Session-level Adversary Intent-Driven Cyberattack Simulator

Martin Drašar
Institute of Computer Science
Masaryk University
Brno, Czechia
drasar@ics.muni.cz

Stephen Moskal, Shanchieh Yang
Department of Computer Engineering
Rochester Institute of Technology
Rochester, NY, USA
{sfm5015,jay.yang}@rit.edu

Pavol Zaťko
Faculty of Informatics
Masaryk University
Brno, Czechia
456131@mail.muni.cz

***Abstract*—Recognizing the need for proactive analysis of cyber adversary behavior, this paper presents a new event-driven simulation model and implementation to reveal the efforts needed by attackers who have various entry points into a network. Unlike previous models which focus on the impact of attackers' actions on the defender's infrastructure, this work focuses on the attackers' strategies and actions. By operating on a request-response session level, our model provides an abstraction of how the network infrastructure reacts to access credentials the adversary might have obtained through a variety of strategies. We present the current capabilities of the simulator by showing three variants of Bronze Butler APT on a network with different user access levels.**

***Index Terms*—DEVS, cybersecurity, adversary behavior, APT**

I. Introduction

Historically, the cybersecurity research of adversary behavior was reactive, rather then proactive. However, with proliferation of Advanced Persistent Threats (APTs), stealthy malware, and incorporation of machine learning into attacker control, there is a growing need to proactively study and develop adversary strategies. Currently, most of the research focuses on the impact of attackers' actions and not on the actions and attack strategies themselves, which is reflected in the state of available simulation approaches.

To address the lack of appropriate adversary-focused simulation tools, this paper brings two main contributions:

- Introduction of the concept and the implementation of a new simulation model, enabling evaluation of adversary behavior on the session level.
- Enabling integration of different attack models within one simulation engine, demonstrating its flexibility.

This paper is structured as follows. Section II provides a review of relevant state of the art. In Section III, we introduce the proposed simulation model and describe its implementation and integration with different attack models. Section IV presents our case study referencing to the Bronze Butler APT (BB) and its implementation in the simulator engine. In Section V, we evaluate the simulator engine by showing and reasoning different BB attack strategies using random and learning attackers. We conclude the paper and discuss future opportunities in Section VI.

This research was supported by ERDF "CyberSecurity, CyberCrime and Critical Information Infrastructures Center of Excellence" (No. CZ.02.1.01/0.0/0.0/16_019/0000822), and by US NSF Award # 1742789.

II. State of the Art

There exist several approaches to simulate the behavior of adversaries in networked systems. Some are designed specifically to test IDS systems, e.g., [1], while others expertly define static attack scenarios with little configurablility [2]. Some use real networks (virtual machines) [3], and the data is often tailored to a specific type of attack [4], [5]. In this section, we review three branches of approaches modelling the interactions between adversaries and the networked systems, which are relevant to the adversary simulation approach we introduce in this paper.

A. Attack Graphs

The attack graphs were first proposed by Swiler et al. [6] and are used to simulate steps an attacker can take within the infrastructure. They describe an abstracted network topology and show the nodes, paths and consequences of network attacks. Once an attack graph is constructed, it enables various tasks of network security analysis. The use of attack graphs is a widely researched topic including attack graph generation [7], [8], application scenarios [9], [10] and analytic methods [11]. To support automatic graph generation, tools such as MulVAL, NetSPA, or TVA were developed, as summarized in [12].

The downside to attack graphs is that they require the totality of knowledge about the target infrastructure and known vulnerabilities. While they model possible attacker actions, they are in effect centered on the defense and represent a vulnerability model rather than an attack model. They offer only limited options to analyze adversarial behavior.

B. Game Theoretic Approaches

Game theoretic approaches applied to cyber security are well researched and traditionally involve an attacker-defender model where the defender optimizes their defensive strategy for risk minimization [13]–[16] or maximize the uptime of network assets [17]–[19]. The games played rely on some amount of information sharing of various amount (complete or incomplete) where typically the defender observes the attacker and responds according to their objective function [20], whereas the amount of works focusing on the attacker is either focused on a specific attack type like DDoS [21] or relies on unspecified mission models [22]. Liang et al. mentions that the attacker-defender model specifically for impact assessments requires extensive data to understand the dynamic relationships between the attacker and defender, creating complex models that may or may not have a solution [20].

C. Simulation Approaches

Similar to game theory, the impacts of attacks and attackers can be realized through the use of configurable cyber attack simulation platforms. NeSSi2, an agent-based simulation platform by Grunewald et al. [23], models a packet-level description of a network with the primary focus on simulating the effects of distributed denial of service (DDoS) attacks. NeSSi2 models the effects of various worm behaviors and how worms propagate through a network. This technique proves to be useful in other contexts such as smart grid networks [24]. Moskal et al. [25], [26] presents a knowledge-based cyber attack simulator CASCADES, where the attacker's actions are determined by the "Attacker Behavior Model" (ABM) and the knowledge obtained about the target network through performing actions on the network. CASCADES focuses on a monte-carlo style approach to attack simulation and generates 1000's of plausible attack scenarios given the ABM and a detailed network description known as the Virtual Terrian (VT)

Several other simulation platforms exist and base mostly on a discrete event formalism, e.g., Chi et al. [27], Liljenstam et al. [28], Futoransky et al. [29], and Kuhl et al. [30]. These platforms offer synthetic network and attack emulation and evaluation. To overcome limited realism, emulators using virtual machines and integrated with offensive tools are available, such as DCAFE [31] and SVED [32]. It is also worth noting that others have experimented cybersecurity simulations based on general-purpose simulators such as OMNet++ [33].

III. Simulation model

In this section we present a new non-stochastic simulation model based on discrete event formalism, which enables synthetic network attack emulation and evaluation, dynamic selection of attack models, and integration with non-simulated IDS systems. This model thus occupies a space between various simulation models and tools described earlier.

The simulator implementing the model as well as the evaluation scripts can be freely downloaded from here: https://muni.cz/go/565e43

A. Model goals

This work aims at developing a cyberattack simulator that models the interactions between the progression in adversary intended outcomes and the network session level responses. Here, the session level means that the units of interaction between adversaries and the attacked environment are requests and responses, i.e., rough equivalent of TCP sessions. Such interactions are meant to maximize autonomy for both the attackers and defenders. The model described in this paper is a step towards the longer-term and broader goals to enable:

- lightweight simulation of multi-agent cybersecurity scenarios,
- integration of different attack models,
- non-stochastic simulation of interaction between attackers and defenders for in-depth analysis of attack strategies,
- rapid prototyping of attack and defense strategies,
- *smooth transition of simulated actors into emulated and real-world settings,
- *modelling of environments, which can be emulated in virtual environments using already provided data,
- *integration of simulation and emulation to remove the need to re-implement existing cyberdefense mechanisms.

Note that the last three goals are outside of the scope of this paper; yet they influence the current model design and development. Section VI briefly describes the relevant projects and activities beyond this paper linking to the long-term goals.

B. Model components

The simulation model adopts the message-based approach and consist of a number of components, which can be divided into four levels: environment, network, host, and logical.

1) Environment level: The environment level is a top-most layer of components, which are used for orchestration of particular scenario runs. There are two components present: *message* and *environment*.

Message is a unit of information exchanged between actors in the simulation. The message carries routing and statistical information, activity descriptions and actors' responses.

Environment keeps track of all simulation elements, manages interaction between these elements by passing messages, controls the simulation time and evaluates an impact of actors' activities. It is the only point of interaction between actors and the simulation.

2) Network level: The network level represents the topology of simulation. The components used mimic the components of the network, with some simplifications enabled by the conceptual level the simulation happens on. The components are: *nodes*, *firewalls*, *connections*, *routers*, and *sessions*.

Nodes represent physical or virtual machines. Each node is accessible from the outside via a set of network *ports*, which are a simplification of an Ethernet port, i.e., these *ports* have an IP address and can be uniquely identified (although no explicit MAC addressing is used).

Firewalls function as their real-world counterparts by controlling inbound and outbound *messages*. They implement an equivalent of a simplified filter table of iptables with source-destination filtering and default filtering policies.

Connections represent links between *ports* of particular *nodes*. *Messages* go through the *connections* and can be affected by connection properties.

Routers partition the networks. Unlike the real network settings, they are the only active switching elements. *Routers* control permeability between different networks and enable fine-grained control depending on both sources and destinations of *messages*.

Sessions represents a set of *connections* going through the *network*, which are not subject to routing policies in the intermediate *routers*. An example of a *session* is a VPN tunnel or a tunnel through several layers of NAT. The name stems from attacker taxonomy, where attacker exploits weaknesses in infrastructure to open sessions to or from their targets, which would be otherwise prohibited by intermediate active network elements.

3) Host level: The host level covers activities happening at a node. In addition to network *ports*, a *node* is modelled as a set of *services* representing running processes. A *node* does not define an OS, as this is instead expressed as a set of *services* representing OS functions required for simulation. *Services* come in two variants, which are the components of the host level: *active services* and *passive services*.

Active services can initiate the communication with their surroundings by sending *messages* through the *environment*. They also processes inbound *messages* and react according to their programmed behavior. Thus, they must understand the semantics of the incoming *messages*. Typical example of *active services* are attackers and defenders, i.e., actors whose behavior is the focus of a simulation.

Passive services, on the other hand, do not initiate a communication. Inbound *messages* are instead evaluated by the *environment* based on the definition of a *passive service*. The definition contains service name, version, ability to create sessions, locality, etc. *Passive services* are used to create a believable environment for the *active services*, while not pushing the burden of implementation on the user of the simulator.

4) Logical level: The logical level represents the activity domain for active actors, and relations between scenario elements and scenario goals. There are four components: *data*, *authorizations*, *exploits*, and *actions*.

Data represent units of information, which may be interesting to an attacker, such as trade secrets or employee records. Obtaining data does not help an attacker with the actual attack, but they may be essential to reaching the goal of the given scenario.

Authorizations encode the ability of actors to access particular *services* or *data*. They can be defined in the scenario configuration or they can be created as a result of actors' activities, e.g., new *authorization* resulting from successful privilege escalation. Terminology-wise, they conflate both authorization and authentication for the sake of simplicity.

Exploits represent mechanisms to abuse vulnerabilities of particular *services* and are tied to the name and the version of a *service*. To enable better machine reasoning about *exploits* and their effects, they are categorized by their effect and locality and allow only limited parametrization to create a bounded exploit domain from which an attacker can choose. The *exploits* are expected to map to real-life exploits, such as those listed through CVE.

Actions represent the type of activity of actors. They can comprise anything from getting the simulation time to launching a DDoS attack. They are a mean to express an attack model within a simulator. In this case, we understand the attack model as an abstraction of activities an attacker can perform. The actions are then the elements of the actions space defined by a particular model. One such model, which we used for our simulation evaluation is presented in the following text.

C. Attack model

Defining the action space of the adversary is a particularly challenging task for cyber-attack simulators as the action space is effectively infinite, constantly expanding, and extremely diverse in the types of actions that can be performed. Modelling each vulnerability in the simulator is time consuming and unsustainable, so we choose to represent the action space as an abstraction of the objective or intent of an attacker given the simulated attack stage of the attacker. The Action-Intent Framework (AIF) [34] is a cyber-attack action classification framework where the focus is to describe attack actions with respect to the intended objective of performing a specific action such as: information discovery, privilege escalation, data exfiltration, etc. The AIF differentiates itself from other attack descriptions by providing significantly more detail then typical Cyber Attack Kill Chains while finding a middle ground between the highly detailed MITRE ATT&CK by remaining network and service agnostic.

The AIF is broken up into two layers of abstraction: the Macro Action-Intent States (Macro-AIS) describe the effect of the actions at a high-level such as reconnaissance or destroy information, whereas the Micro Action-Intent States (Micro-AIS) describe the method used to achieve the corresponding Macro-AIS. An example is a brute-force credential access Micro-AIS for the privilege escalation Macro-AIS. We use the Micro-AIS to represent our desired simulated attack scenarios and as a method to select simulated actions given the network topology and the services running on the network. Given the session-based approach of our proposed simulation architecture the abstracted model of the attacker's process will allow for attack scenarios to be quickly created and then applied to the network topology without the need for detailed exploit definitions. In Section IV we demonstrate how a known description of a real cyber-attack can be described using the AIF and then we use that description as the driving force of our simulation engine.

D. Integration of Components and Simulation Execution

Fig. 1 illustrates the component relations and how messages traverse between components. When a simulation run begins, all active services are executed. The services produce messages, which are inserted into environment queues and distributed on a hop-by-hop basis to their intended targets. Thus the message transport mimics the packet transport over a network. The messages trigger component responses, simulating the actions and responses when the network is attacked.

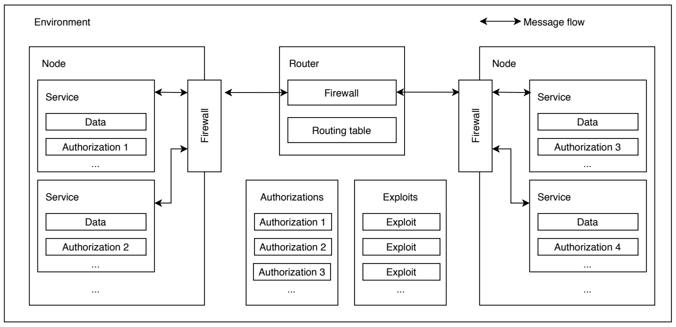

Fig. 1. Diagram of model components

A message passing a connection can arrive into a router, active service, or passive service. Each component computes a simulation-relative processing time, which models link delays and processing complexity. Arriving into a router the message can either be forwarded or dropped based on firewall and routing rules. If the message arrives in a passive service, it is evaluated by the environment, which has an implementation of attack models' semantics (in our case the AIF). The model's implementation decides on the response given the action (in our case a Micro-AIS) and message and passive service parameters. If the message arrives into an active service, the service decides on the response based on the observable properties of the message. Note that active services do not have access to action as it would be equivalent to knowing an attacker's intent just by looking at the packet and would bypass the hard problem of cybersecurity analysis.

IV. Case study: Bronze Butler APT

To present the expressive power of the simulation engine and to show how it can be used to reason about attackers' and defenders' abilities, we consider various scenarios simulating Bronze Butler APT breaching into an organization with the goal of data theft. We model the Bronze Butler APT with the Micro-AIS described in Sec. III-C and compare it to the MITRE ATT&CK framework [35]. We then present the common network topology and an attack graph modelling the three variants of the breach scenario, which we further discuss.

A. Bronze Butler in Micro-AIS

Bronze Butler is a well documented Chinese hacker group infamous for targeting Japanese critical infrastructures between 2012–2017. Bronze Butler has been reported to use a variety of spearphishing techniques, remote access exploits, and web-based zero-day malware to target high profile executives to obtain sensitive business strategies and sales information. Depending on the target, Bronze Butler employed two techniques to gain initial access to their target: 1) a spearphishing email to an executive with a malicious attachment [36] or exploited VPN services to gain access to the target network [37]. The end goal of Bronze Butler is to exfiltrate critical business or user information through the use of file-share servers.

We choose to use Bronze Butler for our case study as the techniques employed by Bronze Bulter are sufficiently complex to demonstrate the capabilities of our simulation engine exhibiting distinct behaviors that are well represented in attack action descriptions such as MITRE ATT&CK and the AIF.

Bronze Butler is comprised of a team of highly skilled attackers. However, we abstract the behaviors of Bronze Butler as a single entity and represent the behaviors as a set of Micro-AIS to represent their scenario in our simulation engine. Using the threat reports from SecureWorks [37] and technique description from MITRE ATT&CK [38], we map attack action evidences to a corresponding Micro-AIS to capture some of the key behavioral properties of Bronze Butler that will be used as the basis of our simulation experiments.

The Table I summarizes Bronze Butler capabilities in terms of MITRE ATT&CK, the AIF, and the simulation engine. The table demonstrates that the simulator using Micro-AIS as an attack model is able to simulate most of Bronze Butler behavior, with the exception of user interaction and host-level interaction, by means of simulated actions and their parameters.

B. Network Topology and Access Control

To emulate the various scenarios how Bronze Butler can penetrate into a network, we prepare a small-scale network as depicted in Fig. 2. The topology is partitioned into four logical segments, separated by routers with firewalls. The first segment is outside of the organization and represent the attacker. Note that Bronze Butler may compromise the organization's partner in a different network domain. For simplicity, we consider all external sources in the same network. The second segment is the DMZ with a Web server, VPN server and an Email server. Each server has two network interfaces, one accessible from the outside and the other accessible from the inside. The third segment contains the desktop machines of an employee and a CTO. Machines in this segment can access the DMZ and can only be accessed from the SRV segment. The fourth is the SRV segment containing an API gateway to the organization's services, a database server (DB), and a domain controller (DC). This segment is accessible from the DMZ and from the CTO's PC. The API gateway is also accessible via a VPN tunnel from the outside. The simulated machines are populated with various services and access credentials to enable the attacker multiple paths through the network.

To limit the scope of the demonstration and to better reason about the simulating engine, the modelled network does not contain any active defenses. The defenses are passive and based on network and host access control. While the resulting configuration cannot be considered secure, it will be shown later that it is hardened enough to resist inept attack attempts.

C. Attack Graph and Scenario Variations

The combination of the network configuration, the deployed services, and the access credentials on the simulated hosts,

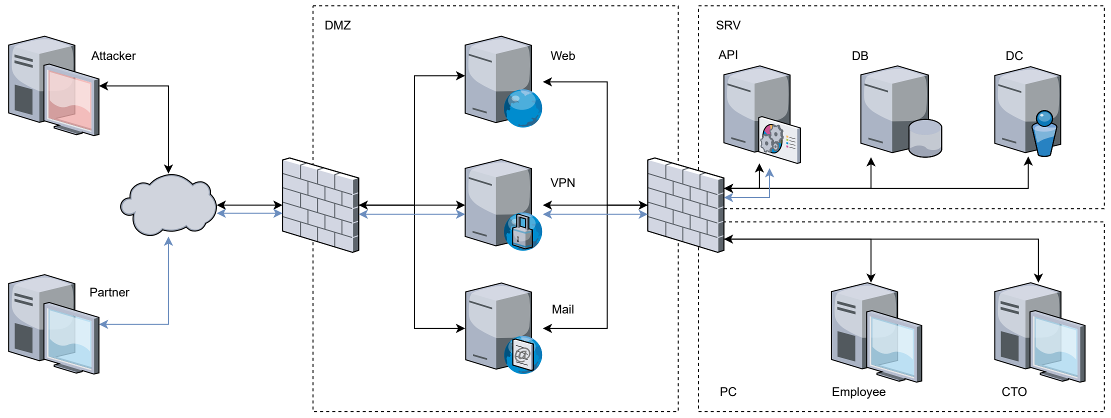

Fig. 2. Scenario network topology

gives the Bronze Butler multiple ways to achieve the ultimate goal of exfiltrating data from the DB server. We have manually crafted an attack graph from the total knowledge of the scenario. The attack graph covering all shortest paths is depicted in the Fig. 3. To preserve clarity of the graph, the paths that do not lead to the goal are excluded; these paths, however, can be explored by the attacker in the simulation and can greatly prolong the attack duration. This can lead to a counter-intuitive behavior of attacker with elevated privileges, which is later discussed in the Section V-C.

Despite the possible variations, each path in the attack graph starts with a successful spearphishing attempt, because it is the predominant entry-point for the Bronze Butler APT. Note that the current simulation concentrates on modeling the interactions between the attacker and the network, and does not include the exact user interactions with the attacker phishing emails, for example. For each scenario variant the attacker begins with the simulated artifact of the phishing attempt - an opened session to the target machine.

There are three main variants of the scenario, which differ by the successfully spearfished machine. The first one represent a successful attack on the CTO of the organization, who can access the DB server and also has the necessary credentials to extract the goal data from the database. The second one represents a successful attack on the partner, who has an API access via VPN into the SRV segment. The attacker has to use the API server as a stepping stone to get access to the domain controller and follow by forging a golden Kerberos ticket, which is then abused to get access to the data. The most complicated variant begins with a successful attack on an employee. The attacker has to go through the Web server in DMZ, discover domain controller credentials there and continue the attack inside the SRV segment as in the previous case. Those three variants require the attacker to execute 2(3), 6(7),

	ATT&CK Technique	Technique example	Micro AIS
Simulated actions	T1087	used net user /domain to identify account information.	information discovery
	T1088	malware xxmm contains a UAC bypass tool for privilege escalation.	user privilege escalation
	T1003	used various tools to perform credential dumping.	information discovery
	T1005	exfiltrated files stolen from local systems.	data exfiltration
	T1039	exfiltrated files stolen from file shares.	data exfiltration
	T1140	downloads encoded payloads and decodes them on the victim.	lateral movement
	T1083	collected a list of files from the victim and uploaded it to its C2 server, and then created a new list of specific files to steal.	data exfiltration
	T1107	uses command to delete the RAR archives after they have been exfiltrated.	data destruction
	T1097	created forged Kerberos Ticket Granting Ticket (TGT) and Ticket Granting Service (TGS) tickets to maintain administrative access.	root privilege escalation
	T1060	used a batch script that adds a Registry Run key to establish malware persistence.	lateral movement
	T1105	used various tools to download files, including DGet (a similar tool to wget).	lateral movement
	T1018	use ping and Net to enumerate systems.	host discovery
	T1053	used at and schtasks to register a scheduled task to execute malware during lateral movement.	lateral movement
	T1102	MSGET downloader uses a dead drop resolver to access malicious payloads.	lateral movement
	T1113	used a tool to capture screenshots.	information discovery
	T1124	used net time to check the local time on a target system.	-
	T1210	used a CVE-2016-7836 to exploit VPN connection	command and control
Action parametrization	T1024	used a tool called RarStar that encodes data with a custom XOR algorithm when posting it to a C2 server.	-
	T1002	compressed data into password-protected RAR archives prior to exfiltration.	-
	T1059	uses the command-line interface.	-
	T1132	encode data with base64 when posting it to a C2 server.	-
	T1022	compressed and encrypted data into password-protected RAR archives prior to exfiltration.	-
	T1086	used PowerShell for execution.	-
	T1064	used VBS, VBE, and batch scripts for execution.	-
	T1071	used HTTP for C2.	-
	T1032	used RC4 encryption (for Datper malware) and AES (for xxmm malware) to obfuscate HTTP traffic.	-
User interaction	T1189	compromised three Japanese websites using a Flash exploit to perform watering hole attacks.	-
	T1193	used spearphishing emails with malicious Microsoft Word attachments to infect victims.	-
	T1203	exploited Microsoft Word vulnerability CVE-2014-4114 for execution.	-
	T1204	attempted to get users to launch malicious Microsoft Word attachments delivered via spearphishing emails.	-
N/A	T1009	included "0" characters at the end of the file to inflate the file size in a likely attempt to evade anti-virus detection.	-
	T1036	given malware the same name as an existing file on the file share server to cause users to unwittingly launch and install the malware on additional systems.	-

TABLE I
BRONZE BUTLER APT CAPABILITIES IN TERMS OF MITRE ATT&CK FRAMEWORK, THE AIF, AND THE SIMULATION ENGINE.

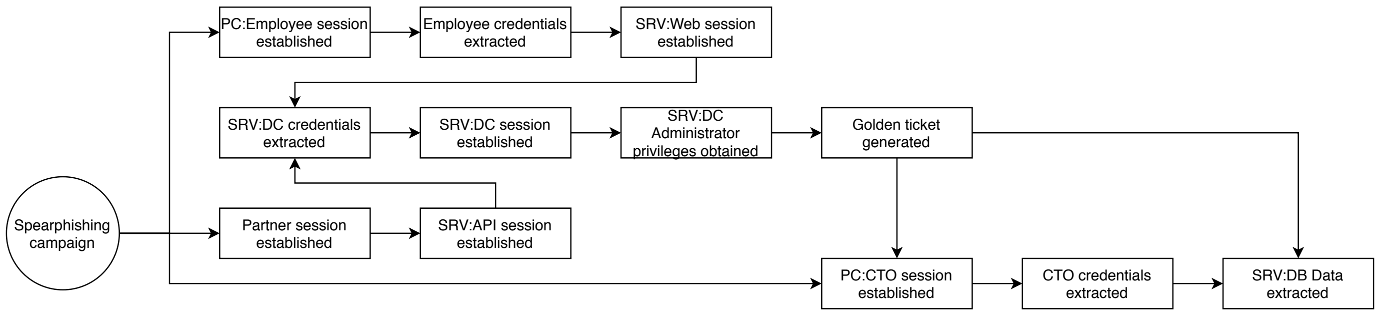
PC:Employee session established
Employee credentials extracted
SRV:Web session established
SRV:DC credentials extracted
SRV:DC session established
SRV:DC Administrator privileges obtained
Golden ticket generated
Spearphishing campaign
Partner session established
SRV:API session established
PC:CTO session established
CTO credentials extracted
SRV:DB Data extracted

Fig. 3. Scenario attack graph

and 7(9) appropriate actions respectively to achieve the goal. The numbers above represent the minimal number of steps in the attack graph from the given start to the terminal node. The numbers in parentheses also include the reconnaissance steps, which would be necessary in a real-world setting. Note that each step could actually be one or many activities in simulation, especially in case of reconnaissance.

V. Evaluation and discussion

In this section, we introduce three different implementations of Bronze Butler, each representing a different attack strategy; namely a scripted, a random, and a learning attackers. We deploy these attackers into the simulator and let them attempt the three scenario variants. By analyzing the results and extracting insights into the attack strategies, we demonstrate how the simulator can be used to reason about particular attack strategies and about attackers' behavior.

A. Scripted attacker

The scripted attacker represents the idealist situation and follows the shortest path in the attack graph from each of the three starting points, as depicted in Fig. 3. This attacker type is implemented as omniscient, i.e. knowing the topology of the infrastructure and all system weaknesses, so it does not need to perform reconassance tasks. Therefore, its number of actions is the lower bound on actions needed to finish each scenario variation. For the three scenario variants (CTO, VPN, and Employee), the abstract actions are:

- **CTO**: Acquire CTO credentials and Exfiltrate data from the DB server (2 total).
- **VPN**: Access infrastructure via VPN exploit, Acquire DC credentials, Establish session to the DC, Get root access to the DC, Get the golden ticket, Exfiltrate data from the DB server (6 total).
- **Employee**: Acquire Employee credentials, Establish session to the Web server, Acquire DC credentials, Establish session to the DC, Get root access to the DC, Get the golden ticket, Exfiltrate data from the DB server (7 total).

The total counts shown above are the minimal steps for each variant and the attacker cannot take a shorter path due to the lack of access or authorization. It is not surprising the CTO case presents much shorter path than the other two in this idealist setting. In reality, however, getting the CTO credential might be harder to achieve than getting such from the large population of employees, especially if the CTO is well versed in cybersecurity hygiene.

Note that the minimal step counts will be used to calculate the so called normalized attack difficulty, i.e., an average number of attack actions between advancing to a next step in the attack graph. This will enable a comparison of efforts and thus difficulty for the attacker to achieve the ultimate goal in each of the CTO, VPN, and Employee cases, respectively, when the idealistic assumption is lifted.

B. Random attacker

The random attacker, as the name implies, selects random actions from the entirety of the action space until the goal is reached or the number of actions in a run exceeds a given threshold. The network being considered seem to be small scale but there are a significant number of sessions and accesses for each attack step. While not resembling real-world attackers, the random attacker provides an important benchmark to evaluate:

- correctness of the system behavior through fuzzying of simulation environment.
- complexity and effect of different scenarios.
- efficiency of different attack strategies.

Fig. 4 illustrates how the random attacker is used to evaluate the complexity of the action space and the impact of different strategies applied to the underlying CTO scenario variant. It shows the number of actions needed to reach the goal over 100 runs. Note that the CTO scenario requires a sequence of only two correct actions to reach the goal. Yet it can take a significant number of actions to reach the goal due to the large action space the CTO has access to. We consider the following random attacker strategies (rules followed by the random attacker) to reduce the action space:

- **Known services**: the attacker targeted only services it knew were running on particular hosts.
- **Live machines**: the attacker did not try again a combination of session and a target if it received a network failure.

Fig. 4 provides insights to the impact of particular strategies as well as to the usage of randomized attackers to test simulated cyberattack scenarios. To begin with, having a random

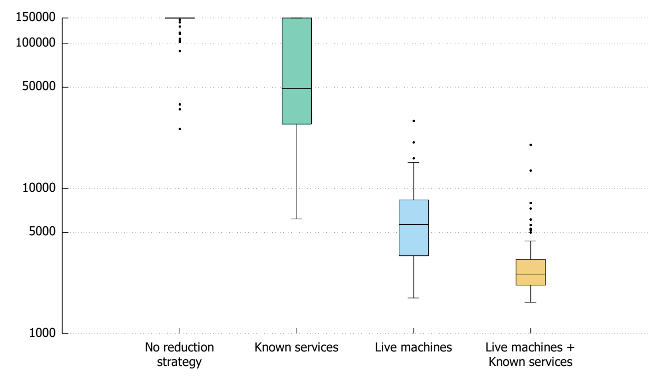

Fig. 4. Number of actions needed to reach the goal when simulating different random CTO attacker strategies and their combinations. Each box-plot represents 100 runs with up to 150,000 actions.

attacker utilize the entire action space without any effort to reduce it is pointless. The current scenario variant had at least 1.5 million possible actions which could double with each successful session-establishing action not leading to the goal. Therefore, it is not surprising that the majority of runs ended by reaching the threshold and failing the goal. Focusing on known services does have an impact which is proportional to the average number of services on each host and the totality of services in the scenario. Focusing only on live hosts has the biggest impact and is the factual prerequisite to running more complex scenarios. Finally, compounding the two strategies reduces the action space even further.

The general insight is that randomized attackers are a viable concept for evaluation and benchmarking simulations, but they require aggressive tactics to reduce the possible action space, especially for more complex scenarios with a lot of different host, services, exploits, and authorization mechanisms. For the benchmark in the next section, we opted-in for the live-hosts strategy as it ensures steepest reduction in activity with minimal interference with random selection process.

C. Learning attacker

The learning attacker treats the simulated scenarios as a multi-armed bandit problem and employs the UCB1 [39] algorithm to reach the scenario goal. The attacker keeps track of the uses of targets, actions, sessions, etc. By assigning specific reward values for successful and unsuccessful activities, it is able to compute the upper confidence bound and choose the next action accordingly. To prevent nonsensical ordering of actions, such as data deletion before extraction, it uses fixed action priorities. Essentially, the learning attacker act more intelligently by selecting the viable targets and sessions. This serves as a step closer to mimic real-world attacks and is used to compare to the random attacker on the actions needed to reach the goal when interacting with the network components. We formulate the following two hypotheses:

- On average, the learning attacker will require considerably less actions to finish the scenario than a random attacker employing the *live machines* strategy.
- The normalized attack difficulty (NAD)[1] of the learning attacker will decrease over time, whereas for random attacker it will remain constant.

The first hypothesis is based on the learning attacker's ability to gradually add possible targets, rather than removing them from the entire target space as the random attacker employing *live machine* strategy does. The second hypothesis is based on the random attacker not understanding any relations between actions and their consequences and selecting actions by chance, whereas the learning attacker learns the appropriate actions over time.

Figures 5 and 6 show the raw and normalized number of actions required to finish each of the scenario variants for the random and the learning attackers. For both attackers, each scenario variant was run 1000 times. It is apparent that the first hypothesis holds and even a cursory glance on the graphs shows that the learning attacker's strategy is between one and two orders of magnitude more efficient.

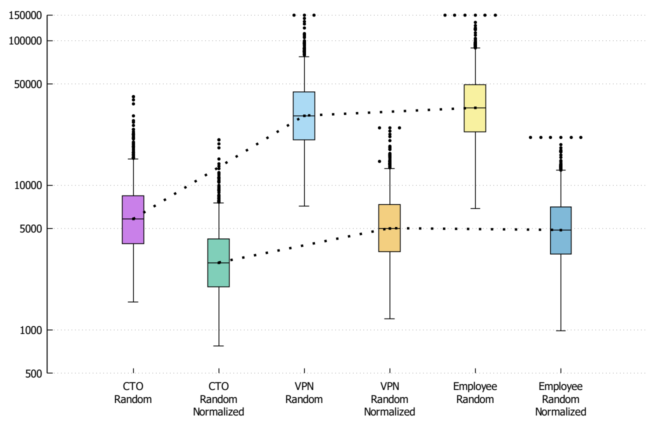

Fig. 5. Number of actions for random attacker to finish under each scenario.

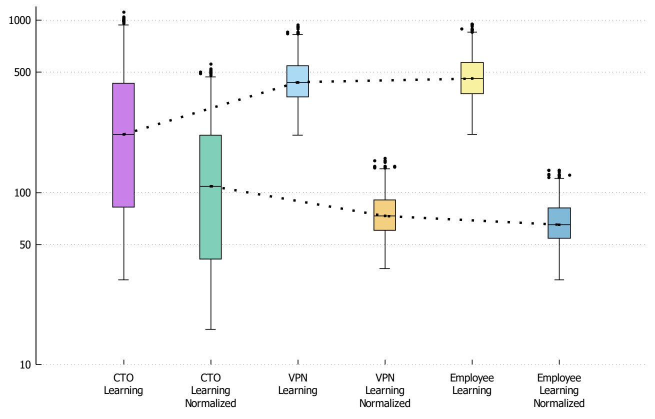

Fig. 6. Number of actions for learning attacker to finish under each scenario.

[1]NAD is the simulated attack actions count normalized by the minimal attack steps under each of the CTO, VPN, and Employee variant.

The second hypothesis also holds, but gives additional insights based on two observations from the plots. The first can be seen in Fig. 5, where the median NAD for the CTO variant is approximately $1/2$ of the other two variants. The reason is that those two variants require the attacker to acquire a new session via exploitation and this new session doubles the attacker's action space. The second observation can be seen in Fig. 6. Event though the median NAD decreases over time as expected, the CTO variant displays unexpectedly large variation and in many cases the easier scenario took longer to finish than the other more complex variants. This counter-intuitive behavior is rooted in that the attacker under the CTO variant has visibility to the entirety of the infrastructure and is free to explore unfruitful branches of the attack graph. This has led to more possibilities and thus large variations where the CTO variant can have very small or very high NAD's comparing to the other two variants, which are constrained in what the attackers have access to.

VI. Summary and Future Work

This paper introduced a new event driven simulation tailored to analyzing and evaluating adversarial behavior. The model fills a gap between different cybersecurity simulation works and tools by focusing on attackers' intent and actions, by enabling integration of different attack models, and by operating on a session level. The model and its implementation were evaluated with three variants of Bronze Butler APT (BB). These variants of attacking agents possess the BB's capabilities and launching them against a simulated corporate infrastructure with insecure configuration. We simulated random and learning attackers for each of the three variants, and assessed the efforts needed in each case to complete the attack goal. Our results showed not only insights on how to realize session level cyber adversary simulation, but also how different levels of accesses (CTO, VPN, and Employee) can lead to orders of magnitude differences in the number of actions needed to retrieve critical data.

The presented simulation engine is the first stage in a coordinated effort to create an infrastructure for development of autonomous cybersecurity agents, spearheaded by the NATO IST-152 research group [40]. In the near term, the simulation model will be used as a basis for automated generation of cybersecurity scenarios for both the agent training and human hands-on training. For the latter, support for transition of the infrastructure from simulator description to an emulation environment, such as KYPO [41], and the ability to use agent algorithms to drive real-world attacking platforms, such as Cryton [42], will be used.

References

[1] F. Erlacher and F. Dressler, "How to test an ids?: Genesids: An automated system for generating attack traffic," in *Proceedings of the 2018 Workshop on Traffic Measurements for Cybersecurity*. ACM, 2018, pp. 46–51.

[2] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Towards generating real-life datasets for network intrusion detection." *IJ Network Security*, vol. 17, no. 6, pp. 683–701, 2015.

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *Proceedings of the International Conference on Information Systems Security and Privacy*, 2018, pp. 108–116.

[4] S. Alzahrani and L. Hong, "Generation of ddos attack dataset for effective ids development and evaluation," *Journal of Information Security*, vol. 9, no. 04, p. 225, 2018.

[5] M. Cermak, T. Jirsik, P. Velan, J. Komarkova, S. Spacek, M. Drasar, and T. Plesnik, "Towards provable network traffic measurement and analysis via semi-labeled trace datasets," in *Proceedings of 2018 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2018, pp. 1–8.

[6] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proceedings of the 1998 workshop on New security paradigms*, 1998, pp. 71–79.

[7] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *Journal of Information Security and Applications*, vol. 29, pp. 27–56, 2016.

[8] X. Ou and A. Singhal, "Attack graph techniques," in *Quantitative Security Risk Assessment of Enterprise Networks*. Springer, 2012, pp. 5–8.

[9] Z. Ye, Y. Guo, C. Wang, and A. Ju, "Survey on application of attack graph technology," *Journal of Communications*, vol. 38, no. 11, pp. 121–132, 2017.

[10] V. Shandilya, C. B. Simmons, and S. Shiva, "Use of attack graphs in security systems," *Journal of Computer Networks and Communications*, vol. 2014, 2014.

[11] J. Zeng, S. Wu, Y. Chen, R. Zeng, and C. Wu, "Survey of attack graph analysis methods from the perspective of data and knowledge processing," *Security and Communication Networks*, vol. 2019, pp. 1–16, 12 2019.

[12] S. Yi, Y. Peng, Q. Xiong, T. Wang, Z. Dai, H. Gao, J. Xu, J. Wang, and L. Xu, "Overview on attack graph generation and visualization technology," in *2013 International Conference on Anti-Counterfeiting, Security and Identification (ASID)*, 2013, pp. 1–6.

[13] C. Xiaolin, T. Xiaobin, Z. Yong, and X. Hongsheng, "A markov game theory-based risk assessment model for network information system," in *Proceedings of International Conference on Computer Science and Software Engineering*, vol. 3. IEEE, 2008, pp. 1057–1061.

[14] K. C. Nguyen, T. Alpcan, and T. Basar, "Security games with incomplete information," in *Proceedings of IEEE International Conference on Communications*. IEEE, 2009, pp. 1–6.

[15] B. Wang, J. Cai, S. Zhang, and J. Li, "A network security assessment model based on attack-defense game theory," in *Proceedings of 2010 International Conference on Computer Application and System Modeling (ICCASM)*, vol. 3. IEEE, 2010, pp. V3–639.

[16] K. Chung, C. A. Kamhoua, K. A. Kwiat, Z. T. Kalbarczyk, and R. K. Iyer, "Game theory with learning for cyber security monitoring," in *Proceedings of 2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 2016, pp. 1–8.

[17] K. Sallhammar, B. E. Helvik, and S. J. Knapskog, "Towards a stochastic model for integrated security and dependability evaluation," in *Proceedings of First International Conference on Availability, Reliability and Security*. IEEE, 2006, pp. 8–pp.

[18] H. Wang, Y. Liang, and X. Liu, "Stochastic game theoretic method of quantification for network situational awareness," in *Proceedings of 2008 International Conference on Internet Computing in Science and Engineering*. IEEE, 2008, pp. 312–316.

[19] K. Sallhammar, B. E. Helvik, and S. J. Knapskog, "A framework for predicting security and dependability measures in real-time," *International Journal of Computer Science and Network Security*, vol. 7, no. 3, pp. 169–183, 2007.

[20] X. Liang and Y. Xiao, "Game theory for network security," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 472–486, 2013.

[21] M. Fallah, "A puzzle-based defense strategy against flooding attacks using game theory," *IEEE transactions on Dependable and Secure Computing*, vol. 7, no. 1, pp. 5–19, 2010.

[22] S. Musman and A. Turner, "A game theoretic approach to cyber security risk management," *The Journal of Defense Modeling and Simulation*, vol. 15, no. 2, pp. 127–146, 2018.

[23] D. Grunewald, M. Lützenberger, J. Chinnow, R. Bye, K. Bsufka, and S. Albayrak, "Agent-based network security simulation," in *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, ser. AAMAS '11. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2011, p. 1325–1326.

[24] J. Chinnow, J. Tonn, K. Bsufka, T. Konnerth, and S. Albayrak, “A tool set for the evaluation of security and reliability in smart grids,” in *Smart Grid Security*, 01 2013, pp. 45–57.

[25] S. Moskal, B. Wheeler, D. Kreider, M. E. Kuhl, and S. J. Yang, “Context model fusion for multistage network attack simulation,” in *Proceedings of Military Communications Conference (MILCOM), 2014 IEEE*. IEEE, 2014, pp. 158–163.

[26] S. Moskal, S. J. Yang, and M. E. Kuhl, “Cyber threat assessment via attack scenario simulation using an integrated adversary and network modeling approach,” *The Journal of Defense Modeling and Simulation*, vol. 15, no. 1, pp. 13–29, 2018.

[27] S.-D. Chi, J. S. Park, K.-C. Jung, and J.-S. Lee, “Network security modeling and cyber attack simulation methodology,” in *Proceedings of the 6th Australasian Conference on Information Security and Privacy*, ser. ACISP ’01. Berlin, Heidelberg: Springer-Verlag, 2001, p. 320–333.

[28] M. Liljenstam, J. Liu, D. Nicol, Y. Yuan, G. Yan, and C. Grier, “Rinse: the real-time immersive network simulation environment for network security exercises,” in *Workshop on Principles of Advanced and Distributed Simulation (PADS’05)*, 2005, pp. 119–128.

[29] A. Futoransky, F. Miranda, J. Orlicki, and C. Sarraute, “Simulating cyber-attacks for fun and profit,” in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST, 5 2010.

[30] M. E. Kuhl, M. Sudit, J. Kistner, and K. Costantini, “Cyber attack modeling and simulation for network security analysis,” in *2007 Winter Simulation Conference*, 2007, pp. 1180–1188.

[31] G. Rush, D. R. Tauritz, and A. D. Kent, “Dcafe: A distributed cyber security automation framework for experiments,” *2014 IEEE 38th International Computer Software and Applications Conference Workshops*, pp. 134–139, 2014.

[32] H. Holm and T. Sommestad, “Sved: Scanning, vulnerabilities, exploits and detection,” in *MILCOM 2016 - 2016 IEEE Military Communications Conference*, 2016, pp. 976–981.

[33] A. Varga, *OMNeT++*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 35–59. [Online]. Available: https://doi.org/10.1007/978-3-642-12331-3_3

[34] S. Moskal and S. J. Yang, “Cyberattack action-intent-framework for mapping intrusion observables,” 2020.

[35] B. E. Strom, J. A. Battaglia, M. S. Kemmerer, W. Kupersanin, D. P. Miller, C. Wampler, S. M. Whitley, and R. D. Wolf. (2017) Finding cyber threats with ATT&CK-based analytics.

[36] J. DiMaggio, “Tick cyberespionage group zeros in on japan,” https://muni.cz/go/de26e1, 2016, [Online; accessed 13-May-2020].

[37] Counter Threat Research Team, “Bronze butler targets japanese enterprises,” https://www.secureworks.com/research/bronze-butler-targets-japanese-businesses, 2017, [Online; accessed 13-May-2020].

[38] MITRE ATT&CK Team, “Bronze butler,” https://attack.mitre.org/groups/G0060/, 2019, [Online; accessed 13-May-2020].

[39] M. M. Drugan and A. Nowe, “Designing multi-objective multi-armed bandits algorithms: A study,” in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 2013, pp. 1–8.

[40] P. Theron, A. Kott, M. Drašar, K. Rzadca, B. LeBlanc, M. Pihelgas, L. Mancini, and F. de Gaspari, *Reference Architecture of an Autonomous Agent for Cyber Defense of Complex Military Systems*. Cham: Springer International Publishing, 2020, pp. 1–21. [Online]. Available: https://doi.org/10.1007/978-3-030-33432-1_1

[41] P. Čeleda, J. Vykopal, V. Švábenský, and K. Slavíček, “Kypo4industry: A testbed for teaching cybersecurity of industrial control systems,” in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, ser. SIGCSE ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1026–1032. [Online]. Available: https://doi.org/10.1145/3328778.3366908

[42] I. NUTÁR, “Automation of complex attack scenarios [online],” Master’s thesis, Masaryk University, Faculty of Informatics, Brno, 2017. [Online]. Available: https://is.muni.cz/th/cry3j/