Towards Automated Cyber Decision Support: A Case Study on Network Segmentation for Security

Neal Wagner, Cem Ş. Şahin, Michael Winterrose,
James Riordan, Jaime Pena, Diana Hanson, and William W. Streilein
MIT Lincoln Laboratory
244 Wood Street Lexington, MA 02420, USA
Email: neal.wagner, cem.sahin, michael.winterrose, james.riordan, jdpena, dhanson, wws@ll.mit.edu

***Abstract*—Network segmentation is a security measure that partitions a network into sections or segments to restrict the movement of a cyber attacker and make it difficult for her to gain access to valuable network resources. This threat-mitigating practice has been recommended by several information security agencies. While it is clear that segmentation is a critical defensive mitigation against cyber threats, it is not clear how to properly apply it. Current standards only offer vague guidance on how to apply segmentation and, thus, practitioners must rely on judgment. This paper examines the problem from a decision support perspective: that is, how can an appropriate segmentation for a given network environment be selected? We propose a novel method for supporting such a decision that utilizes an approach based on heuristic search and agent-based simulation. We have implemented a first prototype of our method and illustrate its use via a case study on a representative network environment.**

I. Introduction

Modern states require secure computer networks to support most, if not all, sectors of their critical infrastructure including those of finance, healthcare, transportation, wastewater, information technology, and defense [1]. Recent high-profile cyber breaches [2], [3] demonstrate that the threat faced by these networks is serious. Network segmentation (sometimes referred to as network *compartmentalization* or *partitioning*) has been proposed by a number of corporations and government agencies as an important defensive mitigation to help counter this threat [4], [5], [6], [7].

Network compartmentalization refers to the practice of partitioning a network into sections or segments in which communications between segments and between segments and the Internet are controlled [8]. The purposes are to limit a cyber attacker's ability to move about the network, to make it difficult for her to gain access to valuable resources, and to increase the ability of the defender to monitor network communications and detect and remediate cyber intrusions. While it is clear that segmentation is a crucial defensive mitigation, it is not clear how it can be properly applied to protect a given network. For even small networks many different segmentation architectures are possible and the number of possibilities grows exponentially with network size. Current standards and best practices, for example as given in [5], [9], offer only vague guidance on how to apply compartmentalization. This forces security practitioners to rely on judgment.

This paper examines the problem from a decision support perspective: that is, *how can an appropriate segmentation for a given network environment be selected?* We propose a novel method for supporting such a decision that utilizes a computational intelligence approach. The method combines testbed experimentation, hierarchical simulation modeling, statistical analysis, and optimization techniques to search the space of segmentation architectures and recommend an optimal/near-optimal partitioning. We have implemented an initial prototype of our method and illustrate its use via a case study on a representative network environment.

The rest of this paper is organized as follows. Section II gives a brief review of the current state of decision support in the cyber security domain, Section III describes network segmentation and its use as a defensive mitigation, Section IV provides the details of the multi-component method to support the segmentation decision problem, Section V discusses a case study conducted to illustrate our method on a representative network environment, and Section VI concludes.

II. The Current State of Cyber Security Decision Support

While there exists a large body of work in the area of cyber situational awareness (for example see [10]), the field of cyber security decision support is still developing. Current literature in this field is largely scattershot, with individual studies focused on specific cyber-related problems rather than generalized cyber decision system approaches.

One recent study proposes a dynamic forecasting methodology that makes use of qualitative assessments by subject matter experts to recommend contingency measures to combat cyber threats [11]. Another study details a cyber infrastructure to facilitate and secure individual-based decision making and negotiation for Internet-of-Things devices with respect to applications in health care [12]. In [13], a combination of Bayesian Belief Network, cyber vulnerability assessment, and expected loss computation are used to compute appropriate premiums for cyber insurance products.

One study that does take a more generalized approach is given in [14]. This study proposes a net-centric architecture to execute a *cyber OODA loop*[1] to support decisions that focus on cyber survivability of Ballistic-Missile-Defense systems under

[1]An OODA loop is a cycle of observing, orienting, deciding, and acting [15]. A cyber OODA loop executes this cycle in the cyber domain.

cyber attack. While the proposed architecture can indeed be beneficial to decision makers, it only supports decisions by enabling users to rapidly produce, consume, and share critical information rather than recommending appropriate or optimal decisions.

One relevant commercially-developed system is Netflix's FIDO [16], which stands for Fully Integrated Defense Operation. The FIDO system is designed to ingest and process detection alerts from various security products (e.g. firewalls and anti-virus packages), prioritize/score these alerts, and execute simple remediation actions (e.g. ending a VPN session or disabling a user account) or notify appropriate (human) parties. While this system is able to act (semi)-autonomously, the function it executes is quite similar to many commercial-off-the-self intrusion detection systems and, thus, its applicability is limited to this one type of problem.

This study differs from the current literature on cyber decision support in two fundamental ways: (1) it proposes a *generalized* cyber decision support methodology that is applicable for a wide range of cyber-security decision problems and (2) the proposed method can generate optimal/near-optimal decisions *automatically* without requiring qualitative input and/or judgment from human analysts. The main contributions of this study are as follows.

- The study proposes a novel and generalized cyber decision support methodology that generates appropriate decisions automatically.
- A prototype of the proposed methodology is developed and applied to a critical cyber-security decision problem, specifically that of selecting an appropriate network segmentation architecture.
- A case study is conducted illustrating the use of the prototype system to automatically generate efficacious segmentation architectures for a representative network environment under cyber attack.

III. Network Segmentation for Security

Network segmentation is a defensive mitigation that attempts to thwart a cyber attacker's ability to move within a network. As described in Section I, it is concerned with partitioning a network into segments and controlling communications between segments and between segments and the Internet. The goal is to protect network resources by restricting communications which, in turn, has several potentially beneficial effects from the standpoint of security by: (i) reducing the number of entry points into a network, (ii) limiting the network access of an attacker who has penetrated the network, (iii) hindering the lateral movement of attacker and her ability to pivot to other network devices, and (iv) making monitoring communications easier and increases the defender's ability to detect and remediate cyber intrusions [17].

Partitioning has been proposed in various forms by several information security agencies such as Microsoft [6], SANS Institute [7], and the Information Assurance Directorate (IAD) of the National Security Agency (NSA) [5]. However, there is no clear method proposed to determine a proper partitioning architecture for a given network environment. Current best practices and information security recommendations offer only vague guidance.[2]

Two highly-visible versions of segmentation guidance are proposed by IAD: (i) a coarse-grained version called *Segregate Networks and Functions* (SNF) and (ii) a fine-grained version coined *Limit Workstation-to-Workstation Communication* (LWC).

SNF posits that different cyber assets (e.g., hosts, servers, subnets) are used for different organizational functions (e.g., public-facing web services, financial transactions, human resource management, etc.) that have differing sensitivity levels and security requirements [17]. This version of segmentation recommends partitioning a network into groups of assets based on the function that these assets are intended to carry out. Partitioning at this level is usually implemented by firewalls, network egress and ingress filters, application-level filters, and/or physical (hardware) infrastructure [18].

LWC proposes a partitioning architecture that is more restrictive and operates on the *principle of least privilege* [19] which specifies that communication is allowed only when necessary for task execution. Here, even devices within the same functional unit may face communication restrictions. This level of partitioning can be implemented by setting device-level firewall rules (e.g., Windows Firewall rules), disabling remote logon access to devices, and using private virtual LANs [19].

However, both of the IAD-recommended versions of segmentation leave much to the judgment of a security practitioner who still must determine how segments are interconnected, the number and kind of communications allowed between these segments, and which segments are allowed to communicate with the Internet. The problem is further complicated by cost considerations (i.e., finer-grained partitioning may be more expensive and/or error prone) and the fact that some cyber assets may be used for more than one function.

There does exist one example in industry of a well-specified version of network partitioning implemented by Google, namely BeyondCorp [20]. Here, the approach is to individually authenticate each network user and device, regardless of the user's network location (on-site or external), and enforce access control to enterprise applications that is customized to each user/device. This can be considered an extreme version of LWC where each device is an individual segment.

While Google's approach is certainly not vague, there are a number of potential pitfalls associated with it. First, the cost of implementing this approach may be too high for many organizations. This approach requires a rather complex infrastructure that includes a complete device inventory and access control that relies on inferred levels of trust based on access-granting rules that change over time. The cost of instantiating, maintaining, monitoring, and supporting such an infrastructure may be beyond the reach of all but the most resource-wealthy organizations. Customized device-level access control for a large organization with many different

[2]For example, see the guidance offered in [9].

kinds of users and enterprise applications may also be prone to mis-implementation. This can allow unintentional access to network resources by insecure users/devices. Finally, even ignoring cost and error concerns, it is not clear that device-level segmentation always leads to better security posture. In a sufficiently complex cyber system there may exist a labyrinth of allowed communications for which overall system security can not be readily determined.

Here, we investigate the network segmentation problem from a decision support point-of-view: *how can we select an appropriate segmentation architecture for a given network environment?* We will refer to an individual segment of a partitioned network as an *enclave*, that is a group of network devices with homogeneous reachability.

IV. Network Segmentation Decision Support

We propose a cyber decision system to recommend suitable segmentation architectures with respect to a network environment. Here, network environment refers to a collection of network users/devices, a cyber threat, a set of cyber defensive measures, and a mission that the network is intended to support. We assert that the mission component of the environment is critical. It is a trivial matter to construct a defensive policy that maximizes security: simply disallow network communications to and from external sources.[3] However, such a policy can make mission operations difficult or impossible and effectively render the network useless. Therefore the selection of segmentation architecture must consider both network security and mission performance.

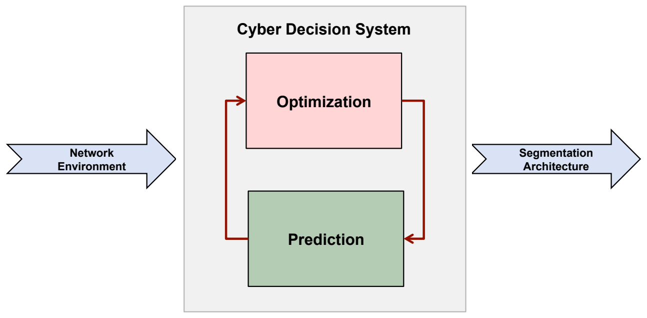


Fig. 1. Generalized cyber decision system framework: an iterative loop between logical constructs of optimization and prediction

To this end we propose a cyber decision support method to search the space of possible partitioning architectures and recommend an optimal/near-optimal segmentation. Our method is based on a computational intelligence framework developed for recommending optimal decisions in the supply chain domain [21]. Fig. 1 depicts this generalized decision system framework. As show in the figure, the system inputs data specifying the network environment to be considered and outputs a recommended segmentation architecture. The system is made of two logical constructs, optimization and prediction, that execute iteratively to search the decision space. The former construct selects candidate cyber decisions and feeds these to the prediction construct for evaluation. The latter predicts the efficacy of candidate decisions and feeds evaluations back to optimization. These evaluations are then used by optimization to select new candidate decisions, which are fed to prediction, and so on. The end result is a recommended decision that has been optimized over multiple iterations of the decision selection/evaluation loop.

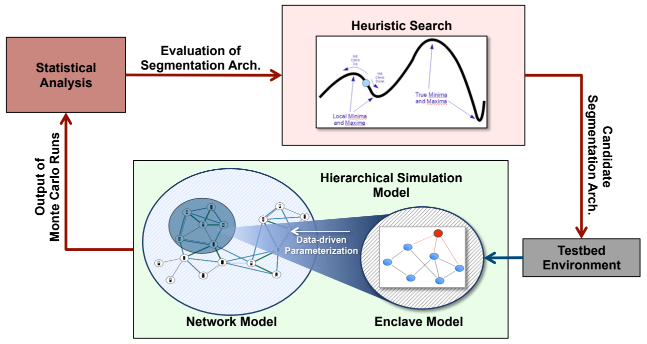


Fig. 2. Network segmentation decision method: testbed, simulation model, statistical analysis, and heuristic search components execute in an iterative loop.

Fig. 2 gives our multi-component decision method for the network segmentation selection problem. In the figure, the heuristic search component fulfills the duties of the optimization construct while the testbed and hierarchical simulation model components fulfill the duties of the prediction construct. The heuristic search component selects a candidate segmentation architecture to be modeled via a combination of testbed experimentation and hierarchical simulation modeling (introduced in our previous work [22] and summarized below). Multiple simulation runs are executed on these components and run outputs are piped to the statistical analysis component. The statistical analysis component is used to quantify these outputs into a single evaluation metric, which is then piped to the search component to help guide its selection of new candidate segmentations. Here, for simplicity, we assume that all network entities (e.g. devices, users, etc.) are known. The following provides the details of each component given in Fig. 2.

A. Testbed environment

A proprietary testbed is used to capture a segmented network at a coarse-grained level of abstraction [22]. It allows for the instantiation of a partitioning architecture that divides the network into enclaves and restricts communications between enclaves and between enclaves and the Internet. The testbed environment specifies communication channels by allowing or disallowing software services between enclaves. The environment also includes the notion of enclave cleansing by the defender: compromised enclaves are periodically cleansed and restored to an uncompromised state. The testbed uses data

[3]Note that this assumes a completely compromise-free network at the time of policy enactment: if this is not the case, then even this will not ensure perfect security.

from real software vulnerabilities and corresponding exploits to characterize the vulnerability level of individual enclaves with respect to a given network segmentation architecture and enclave cleansing rate. The environment measures the probability that an enclave has been penetrated but does not capture instances of actual device compromise within an enclave.

Planned future work will focus on developing a simulation model to replace the testbed environment so that more segmentation scenarios can be easily examined as the resource cost of executing scenarios on the testbed is relatively high.

B. Hierarchical simulation model

The simulation model seeks to capture device-level dynamics not captured by the testbed environment. Specifically, the goal is to capture dynamics of mission performance, infection spreading, and cleansing within individual enclaves of the network. The simulation model is built using a hierarchical structure with two components: (i) the enclave model and (ii) the network model. The former captures device-to-device infection spreading and enclave cleansing dynamics within a single enclave while the latter characterizes attack/defense dynamics and how these dynamics impact mission performance at the full network scale.

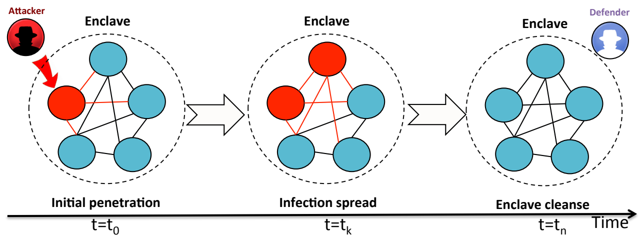


Fig. 3. Enclave model attack/defense dynamics. Red nodes represent infected devices within an enclave.

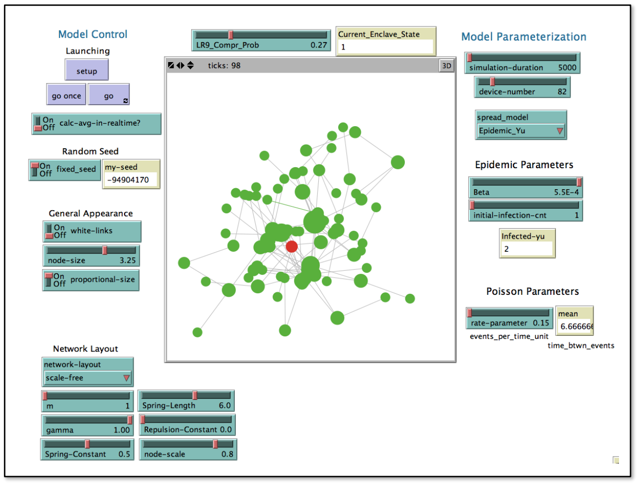
Enclave model: We utilize the approach given in [23] to model infection spreading within an enclave as:

$$I(t) = I(0) \times e^{\beta \times N \times t} \tag{1}$$

where t is time, $I(0)$ is the infected number of devices at $t = 0$, β is the infection propagation rate and N is the total number of devices in a given enclave. $I(t)$ gives us the total number of infected devices for any given time t. An illustration of initial penetration and infection spreading is shown in Fig. 3.

Our model specifies a random variable to capture the probability that an enclave is in a vulnerable state (i.e., whether or not it has been penetrated by the attacker). Assume that at time $t = t_0$, the enclave is vulnerable. As shown in Fig. 3, the attacker penetrates the enclave by compromising a single device (depicted by the red node in the left-most enclave state of the figure) and then attempts to spread to other devices in the enclave. At time $t = t_k$, the attacker successfully spreads to another device in the enclave (depicted in the middle enclave state of the figure). At time $t = t_n$, the defender cleanses the enclave and dis-entrenches the attacker (depicted in the right-most enclave state of the figure). When the attacker is dis-entrenched, all enclave devices are restored to their original (uninfected) state.

The enclave model was implemented in NetLogo [24]. Fig. 4 provides an example screenshot of the enclave model.[4]



LR9_Compr_Prob 0.27
Current_Enclave_State 1
ticks: 98
calc-avg-in-realtime? Off
fixed_seed On
my-seed -94904170
white-links Off
node-size 3.25
proportional-size On
network-layout scale-free
m 1
gamma 1.00
Spring-Constant 0.5
Spring-Length 6.0
Repulsion-Constant 0.0
node-scale 0.8
simulation-duration 5000
device-number 82
spread_model Epidemic_Yu
Beta 5.5E-4
initial-infection-cnt 1
Infected-yu 2
rate-parameter 0.15
events_per_time_unit
mean 6.66666
time_btwn_events

Fig. 4. Screenshot of the enclave model (green nodes represent uncompromised devices, red nodes represent compromised devices).

Network model: As discussed above, the network model captures attack and defense dynamics at the full network scale and how these dynamics impact mission operations. We utilize agent-based simulation to capture attackers, defenders, and mission actors interacting in a network environment. We model a representative mission based on a military-style Air Operations Center (AOC). The AOC mission is concerned with gathering air tasking requests, processing these into a finalized air tasking order that dictates what flights will occur, and making this order available to the appropriate parties.

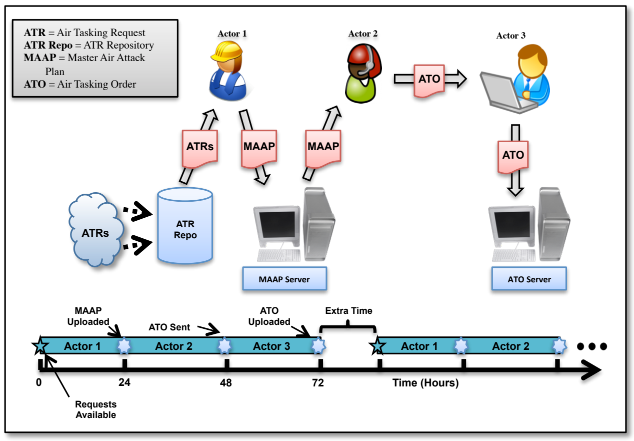


Fig. 5. Abstracted Air Operations Center (AOC) mission concept.

Fig. 5 gives a graphical depiction of the AOC mission. From the figure, the abstracted AOC mission model characterizes

[4]Parameter settings given in the figure are for illustrative purposes and do not represent experimental settings used in this study.

three mission actors that make use of three mission servers to complete their tasks. The first actor periodically collects air tasking requests (ATRS) from an ATR repository, takes some time to process these requests into a master air attack plan (MAAP), and uploads the MAAP to the MAAP server, shown in the center of Fig. 5. The second actor checks the MAAP server for the MAAP and, upon seeing it, downloads and processes this into an air tasking order (ATO) and sends this to a third actor. The third actor then prepares the ATO for distribution and uploads it to the ATO server, shown to the right of the diagram, where it is now available for use.

Network model attack and defense dynamics are an abstraction of the attack and defense dynamics captured in the enclave model but from a full network perspective where outcomes vary depending on the micro-environment specified for individual enclaves in the full network. Devices compromised by the attacker incur downtime (i.e., become unavailable) until they are restored and this downtime may impede mission operations and cause mission delay. The model specifies a random variable to capture the probability that a device in a given network enclave is compromised. At simulation time $t = 0$, this variable is used to determine which devices in a given enclave are compromised and, for those that are compromised, a second random variable determines the duration of compromise. This initialization process is repeated separately for each enclave of the network. As simulation time progresses, compromised devices are cleansed and restored when their compromise durations have completed. Network model outputs measure overall security and mission impact at the network scale.[5]

Hierarchical integration of testbed and simulation model: As depicted in Fig. 2, the testbed and simulation model are connected in a hierarchical structure. Testbed experiments are executed, results are aggregated, and then these results are used to parameterize the next level of the hierarchy, the enclave model. Finally, simulation runs are executed on the enclave model, results aggregated, and used to parameterize the top level of the hierarchy, the network model. Together, the testbed, enclave, and network models capture the security and mission performance dynamics inherent to a network environment under a given segmentation architecture. As mentioned in Section IV, we first introduced this combination of testbed experimentation and simulation modeling to examine network segmentation in a previous work [22].

C. Statistical Analysis

We aim to build a decision support system that can recommend optimal/near-optimal segmentation architectures for a given network environment. As discussed at the beginning of this section, we wish to consider both security and mission performance. A statistical analysis component (shown in Fig. 2 to the left) is used to measure a candidate architecture's effectiveness and computes the *unified effectiveness metric* proposed in [25]. The unified metric, m_g, incorporates two sub-measures: (i) a security index (sec_i): specified as the expected ratio of device availability time to total time, and (ii) a mission delay measure (m_d): specified as the expected mission delay.

The complete metric, m_g, is defined as a measure to characterize sec_i and m_d inherent to a simulated network environment captured via Monte Carlo experiments. The metric incorporates effects of mean, median, and variance of results from multiple simulation runs. The idea of the metric is to quantify the gain in effectiveness due to a given segmentation architecture relative to a baseline where no segmentation is used (i.e., a flat network in which all devices have uniform reachability). The metric is normalized to $[0, 1]$ where higher values mean greater overall effectiveness with 1 meaning 100% effectiveness (i.e. attacker threat is completely nullified, mission performance is optimal) and 0 meaning 0% effectiveness (i.e. no improvement relative to the baseline).

Algorithm 1 Simulated Annealing Search

```
1: PROCEDURE: Optimize-Network-Architecture ($s_0$, $s_{baseline}$, $k_{max}$) $\{s_0$: initial architecture, $s_{baseline}$: baseline architecture, $k_{max}$: maximum no. of iterations$\}$
2: $s \leftarrow s_0$ $\{$Accept $s_0$ as current solution$\}$
3: $k \leftarrow k_{max}$
4: repeat
5:     $T \leftarrow \frac{k}{k_{max}}$ $\{$Set temperature $T\}$
6:     $s_{new} \leftarrow$ Generate-Neighbor-Architecture($s$) $\{$Pick random neighbor of $s\}$
7:     $E(s_{new}, s_{baseline})$ $\{$Compute effectiveness $E$ of $s_{new}$ relative to $s_{baseline}\}$
8:     $E(s, s_{baseline})$ $\{$Compute effectiveness $E$ of $s$ relative to $s_{baseline}\}$
9:     if $E(s_{new}, s_0) > E(s, s_0)$ $\{s_{new}$ more effective than $s\}$ then
10:        $s \leftarrow s_{new}\{$Accept $s_{new}\}$
11:    else
12:        $r \leftarrow$ random value $\in [0, 1]$
13:        if $r \leq e^{\frac{E(s_{new}, s_{baseline}) - E(s, s_{baseline})}{T}}$ then
14:            $s \leftarrow s_{new}$ $\{$Inferior $s_{new}$ is accepted$\}$
15:        end if
16:    end if
17:    $k \leftarrow k - 1\{$Decrementing $k$ reduces $T\}$
18: until $k = 0$
```

Here, we summarize the metric's computation.

1) We run a set of Monte Carlo simulation experiments for two scenarios: (i) a network environment with a given segmentation architecture facing a given cyber attack threat and (ii) a baseline scenario in which no segmentation is used facing the same attack threat. Simulation outputs sec_i and m_d are collected for these scenarios.
2) These outputs are histogrammed and then each is fitted to the Normal distribution.
3) The area under each fitted curve is integrated and then multiplied by the mean of each measure resulting

[5]The full network-scale attack and defense model is specified in [22]. Due to space limitations, we do not provide a screenshot of the network model here.

in two probability density functions, $seci_g$ and md_g, representing the *enhanced* versions of the security index and mission delay, respectively.

4) The final unified metric, m_g, is computed as the average of $seci_g$ and md_g.

Algorithm 2 Generate Neighbor of Architecture s

```
1: PROCEDURE: Generate Neighbor of Architecture ($s$) {$s$: architecture}
2: $s \leftarrow f\{\text{Add-Remove-Service}(s), \text{Merge-Split-Enclave}(s)\}$
   {$f$: function that randomly selects one of the two architecture-generation procedures (Add-Remove-Service, Merge-Split-Enclave) and executes it}
```

D. Heuristic Search

The final component in our decision system is tasked with selecting new candidate segmentation architectures to evaluate. This component utilizes evaluations received from the statistical analysis component to choose promising, as-yet-unexamined architectures. All components work together iteratively to examine promising segmentations and progressively hone in on the optimal (i.e. the segmentation with maximum effectiveness).

In our prototype system, we leverage the simulated annealing (SA) technique [26] which prescribes an iterative search for the global optimum by examining neighboring solutions of the currently accepted solution. For our application, we search for the global maximum. Algorithms 1-4 specify our SA-based heuristic search.

Algorithm 3 Add/Remove Services from s

```
1: PROCEDURE: Add-Remove-Service(s) {$s$: architecture}
2: $n \leftarrow$ random integer $\in [1, \max]$ {Select the number of services to add/remove}
3: $r \leftarrow$ random value $\in [0, 1]$
4: if $r < 0.5$ then
5:     $s$ is modified to randomly remove $n$ services
6: else
7:     $s$ is modified to add $n$ services in-between random enclaves
8:
9: end if
```

Algorithm 1 requires two architectures as parameters, an initial architecture, s_0 (i.e., “solution 0”), that is the starting point of the search and a baseline architecture, $s_{baseline}$ (i.e., “solution baseline”) that is used as a reference from which to compute the relative effectiveness of candidate architectures. Candidate architectures are generated by constructing “neighbors” of the currently accepted solution as given in Algorithms 2-4. The effectiveness of candidate architectures is computed via the unified metric described in Section IV-C. Currently we explore two basic architecture-generating operations: adding/removing services that allow communication between enclaves (Algorithm 3) and merging/splitting enclaves (Algorithm 4). Future work is planned to explore additional operations such as group-based operations which take sub-structures of an architecture (i.e. a group of enclaves and the services that connect them) and modify and/or replace them with new sub-structures.

Algorithm 4 Merge/Split Enclaves in s

```
1: PROCEDURE: Merge/Split Enclaves ($s$) {$s$: architecture}
2: $i \leftarrow$ random integer $\in [1, \text{max-iters}]$ {Select the number of iterations to execute merge/split}
3: $n \leftarrow$ random integer $\in [1, \text{max-enclaves}]$ {Select the number of enclaves to be merged/split}
4: $r \leftarrow$ random value $\in [0, 1]$
5: if $r < 0.5$ then
6:     repeat
7:         $s$ is modified to randomly select $n$ enclaves and merge them into one one
8:         $i \leftarrow i - 1$
9:     until $i = 0$
10: else
11:     repeat
12:         $s$ is modified to randomly select $n$ enclaves and split each one into two
13:         $i \leftarrow i - 1$
14:     until $i = 0$
15: end if
```

Due to the relatively high cost of executing testbed experiments we are limited in the number of segmentation architectures that can be considered with respect to a given network environment. As mentioned in Section IV-A, we plan future work to develop a simulation model to replace this testbed and support faster experiment execution and the examination of a greater number of segmentations. Once we are free to leverage such a simulation model, we plan to explore population-based heuristic search techniques such genetic algorithms or particle swarm optimization [27], [28].

V. Experiments

The proposed cyber decision system is implemented by a combination of technologies including NetLogo [24] (for the hierarchical simulation model), Python 2.7 (for the statistical analysis and heuristic search components), and a suite of proprietary cyber testbed hardware and software. We illustrate the use of the system via a case study in which an optimized segmentation architecture is generated for a representative network environment. The goal of our experiment is to start with a segmentation architecture based on best practices and by multiple iterations through our decision system generate an improved architecture with respect to security and mission performance. Ideally, the system-recommended architecture would more effectively inhibit a network intruder’s movement within the network and, in turn, improve mission performance.

A. Experimental Setup

For our representative environment, we examine a class-C-sized network with $N = 250$ devices. We model the AOC

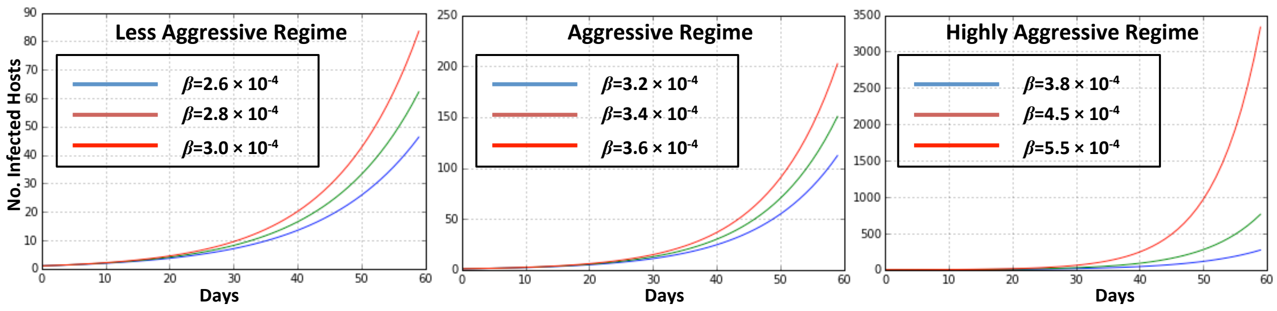


Fig. 6. The impact of infection rate per unit time, β (Eq. 1), on the spreading progress for a vulnerable network with $N = 250$ and $I(0) = 1$.

mission (described in Section IV-B) where a full mission takes three simulated days to complete if uninterrupted and each of the three mission actors requires one day to complete his/her mission task. Forty missions are executed in parallel during a single simulation run in which forty mission actor groups (MAGs), consisting of three actors each, interact with a single set of (three) mission servers. A simulated day is divided into time units where $1,000$ time units = 1 day. For each iteration of the decision system (Fig. 2), we execute $1,500$ simulation runs on the simulation model component where each run is terminated upon completion of all missions or when simulation time reaches a maximum of 30 simulated days.

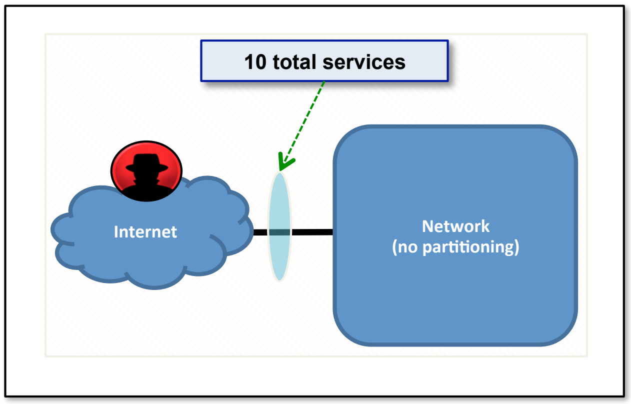


Fig. 7. Baseline architecture $s_{baseline}$ that includes no partitioning.

For our experiments, we examine three threat regimes with respect to infection spreading. These regimes represent differing severities of threat: (i) less aggressive spreading (low threat), (ii) aggressive spreading (medium threat), and (iii) highly aggressive spreading (high threat). As shown in Fig. 6, the infection spread rate, β (from Eq. 1), is varied to model these different severities. To account for the uncertainty inherent to the spreading dynamic, we sweep a range of β values within each regime. Less aggressive spreading is given by $\beta = \{2.6 \times 10^{-4}, 2.8 \times 10^{-4}, 3.0 \times 10^{-4}\}$ which captures an attacker that can infect less than 40% of class C network in 60 days. Aggressive spreading is given by $\beta = \{3.2 \times 10^{-4}, 3.4 \times 10^{-4}, 3.6 \times 10^{-4}\}$ and captures an attacker that can infect up to 80% of the network in 60 days. Finally, highly aggressive spreading is given by $\beta = \{3.8 \times 10^{-4}, 4.5 \times 10^{-4}, 5.5 \times 10^{-4}\}$ and models an attacker who can infect the entire network in less than 30 days.

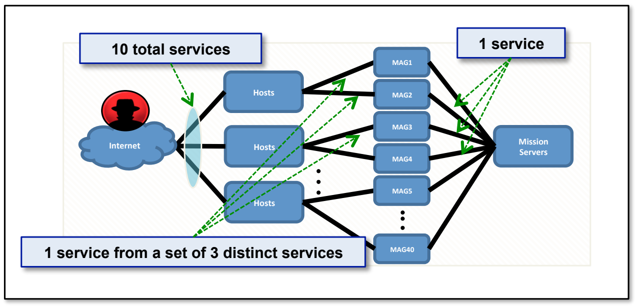


Fig. 8. Initial architecture s_0 that segregates mission and non-mission functions.

As discussed in Section IV-D, the system requires two segmentation architectures as input: a baseline architecture, $s_{baseline}$, used to gauge the effectiveness of candidate architectures and an initial architecture, s_0, that serves as the starting point of the search. Figs. 7 and 8 give these two input architectures.

Fig. 7 depicts $s_{baseline}$, a “flat” network with no partitioning that allows direct communication with the Internet via 10 software services. Note that $s_{baseline}$ represents a network in which all devices are contained in a single enclave.

Fig. 8 gives s_0, a segmented network in which mission and non-mission functions are separated from each other. In s_0, the network is partitioned into 44 total enclaves: three non-mission enclaves each allowing direct communication with the Internet via a total of 10 services evenly split between them, 40 MAG enclaves each consisting of a single MAG that is allowed direct communication with the mission server enclave and one of the three non-mission enclaves, and finally a single mission server enclave consisting of the three mission servers that is allowed to directly communicate with each of the MAG enclaves. This segmentation is meant to represent a version of the SNF architecture recommended by IAD and described in Section III.

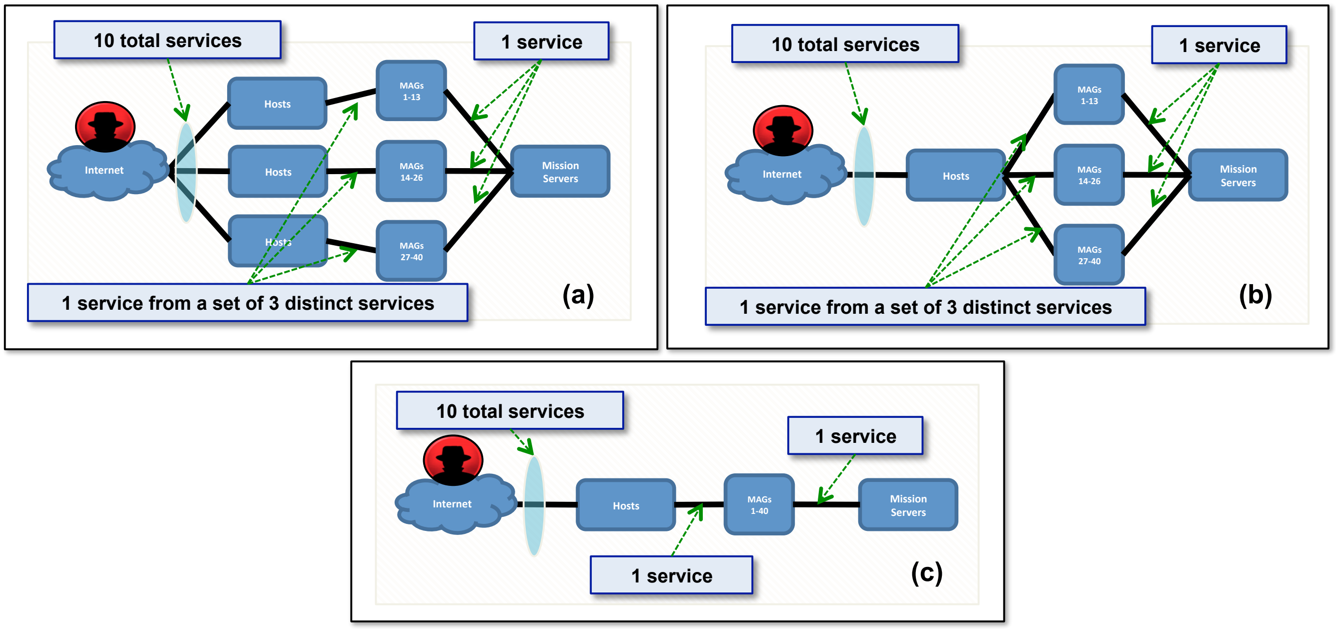


Fig. 9. System-generated segmentation architectures: (a) s_1, (b) s_2, and (c) s_3.

B. Results

As mentioned in Section IV-A, executing experiments on the testbed environment is quite resource intensive. Thus, for this study we execute just 7 iterations of our decision system, due to resource limitations. We report an intermediate solution generated by the system only when that solution is accepted by heuristic search component as the current solution as specified by Algorithm IV-C. For our experiments besides the initial segmentation architecture s_0 (Fig. 8), three new solution architectures, namely s_1, s_2, and s_3, were generated by the system and accepted as the current solution. We discuss results for architectures $s_{baseline}$ and s_0-s_3 below.

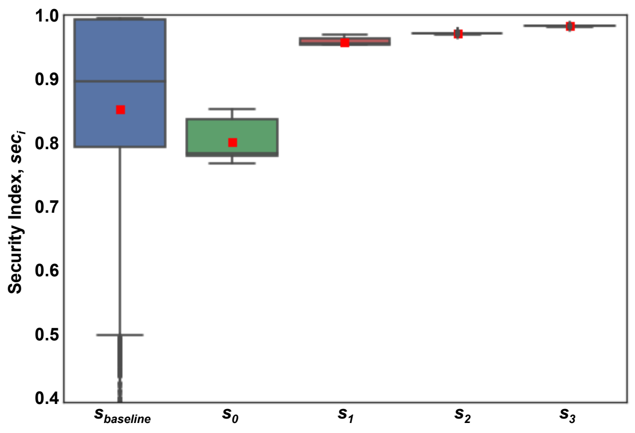


Fig. 10. System simulation component output for security index, sec_i, for (red squares show the respective mean for each architecture).

Fig. 9 depicts the system-generated architectures s_1, s_2, and s_3, respectively. As can be seen in the figures, the progression of generated architectures show merging of several enclaves from s_0 into the final, fairly simple segmentation given by s_3. It is important to note that a candidate architecture was generated in an iteration *after* s_3 that merged enclaves again resulting in an even more simplified architecture. However, this architecture was not accepted by the system due to its inferior measured effectiveness.

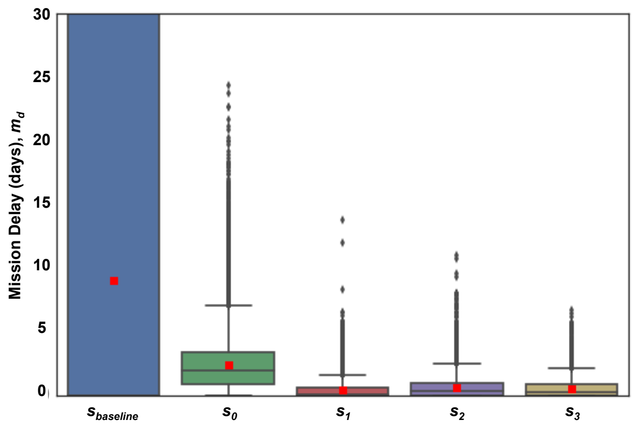


Fig. 11. System simulation component output for mission delay, m_d, (the red squares show the respective mean for each architecture).

Figs. 10 and 11 present plots[6] of outputs from the system's simulation component (described in Section IV-B) for architectures $s_{baseline}$ and s_0-s_3. Fig. 10 gives simulation component output from Monte Carlo simulation runs for security index sec_i while Fig. 11 gives the same for mission delay m_d (sec_i and m_d are described in Section IV-C).

[6]Boxplots are shown with median represented by the mid-line of a plot and mean represented by a red square.

Architecture	$seci_g$	md_g	m_g
s_0	0.037	0.657	0.34
s_1	0.185	0.951	0.56
s_2	0.285	0.91	0.61
s_3	0.29	0.999	0.65

TABLE I
SYSTEM EVALUATION VIA COMPUTED METRIC m_g INCLUDING COMPONENT SUB-MEASURES $seci_g$ AND md_g FOR EACH ARCHITECTURE.

As shown in Fig. 10, system-generated architectures s_1-s_3 yield marked improvements to the security relative to $s_{baseline}$ and s_0 with progressively higher mean and median sec_i and significantly lower variances of sec_i outputs. Interestingly, initial architecture s_0, which is based on SNF best practice recommendations (discussed in Section III), has somewhat lower mean and median sec_i relative to $s_{baseline}$. s_0 does, however, yield greatly reduced variance in sec_i outputs relative to $s_{baseline}$.

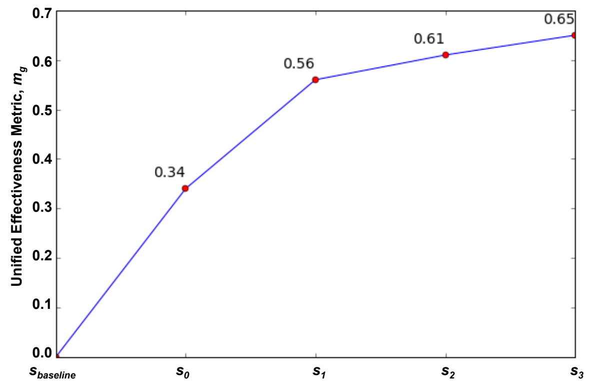


Fig. 12. System evaluation (via the unified effectiveness metric, m_g, computed by the statistical analysis component) for each architecture.

From Fig. 11, architectures s_1-s_3 yield marked improvement in mission performance in the form of reduced mission delay relative to $s_{baseline}$ and s_0. As seen in the figure, mean and median m_d outputs of system-generated architectures are less than the baseline and initial architectures and variance is also markedly reduced.

Fig. 12 provides the system's quantitative evaluation of each architecture (s_0-s_3). As discussed in Section IV-C, the system's statistical analysis component computes a unified effectiveness metric m_g that is used to measure the effectiveness of a candidate segmentation relative to the baseline architecture $s_{baseline}$. This metric measures effectiveness with respect to two aspects, security and mission performance, and can be viewed by security practitioners as a measure of the gain in effectiveness at the network scale due to the protection provided by the candidate architecture [25], [22]. Fig. 12 plots the computed m_g metric for all architectures. Note that the computed metric for $s_{baseline}$ ($m_g = 0.0$) is also provided for reference.

As displayed in Fig. 12, system-generated architectures progressively improve overall effectiveness with respect to security and mission performance relative to the initial architecture s_0. The first system-generated solution s_1 provides a rather large jump in effectiveness relative to s_0, from 34% to 56%, while successive solutions s_2 and s_3 build upon this gain to provide additional incremental improvements.

Table I breaks down the computed m_g metric into its component sub-metrics, $seci_g$ and md_g, representing aspects of security and mission performance, respectively, for each architecture. From the table, both security and mission performance are significantly improved by the system-generated architectures with the mission performance sub-metric indicating near-optimal effectiveness for final system solution architecture s_3.

These results indicate that optimal/near-optimal segmentation architectures for a given network environment and cyber threat can be automatically generated and highlight the efficacy of our computational-intelligence-based approach.

VI. CONCLUSION

This paper examines the cyber decision problem of how to select an appropriate network segmentation architecture with respect to security and mission performance for a given network environment. A novel cyber decision support system is built to recommend appropriate segmentation architectures. The system utilizes a computational intelligence approach that combines testbed experimentation, hierarchical simulation modeling, statistical analysis, and optimization techniques. A case study is performed using the prototype system to generate optimal/near-optimal architectures for a representative network environment under cyber attack.

Future work is focused on developing a simulation model to replace the testbed environment so that more partitioning scenarios can be examined as the resource cost of executing scenarios on the testbed is relatively high. In the current version of the prototype system, system components are not fully integrated and some manual action is required to start component executions. For the next version we plan to fully integrate all system components. Finally, we plan to investigate population-based techniques such as genetic algorithms, particle swarm, or grammatical evolution to improve the performance of the system's search for efficacious candidate architectures.

ACKNOWLEDGEMENT

This material is based upon work supported under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the U.S. Air Force.

REFERENCES

[1] U.S. Department of Homeland Security. (2015) Critical Infrastructure Sectors. [Online]. Available: http://www.dhs.gov/critical-infrastructure-sectors
[2] D. Barret, "U.S. Suspects Hackers in China Breached About four (4) Million People's Records, Officials Say," Wall Street Journal, June 2015.
[3] M. Lennon, "Hackers Hit 100 Banks in Unprecedented $1 Billion Cyber Heist: Kaspersky Lab," Security Week, Feb. 2016.

[4] "Google's Approach to IT Security," Google, Tech. Rep., 2012.

[5] "Top 10 Information Assurance Mitigation Strategies," Information Assurance Directorate, NSA, Tech. Rep., 2013.

[6] "Enterprise Security Best Practices," Microsoft, Tech. Rep., 2015. [Online]. Available: https://technet.microsoft.com/en-us/library/dd277328.aspx

[7] Center for Internet Security, "Critical Security Controls for Effective Cyber Defense Version 6.0," SANS Institute, Tech. Rep., October 2015.

[8] S. Institute. (2013) Sans critical security controls. [Online]. Available: http://www.sans.org/critical-security-controls/

[9] N. Reichenberg, "Improving Security via Proper Network Segmentation," Security Week, March 2014.

[10] S. Jajodia *et al.*, Eds., *Cyber Situational Awareness*, ser. Advances in Information Security. Springer US, 2010.

[11] M. Grimaila and A. Badiru, "A hybrid dynamic decision making methodology for defensive information technology contingency measure selection in the presence of cyber threats," *Operational Research*, vol. 13, no. 1, pp. 67–88, 2013.

[12] A. Hakansson and R. Hartung, "An infrastructure for individualised and intelligent decision-making and negotiation in cyber-physical systems," *Procedia Computer Science*, vol. 35, pp. 822–831, 2014.

[13] A. Mukhopadhyay *et al.*, "Cyber-risk decision models: To insure it or not?" *Decision Support Systems*, vol. 56, pp. 11–26, 2013.

[14] M. Gagnon *et al.*, "Towards Net-Centric Cyber Survivability for Ballistic Missile Defense," in *ISARCS*, 2010.

[15] J. Boyd, "Destruction and creation," U.S. Army Command and General Staff College, Tech. Rep., 1975.

[16] J. Chan *et al.*, "Introducing FIDO: Automated security incident response," Netflix, Inc., Tech. Rep., 2015.

[17] "Segregate Networks and Functions," Information Assurance Directorate, NSA, Tech. Rep., 2013.

[18] R. Gezelter, *Computer Security Handbook*. Wiley, 1995, ch. Security on the Internet.

[19] "Limiting Workstation-to-workstation Communication," Information Assurance Directorate, NSA, Tech. Rep., 2013.

[20] R. Ward and B. Beyer, "Beyondcorp: A new approach to enterprise security," *Login*, vol. 39, no. 6, pp. 6–11, 2014.

[21] Z. Michalewicz *et al.*, "Case Study: An Intelligent Decision-Support System," *IEEE Intelligent Systems*, vol. 20, no. 4, 2005.

[22] N. Wagner, C. Şahin *et al.*, "Quantifying the Mission Impact of Network-Level Cyber Defensive Mitigations," *Journal of Defense Modeling & Simulation (to appear)*, 2016.

[23] S. Yu *et al.*, "Malware propagation in large-scale networks," *IEEE Trans. on Knowledge and Data Engineering*, vol. 27, no. 1, pp. 170–179, 2015.

[24] (2015) Netlogo (version 5.1.0). [Online]. Available: http://ccl.northwestern.edu/netlogo/

[25] N. Wagner, C. c. Şahin, D. Hanson, J. Pena, E. Vuksani, and B. Tello, "Quantitative analysis of the mission impact for host-level cyber defensive mitigations," in *Proceedings of the 49th Annual Simulation Symposium*, ser. ANSS '16, April 2016, pp. 2:1–2:8.

[26] S. Kirpatrick *et al.*, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[27] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proc. of IEEE Intl. Conf. on Neural Networks*, 1995.

[28] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1996.