorithms but can occasionally produce the best answer. For our analysis, we run all four algorithms for each data points and simply choose the best one (i.e., has the minimum size of maximum cluster) as approximation to the problem.

6 Maximizing Chokepoint Traversal

Our secondary goal is to maximize the mean of the minimum number of chokepoints that an attacker must traverse to reach any given target node. The motivation for this is the observation that each traversed chokepoint increases the chance of impeding or detecting the attacker.

For the analysis, we assume that the attack propagates from some initial node s to some target node t ($s, t \in V$) by following the path containing the smallest number of chokepoints (which corresponds to the best case from the attacker's point of view). In practice, the attack propagation path is determined by the attacker's mode of operation, which we often do not know. However, the aforementioned path with the minimum number of chokepoints offers a *universal lower bound* to the actual number of traversed chokepoints.

We say that an $s-t$ path is "*minimal*" if it contains a minimum number of chokepoints amongst all $s-t$ paths. For a given chokepoint assignment S, we ask the following question: what is the empirical distribution of the number of chokepoints in the minimal $s-t$ paths (i.e., the fraction of minimal $s-t$ paths containing $0, 1, 2, 3, \ldots$ chokepoints)? With this distribution, we can compute the average length of a minimal $s-t$ path. Our goal is to choose the chokepoint so as to maximize this average length of $s-t$ path.

To calculate any minimal $s-t$ path, we would like to use standard shortest path algorithms. However, they do not directly apply because there are both ordinary and chokepoint nodes in the graph whereas our distance metric is only concerned with the number of chokepoints along the paths. To enable the use of standard shortest path algorithms, we need a graph transformation such that, in the transformed graph, all nodes (except s and t) are chokepoint nodes and it retains the connectivity relations of the original communication graph. To this end, we propose the following three-step transformation:

Transformation 1 - (Collapsed Clusters): For each cluster in $G \backslash S$ (clusters may be single nodes), we create a single node in a new graph $G2$. For each node in S, we create a node in $G2$. For each edge $e \in G$, we create a corresponding edge in $G2$ if e is incident to at least one node in S. If one of the incident nodes is not in S, then we use the node in $G2$ that corresponds to the appropriate cluster in G. Finally, we replace any multi-edges in $G2$ with single edges. From its construction, in $G2$ no two ordinary nodes will have an edge between them. The chokepoints may have edges to both other chokepoints and ordinary nodes.

Transformation 2 - (Collapsed Leaves): Collapse all leaves surrounding each chokepoint into a single leaf node to form a second graph CG from $G2$.

Transformation 3 - (Ordinary Node Substitution): The previous two transformations can be done once on the entire graph regardless of the location of s and t. To compute the number of chokepoint in a minimal $s-t$ path, we now construct a final graph, HG, as follow: Make a copy of CG and, for each ordinary node x (except for s and t), add edges connecting x's neighbors (forming complete subgraphs) and remove x. Duplicate edges are removed.

This construction provides us with a graph containing s and t in which all other nodes are chokepoints. The edges in the graph represent the connectivity of the original graph G and thus we can use shortest path algorithms on the final graph HG to determine the minimal number of chokepoints between s and t in G. We now compute all minimal paths from each pair of points in the graph. Our work can be substantially reduced by realizing the all nodes in a single cluster

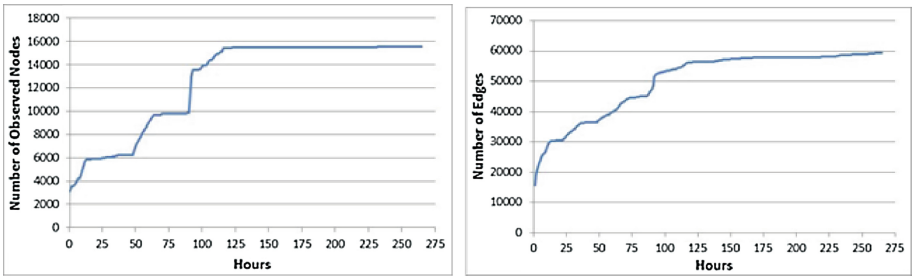
in G can be computed as a single node in CG and HG (since they will all have the same path length to any target node). By computing these values for each (s,t) pair in the collapsed graph CG and making the appropriate updates after each run, we derive the empirical distribution of minimal $s-t$ paths containing l $(l = 0, 1, \ldots, n-1)$ chokepoints. Note that using this procedure, we have considerably reduced the number of (s,t) pairs to consider. Also, the shortest path algorithm is now run on a much small graph (HG) which has only $k+2$ nodes.

7 Experimental Data

For our experiments, we generate random graphs that conform to the degree sequence we obtained by observing the transport layer communication between nodes in a large operational network of 15509 nodes. Unlike typical flow monitoring done on the network perimeter, we monitored flows internal to the network in order to obtain a complete view of the transport layer connectivity.

Note that the monitored network was an IPv4 network while our study pertains to IPv6 networks. However, the flow data represents communications above the IP layer and thus would be similar (if not the same) on an IPv6 substrate.

We monitor the network for 11 days and use all observed links and nodes to construct its connectivity graph. Figure 1a shows the number of nodes observed over time and Fig. 1b shows the number of observed links. By hour 116, almost all of the nodes and links have been observed. This indicates that most of the nodes can be seen communicating with other nodes within 4.83 days. The latter 6 days add only 0.72 % of the total observed nodes and 5.83 % of the total observed links. This indicates that the cumulative data obtained at hour 264 is an accurate description of the real network.



(a) Number of nodes.

(b) Number of edges.

Fig. 1. Evolution over time.

We use the cumulative degree distribution of the built connectivity graph to generate random graphs that are "uniformly" sampled from the set of possible graphs with the given distribution. We do so by using to the following two-step algorithm:

– First, we build an initial random realization of a network according to the given degree sequence using the known Havel-Hakimi algorithm [12,13].
– Second, we use this initial realization to create a random graph by performing a series of random double edge swaps as follows: randomly choose two edges (u, v) and (x, y), where (u, x) and (v, y) are not already in the graph, remove the chosen edges from the graph, and add new edges (ux) and (v, y).

These edge swaps preserve the degree distribution of the graph, and generates a Markov chain that is ergodic over the space of all random graphs with the same degree distribution. Individual samples from this chain at steady state– assuming that the samples are separated by sufficiently large intervals–can then be treated as graphs sampled uniformly at random from the set of all graphs having the same degree distribution. Although the best known upper bound to the convergence rate is very large ($\approx n^{24}$) [11], it has been observed that in practice, the algorithm converges very quickly. In fact, the bound is conjectured to be $O(mlog(m))$, but it still remains an open question [11].

8 Empirical Results

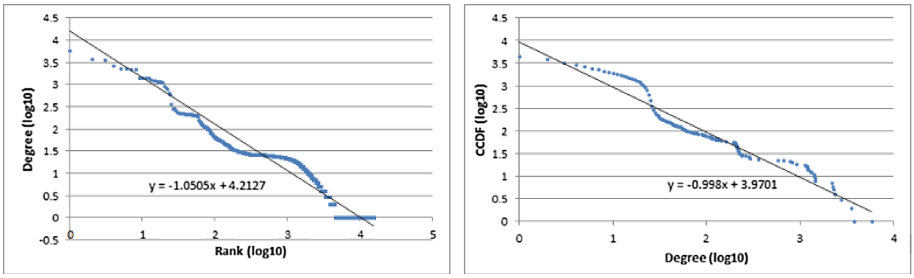
We first examine the degree distribution of the operational network, and compare it to a power law distribution fit to the same data. We use the above algorithm to generate 10000 random graphs with the same degree sequence as the real operational network, and examine the effect of the sampling on the assortativity of the graph. We evaluate the effectiveness of our chokepoint placement procedure by varying the number of chokepoints k. We use the heuristic algorithms described earlier to select the chokepoint nodes and examine the two aforementioned metrics for chokepoint placement. First, we analyze the size of the single largest connected component in the graph formed by removing all nodes designated as chokepoints. Next, we consider the average minimum (again, worst-case for the defender) number of chokepoint nodes that an attacker would have to pass through to target a specific node given a random entry node. For this, we use the graph transformations and algorithm that we describe above in Sect. 6. Our analysis shows that the heuristic node selection method we describe above produces good results for both metrics simultaneously.

8.1 Degree Distribution

We first analyze the degree distribution of the operational network to test whether the network is scale-free [8] or not. We test both the size–rank relationship, as well as the frequency–size relationship proposed respectively in [15] and [8] to check for scale-freeness. Figure 2a shows a log-log plot of the values of the node degrees as a function of their ranks (in reversed order, i.e., highest degree value has lowest rank). Figure 2b shows a log-log plot of the (non-normalized) empirical complementary cumulative distribution function (CCDF) (i.e., for a given degree d, $y(d)$ is the number of nodes with degree strictly larger than d). Notice that these two

plots are related–the former produced by ranking the degrees and the latter produced by looking at the proportion of each degree [1].

Both plots show a linear trend in the log-log plot indicating that the degree distribution of the network follows an approximate power law. In both cases, the exponent of the fitted power law is close to 2. This conforms to the widely observed phenomenon that many engineered networks have a power law distribution (which many authors refer to as scale-free networks).



(a) Rank distribution (b) CCDF

Fig. 2. Log-log plots of (a) the degrees as function of the rank in reverse order. (Slope) $s = -1.05 \implies$ (Exponent) $\alpha = 1.951(1+1/|s|)$. (b) the CCDF of the degree sequence as function of the degree. $s = -0.992 \implies \alpha = 1.992(1+|s|)$.

8.2 Assortativity

Figure 3 shows the empirical cumulative distribution function of the assortativity coefficient of the sampled networks. All sampled graphs had an assortativity coefficient less than zero. This is consistent with previous studies that show that engineered networks such as the Internet, the World Wide Web, the power grid, and many other communication networks are disassortative [18]. Indeed, in this network, on average 51 % of the links are between a low-degree node and a hub. Only 7 % of the edges join low-degree nodes and connections between large hubs are 1 % of the links. A degree is said to be low if it is in the bottom 15 % (i.e., less or equal to 23) and it is said to be high if it is in the top 15 % (i.e., greater or equal to 753). Notice that the assortativity of the operation network falls close to the center of the CDF indicating that our sampling procedure approximates a uniform sampling.

8.3 Size of Maximum Cluster

In this section, we analyze the size of the largest set of nodes that an attacker could potentially have access to–upon compromising some arbitrary (non-chokepoint) host–without encountering a chokepoint. This corresponds to the size of the maximum cluster after the chokepoints are removed from the graph.

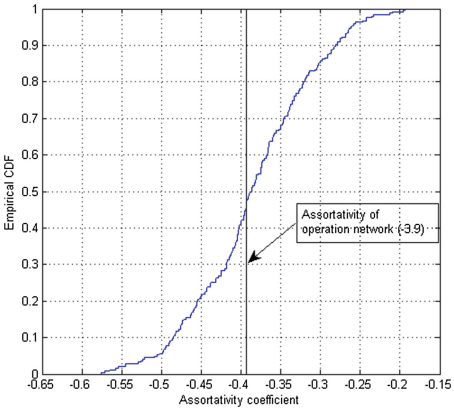


Fig. 3. Empirical distribution of the assortativity coefficient.

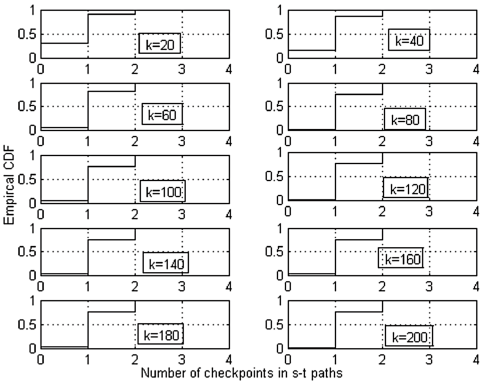
We place the chokepoints using the heuristics described earlier. We consider different values of the parameter k. Figure 4a shows the empirical distribution of the fraction of nodes in the largest cluster for different values of k.

As we can see from the figure, the fraction of nodes in the largest component is less than 40 % of the total number of nodes for all values of k considered in this study. As expected, this fraction of nodes in the maximum cluster decreases as k increases, from around 40 % $k = 20$ to less than 15 % for the maximum value of $k = 200$. It can be observed visually that there is a point of diminishing returns for values of k greater than or equal to roughly 100, beyond which the empirical distribution does not change drastically. This indicates that, for the network under consideration, deploying enhanced security measures to only 100 nodes (0.65 %) of the network is sufficient to reduce the size of the maximum cluster to about 15 % of the total number of nodes, while the use of additional resources produces a relatively minor benefit.

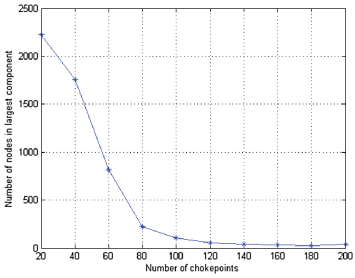
Another way to see this is to plot the size of the largest component as a function of the number of chokepoints, as shown in Fig. 4b. We can see that there is a sharp decrease in the size of the maximum cluster when the number of chokepoints increases from 20 to 80. However, once again diminishing returns can be observed after more than 100 chokepoints.

8.4 Number of Traversed Chokepoints

With the chokepoint defense architecture, an attack may be contained to a small set of nodes (those nodes that share the same cluster) with high probability before being detected. In order to infect other nodes in the network, the attacker needs to traverse one or more chokepoints. In this section, we study the empirical distribution of the minimum number of chokepoints an attacker needs to go through to infect other nodes. In Sect. 6, we have discussed how the minimum number of chokepoints between a pair of nodes is computed. Figure 5a shows



(a) Empirical CDF.



(b) Size of the maximum.

Fig. 4. Characteristics of the size of the maximum cluster. (a) Empirical cumulative distribution of the fraction of nodes in the largest cluster for a varying k. (b) Size of maximum cluster as a function of k.

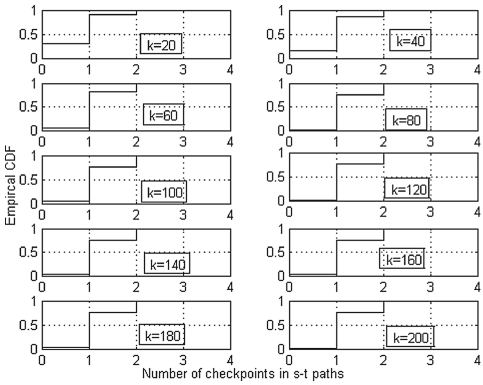
the cumulative distribution function for different values of the parameter k. Figure 5b is an alternative representation where we plot the fraction of minimal paths with a varying number of chokepoints as a function of the number of deployed chokepoints.

First, notice that, an attacker only needs to traverse two chokepoints to reach any node. Recall that this is a lower bound to the actual number of traversed chokepoints. Also, this number is not the shortest path in the network. In fact, an attacker might need to traverse some clusters between the chokepoints and, within each cluster; the attacker might traverse many nodes before reaching a chokepoint (both of these cases are ignored when we compute the minimum number of traversed chokepoints.

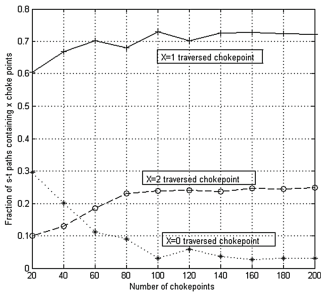
Second, observe (Fig. 5a) that the distribution of the number of chokepoints in minimal paths does not change significantly for $k > 100$. This can also be seen in Fig. 5b where the plots become "flat" for the values of k greater than 100. This implies that, similarly to the maximum cluster size metric, approximately 100 hubs (0.65 %) are sufficient to achieve a significant improvement in security.

The fraction of pairs with zero chokepoints between them represents the fraction of attacks that are undetectable or unpreventable by the chokepoint approach. Figure 5a and b both demonstrate that this fraction decreases as the number of placed chokepoints increases, indicating that placement of chokepoints according to our approach can significantly reduce the mobility of attackers.

The figures also show that most of the minimal paths (around 72 % for $k > 100$) traverse only one chokepoint, suggesting that most nodes in the network are separated by only one chokepoint. This is consistent with (and could be explained by) the observation that about 71 % of the nodes are leaves. Indeed, two leaves are only separated by their common hub.



(a) Empirical distribution.



(b) Fraction of $s-t$ paths containing $X = 0, 1, 2$ chokepoints.

Fig. 5. Characteristics of the minimum number of chokepoints that a node needs to traverse to infect another node.

Finally, we see that the number of minimal paths with one or two chokepoints increases as the number of deployed chokepoints increases (while the fraction of minimal paths with zero chokepoint decreases). At the same time, we never observe the appearance of minimal paths with three or more chokepoints. This suggests that, for this data and this range of values for k, the partially collapsed graph ($G2$) has a diameter of at most 4.

9 Related Work

The problem of placement of monitors for network intrusion detection has been evaluated in a number of other contexts. The recent work by Talele *et al.* [25,26] addresses a similar problem: that of finding a minimal set of IDS placements that will obtain a given graph property. Their work, however, requires knowledge of information flows which we do not consider.

Noel and Jajodia [20] describe the combination of attack graphs with network maps and other security information such as firewalls and known vulnerabilities to construct a topological vulnerability analysis for a network. Sawilla and Ou [22] use attack graphs with a ranking method as a tool for monitor placement, explicitly invoking minimal graph cuts as a method for selecting remediation or focused monitoring. While exploring a closely related problem space to the one we consider, both works makes use of significant side information in the form of attack graphs and the result of vulnerability scans and analysis that we do not consider.

Anjum *et al.* [2] investigates the placement of costly "tamper resistant sensor" nodes within a sensor network that may be under attack by an adversary attempting to compromise the 'sink' node responsible for collecting the sensor data. This work is specialized to the sensor network case, in which all data flows

to a single sink node. We consider a different problem where any node can be the source/target of an attack.

The work by Shaikh *et al.* [24] is the only work to explicitly model cost, modeling a value function for the deployment of intrusion detection sensors within a network. Each sensor location is characterized by its value, and sensors are then deployed in order of descending location value up to the available budget. Unlike our work, they do not incorporate any topological data regarding the network (although some related cost metrics are likely to be influenced by topological considerations).

Mell and Harang [17] examine the use of flow-based analysis (such as we consider here) to perform a triage operation on a network following a compromise. In contrast to the pre-emptive analysis of static graphs that we consider here, [17] focuses on identification of critical nodes ("hubs") for examination following the discovery of an intruder, in such a way that these hubs separate some minimal set of nodes containing the intruder's entry point from the rest of the network.

A very related problem is that of immunization where the goal is to find a set of 'agents' to immunize in order to suppress the spreading of a disease under various propagation models. Those nodes can be viewed as chokepoints as they impede the propagation of a disease. Tong *et al.* [27] has proposed a greedy algorithm that iteratively selects the current node with highest eigen-drop as the one to be immunized next. However, this algorithm was designed for a different metric and a quick test has shown that it performs poorly in our framework. In [19] Nian and Wang propose a high-risk immunization approach that targets susceptible nodes whose neighbors have been infected. This requires knowing all infected nodes in advance making it inapplicable to our problem. A greedy approach based on immunizing the most connected nodes has been proposed by several authors [16,21]. In this paper, we have implemented a version of such an approach (Iterative removal of nodes with largest degree-DEG) that runs in linear time. Our BET algorithm is also very similar to the immunization approach in [23].

Finally, the work by Chen and Hero [6] considers the same problem of removing a set of nodes to minimize the size the maximum cluster in the remaining graph (although for a different purpose). They relate the optimization problem to the spectrum of the graph and propose a heuristic approximation algorithm. Their algorithm has similar performance as DEG in a small test algorithm of 300 nodes. Unfortunately, it involves solving a eigenvalue problem which makes it computationally very expensive for large graphs. For this reason we did not include Chen-Hero's heuristic algorithm in our suite of heuristic approximations.

10 Conclusion and Discussion

Properly configured IPv6 networks are resistant to scanning and other forms of target acquisitions. Attackers attempting to propagate through the network are hence limited to existing benign communication paths. We leverage this limitation of attacker movement to propose a chokepoint-based defensive approach

that maximally bounds attacker propagation. With this approach, attackers are forced to limit their propagation to a small set of nodes or have to penetrate one or more chokepoints. We have considered two optimization criteria to choose the chokepoints: (1) minimizing the maximum number of nodes an attacker can compromise without encountering a chokepoint and (2) maximizing the average number of chokepoints an attacker has to traverse to attack a random target. Optimal placement of these chokepoints is a NP-hard problem and, as a consequence, we have used a set of heuristics to approximate the solution. We have tested our approach and algorithms using data from a large operation network of over 15500 nodes. Our experiments have shown that enhanced security solutions have to be deployed to only 0.65 % of the nodes to limit the propagation of a random attack to less 15 % if the nodes of the network. Our approach thus enables a novel defense-in-depth approach that complements traditional security approaches.

Acknowledgements. This research was sponsored by the U.S. Army Research Labs (ARL) and the National Institute of Standards and Technology (NIST). It was partially accomplished under NIST and University of Maryland, College Park Cooperative Agreement No.70NANB13H012.

References

1. Adamic, L.A.: Zipf, Power-laws and Pareto - a ranking tutorial (2002)
2. Anjum, F., Subhadrabandhu, D., Sarkar, S., Shetty, R.: On optimal placement of intrusion detection modules in sensor networks. In: Proceedings of the 1st International Conference on Broadband Networks, 2004. pp. 690–699, October 2004
3. Apple-Inc: Bonjour for developers. https://developer.apple.com/bonjour/index.html. Accessed on 12 November 2014
4. Bellovin, S.M., Keromytis, A., Cheswick, B.: Worm propagation strategies in an IPv6 internet. Login **31**, 70–76 (2006). Please check the edit made in Ref. [4]
5. Bui, T.N., Jones, C.: Finding good approximate vertex and edge partitions is NP-hard. Inf. Process. Lett. **42**(3), 153–159 (1992)
6. Chen, P., III, A.O.H.: Node removal vulnerability of the largest component of a network. CoRR abs/1403.2024 (2014)
7. Chown, T.: RFC 5157: IPv6 implications for network scanning. Internet Eng. Task Force (IETF) RFC (March 2008). https://tools.ietf.org/html/rfc5157
8. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. SIAM Rev. **51**(4), 661–703 (2009)
9. Cohen, R., Erez, K., ben Avraham, D., Havlin, S.: Breakdown of the internet under intentional attack. Phys. Rev. Lett. **86**, 3682–3685 (2001)
10. Convery, S., Miller., D.: IPv6 and IPv4 threat comparison and best-practice evaluation. http://www.cisco.com/web/about/security/security_services/ciag/documents/v6-v4-threats.pdf. Accessed on 11 June 2014
11. Gkantsidis, C., Mihail, M., Zegura, E.: The markov chain simulation method for generating connected power law random graphs. In: In Proceedings of 5th Workshop on Algorithm Engineering and Experiments (ALENEX), SIAM (2003)
12. Hakimi, S.L.: On realizability of a set of integers as degrees of the vertices of a linear graph. I. J. SIAM **10**(3), 496–506 (1962)

13. Havel, V.: A remark on the existence of finite graphs. Casopis Pest. Mat. **80**, 477–480 (1955)
14. Landry, B.J.L., Koger, M.S., Blanke, S., Nielsen, C.: Using the private-internet-enterprise (PIE) model to examine it risks and threats due to porous perimeters. Inf. Secur. J.: Glob. Perspect. **18**(4), 163–169 (2009)
15. Li, L., Alderson, D., Doyle, J.C., Willinger, W.: Towards a theory of scale-free graphs: definition, properties, and implications. Internet Math. **2**(4), 431–523 (2005)
16. Madar, N., Kalisky, T., Cohen, R., ben Avraham, D., Havlin, S.: Immunization and epidemic dynamics in complex networks. Eur. Phys. J. B Condens. Matter Complex Syst. **38**(2), 269–276 (2004)
17. Mell, P., Harang, R.E.: Using network tainting to bound the scope of network ingress attacks. In: SERE 2014, pp. 206–215 (2014)
18. Newman, M.E.J.: Mixing patterns in networks. Phys. Rev. E **67**(2), 026126 (2003)
19. Nian, F., Wang, X.: Efficient immunization strategies on complex networks. J. Theor. Biol. **264**(1), 77–83 (2010)
20. Noel, S., Jajodia, S.: Optimal ids sensor placement and alert prioritization using attack graphs. J. Netw. Syst. Manag. **16**(3), 259–275 (2008)
21. Pastor-Satorras, R., Vespignani, A.: Immunization of complex networks. Phys. Rev. E **65**, 036104 (2002)
22. Sawilla, R.E., Ou, X.: Identifying critical attack assets in dependency attack graphs. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 18–34. Springer, Heidelberg (2008)
23. Schneider, C.M., Mihaljev, T., Havlin, S., Herrmann, H.J.: Suppressing epidemics with a limited amount of immunization units. Phys. Rev. E **84**, 061911 (2011)
24. Shaikh, S.A., Chivers, H., Nobles, P., Clark, J.A., Chen, H.: A deployment value model for intrusion detection sensors. In: Park, J.H., Chen, H.-H., Atiquzzaman, M., Lee, C., Kim, T., Yeo, S.-S. (eds.) ISA 2009. LNCS, vol. 5576, pp. 250–259. Springer, Heidelberg (2009)
25. Talele, N., Teutsch, J., Erbacher, R.F., Jaeger, T.: Monitor placement for large-scale systems. In: Proceedings of the 19th ACM Symposium on Access Control Models and Technologies. pp. 29–40 (2014)
26. Talele, N., Teutsch, J., Jaeger, T., Erbacher, R.F.: Using security policies to automate placement of network intrusion prevention. In: Jürjens, J., Livshits, B., Scandariato, R. (eds.) ESSoS 2013. LNCS, vol. 7781, pp. 17–32. Springer, Heidelberg (2013)
27. Tong, H., Prakash, B., Tsourakakis, C., Eliassi-Rad, T., Faloutsos, C., Chau, D.: On the vulnerability of large graphs. In: 2010 IEEE 10th International Conference on Data Mining (ICDM), pp. 1091–1096, December 2010