Closing the Gap with APTs Through Semantic Clusters and Automated Cybergames

Steven Gianvecchio(✉), Christopher Burkhalter, Hongying Lan, Andrew Sillers, and Ken Smith

The MITRE Corporation, McLean, VA 22102, USA
{gianvecchio,cburkhalter,hlan,asillers,kps}@mitre.org

Abstract. Defenders spend significant time interpreting low-level events while attackers, especially Advanced Persistent Threats (APTs), think and plan their activities at a higher strategic level. In this paper, we close this semantic gap by making the attackers' strategy an explicit machine-readable component of intrusion detection. We introduce the concept of *semantic clusters*, which combine high-level technique and tactic annotations with a set of events providing evidence for those annotations. We then use a fully automated cybergaming environment, in which a red team is programmed to emulate APT behavior, to assess and improve defensive posture. Semantic clusters both provide the basis of scoring these cybergames and highlight promising defensive improvements. In a set of experiments, we demonstrate effective defensive adjustments which can be made using this higher-level information about adversarial strategy.

Keywords: Intrusion detection · Advanced Persistent Threats · Cyber forensics · Semantics · Tactic techniques procedures · Cybergaming

1 Introduction

There is an important semantic gap between network defenders and their adversaries. Attackers operate at the level of strategy and tactics, focused on discovering targets, and deploying various "kill chain" [7] tactics to reach their goals. Defenders, however, spend significant time processing low-level, rule-generated alerts. This gap makes it challenging for defenders to work at the higher strategic level of their attackers, putting them at a disadvantage implementing an effective and relevant defense. Security Operations Centers (SOCs) we interact with continually tell us they are interested in better understanding the behaviors of adversaries. The low-level details they have now make this very difficult.

APT (Advanced Persistent Threat) attackers have turned this semantic gap into a critical problem. APTs use an increasingly rich arsenal of over 200 distinct documented techniques [1] to accomplish their goals. Spending their days focused

S. Chen et al. (Eds.): SecureComm 2019, LNICST 304, pp. 235–254, 2019.
https://doi.org/10.1007/978-3-030-37228-6_12

on low-level events, defenders find themselves at an increasing distance from the playbooks of experienced and skilled attackers. To keep up with the offense, the defense needs to connect its low-level detection details to the offense's game plan, tactics, and techniques.

Two challenges now faced by SOCs magnify the impact of this semantic gap:

1. *Alert Volume.* SOCS face an overwhelming volume of alerts. A recent blog cited pervasive "alert fatigue" as defenders deal with "a flood of alerts and out of tune tools" resulting in an average of 4.35 days for a SOC to resolve a security incident [14].
2. *Talent shortage.* There is a significant talent shortage and turnover at the Tier 1 level of SOCs [14,28], people in the front lines of alert processing. This means the first people to see an attacker signal are often new at their jobs, and less likely to possess a clarifying mental context for interpreting the alerts they see (e.g., is this malicious or not?).

One fallout of defenders operating at this lower level is a winding journey of defensive rule-tuning. Because false positives have a high labor cost, noisy alert-generators may be discarded to improve precision in detection. Turning off alerts can lead to decreased recall and missing dangerous events. This in turn leads to new rules to address known blind spots. Eventually, this process would converge to a stable rule set with higher precision and recall. However in practice, because attackers constantly evolve their techniques this stable state is elusive and rule-tuning becomes hysteresis between precision and recall. In an ideal world, defenders would be able to advance both precision and recall together, keeping pace with their attackers.

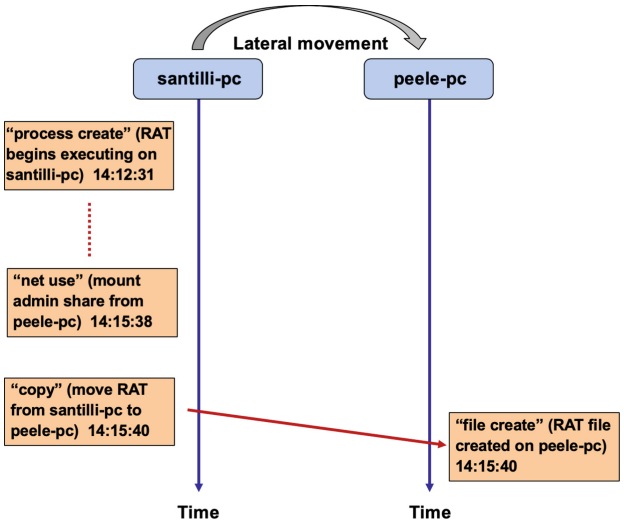

Fig. 1. A set of host-based events implementing a lateral move by remote file copy. (Color figure online)

The problem we are addressing in this paper is closing this semantic gap by elevating the level at which the defense operates to better match the offense. Our approach to address this problem involves three parts: First, we tag alert rules with TTP (Tactics, Technique and Procedures) labels, representing likely strategic reasons for detected alerts. Second, we coalesce TTP-labelled alerts into *semantic clusters* which link adversary tactics and techniques to low-level evidence in a single rich knowledge structure. Third, we repeatedly play automated cybergames, using a red team programmed with TTP-level threat intelligence to attack the defense in varying and realistic ways. We then use scored cybergame outcomes to tune our defense. Our cybergames are scored by comparing red and blue semantic clusters, which highlights defensive blind spots and offensive patterns, enabling high-level adjustments by the defense.

Figure 1 illustrates the notion of a semantic cluster. Taken individually, the four events do not clearly communicate adversary TTPs. However when grouped and labeled as in this figure, these events are clearly linked by the semantics of a strategy to move laterally by means of a remote file copy. Such annotated clustering reduces the semantic gap between attackers and defenders, and immediately suggests adversary capabilities and previous actions (e.g, stealing credentials for the lateral move target), the current stage of the attacker's kill chain and possible future actions. Multiple semantic clusters in sequence would begin to reveal patterns of strategic activity.

In this paper we show that, beyond simply providing a visual aid to humans, semantic clustering of events around TTP annotations enables automated self-assessment and expedited self-improvement of defensive posture. Through automated cybergames, scored in terms of semantic clusters instead of traditional events, we demonstrate significant and simple improvements that were not otherwise obvious, enabling more rapid and focused defensive reconfiguration.

We make the following **contributions:**

1. We describe BSF, a semantic clustering format for cyber-detection. BSF provides a "step" level which includes a label for the adversary's believed malicious technique, and a hierarchically nested "event" level linking in events which provide evidence for that technique.
2. We describe a fully automated cybergaming environment, and show how it can be used to assess and improve an organizations' cyberdefensive posture.
3. In the context of cybergames, we show how the use of semantic clustering provides a significant advance over individual events as a means of assessing and improving defensive posture.

In Sect. 2 we provide background and discuss related work. In Sects. 3 and 4 we describe our "semantic cluster" format for recording cyber attacks and our automated cybergaming environment respectively. In Sect. 5 we discuss the scoring of automated cybergames, including experiences with event-based scoring and the alignment of semantic clusters. In Sect. 6 we describe experiments using automated cybergames scored with semantic clusters to improve an organization's defensive posture. Finally, in Sect. 7 we summarize our results and describe future work.

2 Background

In this section we provide background for the key concepts discussed in this paper: semantic gaps in cybersecurity, cyberdetection formats, and cybergaming environments.

Semantic Gaps. The theme of a semantic gap has arisen across computer science, given the importance of implementing useful abstraction layers in databases, programming languages, networks, and operating systems. In Cybersecurity, a semantic gap has been cited between outliers and actual attacks in anomaly-based intrusion detection [36] and between signature-based and behavior-based intrusion detection [30], which is closer to our use of the term. Specifically, by "semantic gap" we mean the difference between conducting standard cyberdefensive activities such as adversarial activity detection and false positive reduction at the level of:

1. System-level events, such as flow initiation and copying a file, which may be benign or malicious, and
2. Abstract behavioral tactic and technique labels from an actively curated taxonomy of malicious behavior.

This gap and its importance has been publicized in Bianco's "Pyramid of Pain" [13].

Several behavioral taxonomies exist [1,2,8]. In this work we use the MITRE-developed framework Adversarial Techniques, Tactics, and Common Knowledge (ATT&CK™[1]), currently employed in security products, third-party evaluations, and research projects [6,9,25]. As a model of system-level events, we use the CAR framework [4].

Malicious Behavior Formats. In complex enterprises, common representation formats are needed to enable information sharing among components. This is especially true in cyber-security: standard formats are needed to ensure individual components (e.g., logging, filtering, alert-generation, analysis), potentially from multiple vendors, can work together effectively in cyber-defense. Such formats can also be used by human analysts to log and forensically analyze observed malicious behaviors.

In this paper, we describe a representation format (BSF) for episodes of adversarial activity which relates high-level malicious behavior to specific events providing evidence for those behaviors in a single unified representation.

Splunk's CIM (Common Information Model) [19] supports search (e.g., in a Splunk repository), and provides an interoperability standard for transmitting cyberdefensive information in an enterprise. Its *intrusion detection* submodel enables the description of system events in an intrusion, and is analogous to the CAR event model. The Elasticsearch Common Schema [12] similarly contains an event-based sub-schema. Neither schema contains high-level behavioral designators, enabling behavior-event linkage.

[1] https://attack.mitre.org/wiki/Main_Page.

Another format for describing malicious behavior is MIST (Malware Instruction Set) [37]. MIST is a space-optimized format for describing malware behaviors, providing useful features for malware classification [31]. MIST provides a taxonomy for aggregating related system-level actions (e.g., file system accesses) into a single category. However, MIST does not relate these system-level actions to adversarial tactics and strategy.

Cybergames. Cybergames echo military exercises in which a "friendly" adversary – i.e., a red team – is engaged to purposefully attack an enterprise guarded by defenders – i.e., the blue team – for the purpose of stress testing the blue team to see how they would respond to a real threat. Such cybergames are self-assessments in that they use a trusted red team to tell them their weaknesses as opposed to learning what their weaknesses are during a real breach. Using the results of the assessment can further be a catalyst for self-improvement, where the enterprise uses the cybergame results to identify how they should make improvements.

Conducting a cybergame requires several components including: a gameboard (the network of computers in which the cybergame is played), a red team, a blue team and supporting services (e.g., data collection and logging, gameboard provisioning and cleanup, game scoring and analysis).

Game logs record events detailing the activity of both red and blue teams during the game, enabling post-game analysis. For example, red might log a credential dumping action on host H at time T, and blue might log a protected memory access alert on H at a time near T. Examining logs after the game would align these, crediting blue with successful detection.

Cybergames were conceived of as a powerful way to address the subjective nature of security assessments [21]: games have measurable outcomes and security vulnerabilities are recorded in game logs. Studying game outcomes can yield insights, such as blindness to a specific exploit, which can be used to improve defensive posture. In recent years additional benefits of cybergames have been realized, including: evaluating new and potentially useful security technologies [17], enhancing the robustness of system designs [20,40], using threat intelligence to emulate a specific advanced threat and assess enterprise readiness [15,27], and using cybergaming as an experimental framework in which to test hypotheses [24].

A particular value of cybergames, in the context of the potential for precision and recall hysteresis (in defensive rule tuning, as described in Sect. 1), is that a cybergame effectively reveals *both* false negatives and false positives at once, leading to steadier progress in the improvement of enterprise defenses.

Cybergame activities can be labor intensive, however, precluding many of the anticipated benefits. Manual game preparation can take over a year [29] and post-game data analysis can involve months [24]. Thus a recent research goal has been *cybergame automation*, in which one or more components of a cybergame are automated as invokable software systems.

As an example, cyber ranges (e.g., [18,32]) help automate the gameboard component by providing a dedicated computer network for hosting cybergames

and automated services for network configuration, virtual machine provisioning, and synthetic user noise generation (e.g., [33]). Research and systems automating the red team also exist [5,15,16,22,34].

In this paper we describe a *fully* automated cybergaming environment (BRAWL) in which every component operates under software control; we are unaware of any other such environment. Beyond mitigating costs, the primary value of fully automated cybergaming is the ability to learn improved defensive strategy through experimentation involving repeated cybergames.

3 BSF Syntax

In this section we present BSF (BRAWL Shared Format), a language for describing the progression of cyber attacks. While BSF has many potential uses (e.g., forensic analysis of attacks in a SOC setting) in this paper we focus on its use enabling cybergame scoring, that is: scoring how well the defense detected the offense's activities. Specifically, due to issues identified in scoring cybergames solely by events described in Sect. 5, we realized that we needed a representation that could be used to correlate both individual events as well as semantic intent. This motivated us to create BSF which explicitly links events to semantic annotations.

As illustrated in Fig. 2, BSF is hierarchical and consists of three levels: operations, steps, and events. The *operation* serves as a delimiter for an adversary activity of interest (e.g., one day of forensic log analysis, a single cybergame), and contains a sequence of steps.

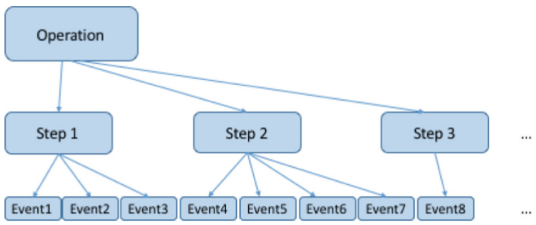

Fig. 2. BSF structure

BSF Steps. A step corresponds to the execution of a single technique (e.g., Remote Desktop Protocol, Powershell) by the adversary. More formally, a step S is a six-tuple $\{T, \mathcal{T}, E, \mathcal{E}, H, D\}$ such that:

- T is the *key technique* executed in step S; the single technique which describes this step.
- $\mathcal{T}$ is a set of *ancillary techniques* describing S [2]. We assume $T \in \mathcal{T}$.

[2] Ancillary techniques are used due to some semantic overlap in ATT&CK. For example, using a powershell command to dump credentials (T1003) could also be correctly labeled as an instance of execution via powershell (T1086).

- E is the *key event* in S; a single event initiating T.
- $\mathcal{E}$ is a set of ancillary events in S, including: attacker-executed events following E, and events causally-related to events in $\mathcal{E}$ (e.g., triggered child processes). We assume $E \in \mathcal{E}$.
- H is the host on which step S occurs.
- D is the destination host for techniques in which action moves from one host to another (e.g., a lateral movement).

Techniques are chosen because they link low-level event detection to higher-level strategy: techniques are specific enough for detection by alert rules but they are also indexed to higher-level adversary tactics (e.g., defensive evasion, lateral movement) which convey the phase of the adversary's attack. Techniques are well-documented [1] which provides an operational semantics for BSF steps, including: a unique ID, a textual description of the technique, associated tactics, platforms, examples of malware and APTs known to use this technique (useful for linking threat intelligence), mitigations, detection methods, and a list of literature references.

An example JSON-formatted BSF step from the blue bot in a cybergame is shown in Fig. 3. In this step, the attacker is detected using key technique T1105, remote file copy, to perform a lateral movement from host `sounder-pc` to destination host `fulco-pc`.


```
{
    "provenance": "Analytic - Copy Into Network Share (via \"cmd /c copy\" or \"xcopy\")",
    "key_event": "5b3512df6443110d68c617b8",
    "key_technique": "T1105",
    "attack_info": [
        {
            "technique_name": "Remote File Copy",
            "tactic": ["Lateral Movement"],
            "technique_id": "T1105"
        }
    ],
    "time": "2018-03-01T14:56:30.319000+00:00",
    "events": [
        "5b3512db6443110d68c6177b",
        "5b3512df6443110d68c617b8",
        "5b3513986443110d68c620af",
        "5b3513466443110d68c61cc3"
    ],
    "nodetype": "step",
    "id": "3aaca9db-bb3b-4052-a1fa-59f16e9f8eb9",
    "host": "sounder-pc",
    "dest_hosts": [
        "fulco-pc"
    ],
    "confidence": 0.9426254358169935
},
```


Fig. 3. Example BSF step: lateral movement by remote file copy

The step in Fig. 3 contains four events whose ids are in the *events* array. Although not shown here, these events are similar to the events shown in Fig. 1. From the red perspective, events are the implementation of a step. From the

blue perspective, events provide evidence for a step. The events associated with a step are represented using the CAR data model [4].

In Sect. 4.1 we describe how such steps, or semantic clusters, are generated.

4 Experimental Environment

Our automated cybergaming environment is illustrated in Fig. 4. It contains four major components: gameboard, red bot, blue bot, and game analysis pipeline. The gameboard is generated by a system called BRAWL which automates Virtual Machine (VM) provisioning and game board setup for each game. Gameboards consist of a virtual network, including Windows hosts and an Active Directory server. Each Windows hosts runs Sysmon [11] and forwards events to an Elasticsearch server, which acts as a central database of all game logs. Sysmon generates events involving processes, flows, files, the registry, and other operating system entities. We use a popular Sysmon config that filters various high-volume system activity [10].

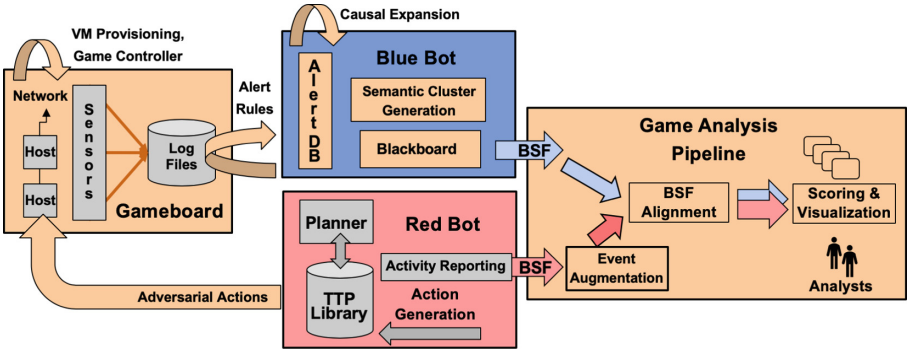

Fig. 4. Automated cybergaming architecture

The BRAWL game controller provides an API for communicating with the red and blue teams, or *bots*, which are automated software systems responding to start and stop messages from the controller. After a game begins, the game controller turns on logging, and game activities are captured including: (a) the Sysmon event logs, (b) the red bot's self-report, (c) the blue bot's reported detections. Both bot reports are recorded in BRAWL Shared Format (BSF), described in Sect. 3, and finally forwarded to the game analysis pipeline for alignment and scoring, as described in Sect. 5.

We use CALDERA [15,16] as the default red bot for blue to play against. CALDERA uses a planner to generate and automatically execute a post-compromise attack through a network of computers. At each step in its plan, high-level tactics and techniques are selected from ATT&CK and executed by CALDERA in the gameboard, resulting in detectable system-level events.

CALDERA can be programmed to emulate specific adversary behaviors (e.g., the techniques and tactical sequences used by APT 21). For this paper, we assume the red bot is programmed to execute a series of attacks consistent with behavioral threat intelligence, such as a SOC might obtain, in order to test its defenses against such attacks.

The blue bot is an ongoing research project at MITRE. It uses a two-phased approach to identify adversary activity in a network. The first phase, called CASCADE [3], executes alert rules over the log files as the game progresses and caches the resulting suspicious events in a database. These events are then expanded by recursively following system-level causal relationships (e.g., parent and child processes, flow connections) resulting in a larger set of suspicious events. The second phase, called BB-ATE, post-processes BSF in an agent-based blackboard architecture to infer additional information (the details of which are beyond the scope of this paper). Between these phases, semantic clusters (BSF steps) are generated. The most important functions of our blue bot, for the purposes of this paper, are causal expansion of events and semantic cluster generation.

4.1 Semantic Cluster Generation

To construct blue BSF steps, such as the example in Fig. 3, we begin by labelling alert-generating rules with the presumed detected technique(s). An alert rule consists of a query that targets a particular adversary behavior and is tagged with the related techniques and tactics. For example, the "Winlogon Spawning Cmd" alert rule triggers on a `cmd.exe` process with a parent process of `winlogon.exe` and is tagged with the related technique "T1015 Accessibility Features", which can be used to implement persistence and privilege escalation tactics. When these rules are triggered, an initial step is formed with the triggering event as the step's *key event* and the rule's technique as the *key technique*.

Ancillary events for a step are identified by applying the causal expansion logic of the first phase, and recursively following system-level chains of causality from the key event, such as process creation and flows. Thus, for example, all descendent and antecedent process creation events would be added. Through experience, we have learned to heuristically truncate this set at the host boundary of the key event, including all events on that initial host plus the first event on any connecting host. The resulting set of events becomes the *ancillary events* of the step being formed.

Red step construction proceeds in a similar manner, the major difference being the knowledge asymmetry of red and blue: red's internal planning logic generates its TTPs and thus, in contrast to blue, it has no need to infer them. Every time red executes a technique it forms a step based on that technique, and records it as BSF. On the other hand, red lacks blue's ability to query log files for causally-generated events, and is uncertain about the full effects of its actions. Thus the red bot records in each step only the events it is aware of.

5 Scoring Cybergames

Our intent for scoring cybergames is to rate the defense's performance, enabling improvement of the defensive posture. While computing scoring metrics is straightforward, the most challenging aspect of scoring cybergames is the alignment, or matching, of red reports to blue detections. In this Section we discuss how scoring is accomplished, including our initial event-based efforts, some of the challenges we encountered that motivated semantic clustering, and our final semantic-cluster-based scoring approach illustrated by the final component of Fig. 4: the *game analysis pipeline*.

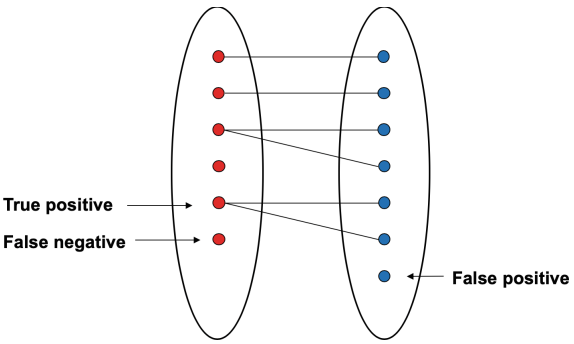

Fig. 5. Bipartite scoring graph aligning red activities with blue detections (Color figure online)

At the end of each game red reports its activities and blue reports its detections, represented in Fig. 5 as colored nodes. These reports are extracted from the Elasticsearch server and processed by the analysis pipeline. To score the game, we assign edges *aligning* blue's detections with red's actions and then then compute metrics over the resulting graph. Red nodes matched by a blue node are considered *true positives* (TP) [3], unmatched blue nodes are *false positives* (FP), and unmatched red nodes are *false negatives* (FN). We use the standard metrics of precision $= TP/(TP + FP)$ and recall $= TP/(TP + FN)$, and $F1$ which is the harmonic mean of precision and recall. In Fig. 5 precision is .80, recall is .67, and $F1$ is .73.

Initial Event-based Alignment and Scoring. We did not initially use semantic clustering. In our first games the blue bot reported the system-level *events* it detected, such as process and file creations, and the red bot similarly reported the events it caused. Our initial approach to scoring was thus event-based: each node in the bipartite scoring graph represented an individual red or blue event,

[3] Note that blue can report on the same red activity more than once if multiple sensors detect different aspects of the same red action. In this case, we only count one true positive.

and scoring a game to determine how well blue detected red's activity involved assigning edges aligning blue events to red events and computing metrics.

Events were aligned by computing a match-strength metric, and declaring if the metric exceeded a threshold. This metric scores the similarity of key value pairs in the two events, including: hostnames, object and action (e.g., "process create"), and times within a window of parameterized size (set to 30 s).

We encountered several alignment challenges in scoring these initial games. A simple one was due to syntactic differences in red and blue event reports. For example, red reported file creation as `type:create_file` while blue reported file creation as `action:create` and `object:file`. We harmonized these through introducing standards and post-processing, improving alignments.

A second such issue was due to red's inability to know the full extent of its actions. For example, in Fig. 1, red would not see the file create event on peele-pc, which was a downstream effect of its actions. Blue, in contrast, sees such actions by querying the logging infrastructure. Due to this inherent asymmetry, blue was often incorrectly charged for false positives (i.e., false false positives) because blue detected actual events red had caused, but which red did not observe and report. This in turn distorted game scores, especially for precision.

We remediated this discrepancy by post-game "event augmentation" (as shown in Fig. 4): for each event reported by red, we applied the same causal expansion logic available to the blue bot, augmenting red's reported events for alignment as if red had equivalent power to view logs.

However, rectifying these basic alignment issues exposed more fundamental flaws in our focus on individual events, which resulted in our move to a semantic cluster representation for detection, alignment and scoring:

- Generating scores based on individual events weights all events equally and does not distinguish event importance. Certain events, such as the remote file copy event in Fig. 1, more clearly suggest an adversary activity (e.g., a potential lateral move by remote file copy) than other events, such as causally-resulting file create event in Fig. 1. These resulting events provide useful redundant evidence supporting that activity but it did not seem these should be scored identically to events indicating a *new* activity.
- Even if blue achieves a perfect precision and recall on individual events, it has revealed practically nothing about red's strategic plan in terms of tactics such as discovery, lateral movements, persistence, data collection, and exfiltration (i.e., a large semantic gap exists between blue's detection and red's activity). In particular, if blue achieved a 75% recall of individual events, we cannot tell the difference between detecting each tactic at a 75% level and entirely omitting the detection of certain tactics. Such inability to measure tactical coverage precludes promising defensive strategies, such as the *threat tuple* of [25], for evaluating the current extent of a cyber attack.

Alignment and Scoring Using Semantic Clusters. Moving to semantic clusters closed this semantic gap, however alignment is more complex because BSF steps have multiple components (i.e., key technique, key event, ancillary technique set, ancillary event set) making broad spectrum of red-blue alignments are possible.

In particular, matching the events portion of steps (i.e., the key and ancillary events of red and blue) could be done in multiple, increasingly permissive, ways:

1. Key events match between red and blue.
2. Key event of red is in the ancillary events of blue *and* vice versa.
3. Key event of red is in the ancillary events of blue *or* vice versa.
4. Ancillary events of red and blue intersect.

The same four options exist for matching the *techniques* portion of a step as well (i.e., the key technique and ancillary techniques of red and blue), leading to a *sizable* space of potential overall step-matching methodologies.

In our experiments, we have chosen to match steps by requiring that at least the third criteria hold for *both* events and for techniques. Intuitively, this requires that matching steps agree, at least in a loose sense, on both a key event and a key technique.

6 Experiments

In this section we describe the use of automated cybergaming with BSF semantic clusters to improve an organization's defensive posture, demonstrating how a SOC environment can benefit from this approach to narrowing the "semantic gap". In particular, we study two defensive adjustments made possible by the improved organizational insight into TTPs provided by semantic clustering, and by the rapid feedback enabled by scored cybergames.

6.1 Setup

The BRAWL game board consists of a Windows domain with a domain controller and ten Windows 8.1 virtual machines. We collect a dataset of 48 hours of system logs, including a total of 31,073 Sysmon events. During this period, we use CALDERA to emulate three distinct intrusions, with the red bot exercising a range of post-exploit behaviors, including discovery, credential access, lateral movement, execution, privilege escalation, and collection. The system logs in our dataset also include benign activities, such as typical Windows background processes and services, installing software, and remote admin commands.

The results in this section are obtained by repeatedly [4] applying the blue bot, in varied *configurations*, to detect red activity in our dataset. A configuration is a specific set of alert rules. We score each configuration based on how well blue detected BSF steps which match the BSF steps reported by red.

[4] We can apply the blue bot repeatedly to the same red activity because the blue bot does not alter the gameboard.

6.2 Baseline

Our baseline configuration consists of 43 commonly-used alert rules, distributed over tactics, from the Cyber Analytics Repository (CAR) [4]. In this and the following exercises, we generate a *range* of detection configurations by incrementally adding rules into an initially-empty configuration in order of increasing noise generation (i.e., false positive rate), until the full set of rules is reached. This emulates a typical SOC's challenge of configuring their rule sets based on the levels of noise they are capable of handling, thus trading precision for recall as more rules are added. To determine noise levels, we measured the false positive rate of each alert rule over a 24 hour period on a production network, separate from our game board, consisting of about 250 operational hosts.

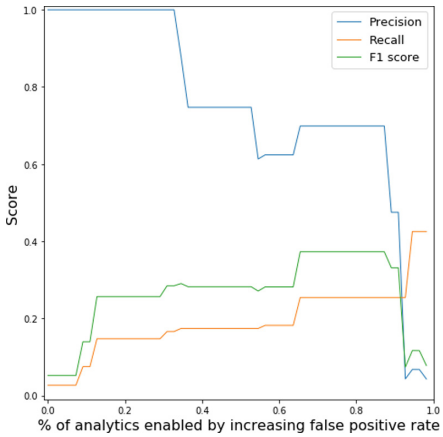

Fig. 6. Precision, Recall, and F1 scores with baseline alert rules

Figure 6 shows precision, recall, and F1 scores by percentage of alert rules enabled. The left-side of the graph represents more conservative configurations, i.e., higher precision and lower recall, whereas the right side represents more aggressive configurations with lower precision and higher recall. As can be seen, some of the noisier rules which are added toward the end are also among the most effective detectors. The maximum F-score of 0.373 with 65% to 87% of alert rules enabled represents an optimal balance between precision and recall, with a precision of 0.699, but a recall of only 0.254. The most aggressive configuration with 100% of alert rules has a slightly higher, but still low recall of only 0.425. Even the most aggressive configuration clearly contains blindspots and fails to detect most adversary activity.

To improve recall, a SOC would typically add rules. Given semantic annotations on rules, and that recall is being measured in terms of semantic clusters (for which a match requires TTP-level agreement), in this exercise we can readily determine coverage gaps in which adding more rules is likely to improve recall.

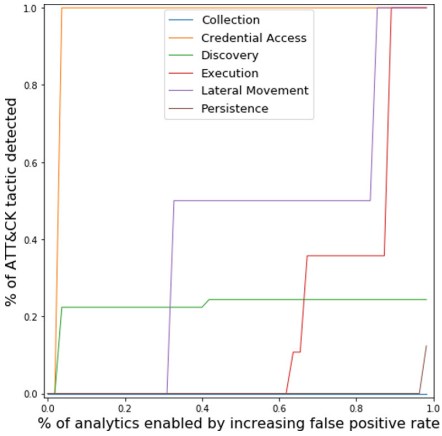

Fig. 7. Tactic detection with baseline alert rules

Figure 7 shows the detection rates for different tactics as alert rules are enabled. Even in the most aggressive configuration at the far right, the discovery, persistence, and collection tactics are being detected at no more than a 0.243 rate, illustrating the most promising semantic regions for additional rules.

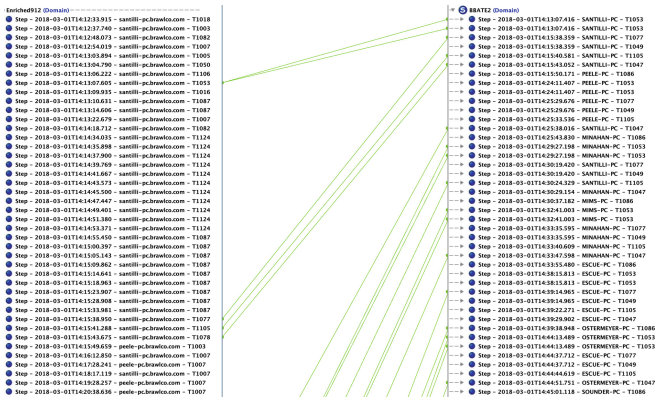

Fig. 8. BSF matcher visualization of missed discovery techniques

As another illustration of TTP annotations revealing tactical detection gaps, Fig. 8 is a screenshot of a BSF matching tool showing red BSF from a single cybergame on the left and the corresponding blue BSF on the right. The lines show detected step alignments, and the unmatched gap areas on the left show blue missing much of red's initial activity. Upon examination, the technique IDs red reported in these detection gaps (e.g., T1018 "remote system discovery",

T1082 "system information discovery") are dominantly discovery activities, one of the tactics shown to be poorly detected in Fig. 7.

6.3 Reducing False Negatives

Given that the baseline configurations have gaps for discovery, persistence, and collection, and that discovery is an important tactic early in the adversary kill chain we focus on improving detection for discovery by adding 14 new discovery alert rules to the baseline configuration, increasing the alert rule set from 43 alerts to 57 alerts. This raised the detection rate for the discovery tactic from 24% (in Fig. 7) to 75%. Figure 9 shows the precision, recall, and F1 scores for the new configuration with these added discovery alert rules.

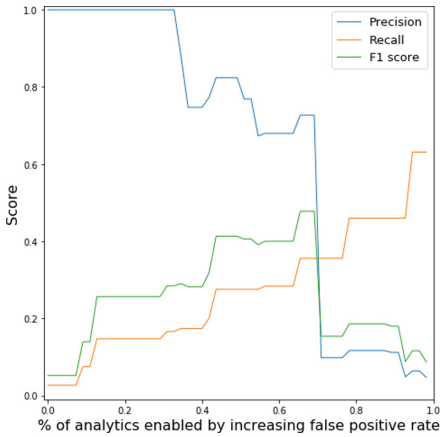

Fig. 9. Precision, Recall, and F1 scores with added discovery alert rules

The maximum F1 score increased from 0.373 to 0.478, with recall in this configuration increasing from 0.254 to 0.356 and precision increasing slightly from 0.699 to 0.727. The most aggressive setting yields an even higher recall of 0.631, but precision drops precipitously because many of the new rules are noisy.

6.4 Reducing False Positives

To reduce time-consuming false positives, SOCs typically disable alert rules or whitelist events known to generate a large number of false positives. However, both methods can also increase false negatives because the triggering conditions of rules and whitelisted events frequently can occur in both benign and malicious contexts. For example, post-exploit adversary behaviors and system administrator actions have a large overlap.

By detecting semantic clusters, another disambiguation technique becomes an option for false positive removal. *Sequences* of adversary TTPs, such as the

key techniques in semantic clusters, provide a disambiguating context for distinguishing benign and malicious events.

For this exercise we collect a separate dataset of two weeks of system logs to train a model, including a total of 30,332 Sysmon events. During this period, similar to our main dataset, we use CALDERA to emulate three intrusions. We then run the blue bot on this training dataset and extract trigrams of reported adversary activity (i.e., techniques). For each trigram, we compute the number of true positives and false positives, retaining any trigrams with only false positives. We then use the remaining trigrams as a filter, removing any matching trigram sequences in blue's output.

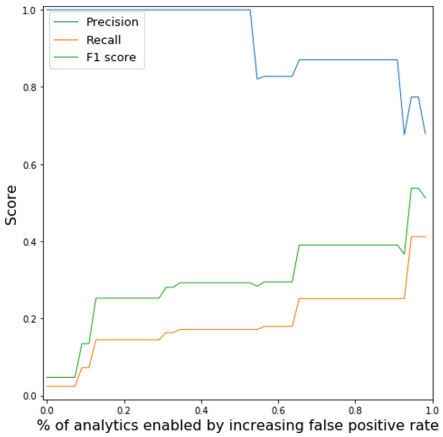

Fig. 10. Precision, Recall, and F1 scores with ngram filtering

Figure 10 shows precision, recall, and F1 scores with ngram filtering. The graph shows a significant improvement in precision across all configurations relative to the baseline graph (Fig. 6). The maximum F1 score increases from 0.373 to 0.538, with precision in this configuration increasing from 0.699 to 0.774. Recall for the maximum F1 score also increases from 0.254 to 0.412, because high precision is achieved with a more aggressive configuration (i.e., more rules enabled) than in the baseline.

There are challenges to our approach as performed in this exercise: clever adversaries will rotate techniques to accomplish the same tactic, thus it would be important to include sequences of tactical information as well. Also, adversaries could include "no-op" actions to disrupt trigram patterns [39], which would require the use of more sophisticated data structures such as skipgrams [23]. While such improvements are possible, this exercise demonstrates the power of addressing false-positive elimination at the semantic level (i.e., in the conceptual world of adversary), instead of at the level of system alerts.

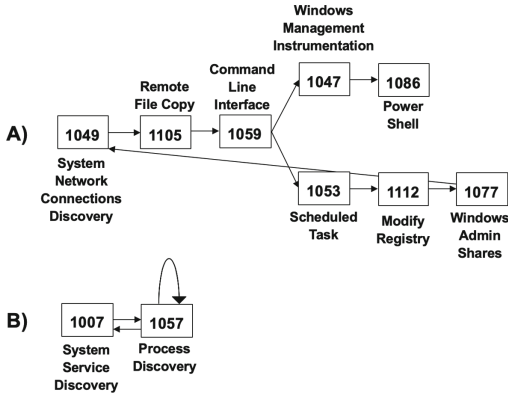

Fig. 11. Adversary state diagrams derived from N-Grams

Figure 11 provides an interesting illustration of why these trigrams work as a false positive filter. These two state diagrams were obtained in a data-driven manner by concatenating true positive trigrams into longer sequences involving cycles and branches. However, the red research team quickly recognized these diagrams as logic their planner would generate.

6.5 Summary of Improvements

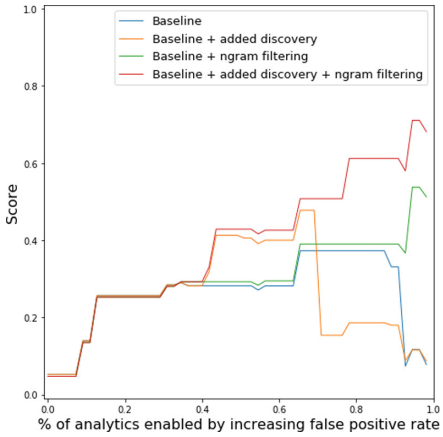

Fig. 12. Summary F1 graph

Figure 12 summarizes F1 scores for the baseline, adding discovery alert rules, using ngram filtering, and both improvements together. The results show consistent increases in the maximum F1 score. The added discovery alert rules

results in some less optimal alert configurations (when more than 70% of the alert rules are enabled) due to including noisier (i.e., high false positive rate) alert rules. These alert rule configurations become much more effective when we enable ngram filtering, reducing their false positive rates. Interestingly, ngram filtering was meant to decrease false positives and improve precision, yet also results in reducing false negatives and improving recall, since it can achieve high precision with a broad set of alert rules enabled. Lastly, we receive the largest improvement when we combine the two improvements.

7 Summary and Future Work

In this paper we have shown the advantages of a new method for improving enterprise cyberdefense: semantic clusters in automated cybergames. Semantic clusters make a model of attacker strategy *explicit* as machine-readable data structures, clustering traditional events around TTP annotations from a well-known behavioral taxonomy, enabling the defense to operate at the strategic level of APT attackers without sacrificing the ability to drill into evidential details.

We have also demonstrated the use of a *fully* automated cybergaming environment. Beyond mitigating costs, a key benefit of fully automated cybergaming is the ability to learn improved defensive strategy through experimentation. We illustrate this using cybergames scored in terms of semantic clusters, revealing effective and automatable defensive enhancements, such as covering tactical detection gaps and using patterns of strategic behavior to reduce false positives.

This research opens the door on many potential future research directions. The rich semantics available in our BSF semantic cluster format enable the application of various intelligent techniques, such as automatically infering and filling detection gaps. BSF streams can support dynamic response to attacks. While ordinary event streams can be used, the additional strategic information should improve the defense's ability to respond effectively [38]. Finally, the ability to automatically run and score cybergames on a real network provides a foundation for red and blue teams to co-evolve through reinforcement learning [26] (e.g., as illustrated by Google's Alpha Go learner [35]).

Acknowledgements. We would like to thank Andy Applebaum for his helpful comments and suggestions reviewing this manuscript. We also want to acknowledge the generous support provided by the BRAWL, CALDERA, and CASCADE teams. This work was supported by a grant from the MITRE Innovation Program.

References

1. ATT&CK: Adversarial Tactics, Techniques, and Common Knowledge. https://attack.mitre.org. Accessed 24 Apr 2019
2. CAPEC: Common Attack Enumeration and Classification. https://capec.mitre.org. Accessed 24 Apr 2019
3. CASCADE. https://github.com/mitre/cascade-server. Accessed 30 Apr 2019

4. Cyber Analytics Repository. https://car.mitre.org/data_model/. Accessed 24 Apr 2019
5. Endgame RTA: Red Team Automation. https://www.endgame.com/blog/technical-blog/introducing-endgame-red-team-automation. Accessed 24 Apr 2019
6. First Round of MITRE ATT&CK Product Evaluations Released. https://medium.com/mitre-attack/first-round-of-mitre-att-ck-evaluations-released-15db64ea970d. Accessed 24 Apr 2019
7. MANDIANT: Exposing One of China's Cyber Espionage Units. https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf. Accessed 24 Apr 2019
8. NSA/CSS Technical Cyber Threat Framework v2. https://www.nsa.gov/Portals/70/documents/what-we-do/cybersecurity/professional-resources/ctr-nsa-css-technical-cyber-threat-framework.pdf. Accessed 24 Apr 2019
9. Red Canary ATT&CKs (Part 1): Why We're Using ATT&CK Across Red Canary. https://redcanary.com/blog/red-canary-and-mitre-attack/. Accessed 24 Apr 2019
10. Swift On Security - Sysmon Config. https://github.com/SwiftOnSecurity/sysmon-config. Accessed 24 Apr 2019
11. Sysmon 9.0. https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon. Accessed 24 Apr 2019
12. The Elasticsearch Common Schema. https://github.com/elastic/ecs/tree/master/schemas. Accessed 24 Apr 2019
13. The Pyramid of Pain. http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html. Accessed 24 Apr 2019
14. The SOC Gets a Makeover. https://www.darkreading.com/risk/the-soc-gets-a-makeover/d/d-id/1332744/. Accessed 24 Apr 2019
15. Applebaum, A., Miller, D., Strom, B., Foster, H., Thomas, C.: Analysis of automated adversary emulation techniques. In: Summer Simulation Multi-Conference, p. 16 (2017)
16. Applebaum, A., Miller, D., Strom, B., Korban, C., Wolf, R.: Intelligent, automated red team emulation. In: 32nd Annual Conference on Computer Security Applications, pp. 363–373. ACM (2016)
17. Bodeau, D., McCollum, C., Fox, D.: Cyber threat modeling: survey, assessment, and representative framework. Tech. Rep. 16-J-00184-01, The MITRE Corporation: Homeland Security Systems Engineering and Development Institute (April 2018)
18. Ferguson, B., Tall, A., Olsen, D.: National cyber range overview. In: Military Communications Conference (MILCOM), 2014 IEEE, pp. 123–128. IEEE (2014)
19. Fletcher, T.A., Sharp, C., Raghavan, A.: Optimized common information model, US Patent App. 14/800,678 (2016)
20. Fox, D., McCollum, C., Arnoth, E., Mak, D.: Cyber wargaming: framework for enhancing cyber wargaming with realistic business context. Tech. Rep. 16-J-00184-04, The MITRE Corporation: Homeland Security Systems Engineering and Development Institute, November 2018
21. Goldis, P.D.: Questions and answers about tiger teams. EDPACS **17**(4), 1–10 (1989)
22. Hoffmann, J.: Simulated penetration testing: from dijkstra to turing test++. In: 25th International Conference on Automated Planning and Scheduling (2015)
23. Huang, X., Alleva, F., Hon, H.W., Hwang, M.Y., Lee, K.F., Rosenfeld, R.: The sphinx-ii speech recognition system: an overview. Comput. Speech & Lang. **7**(2), 137–148 (1993)

24. Kewley, D.L., Bouchard, J.F.: Darpa information assurance program dynamic defense experiment summary. IEEE Trans. Syst., Man, Cybern. - Part A: Syst. Hum. **31**(4), 331–336 (2001)
25. Milajerdi, S.M., Gjomemo, R., Eshete, B., Sekar, R., Venkatakrishnan, V.: Holmes: real-time apt detection through correlation of suspicious information flows. In: 2019 IEEE Symposium on Security and Privacy, pp. 430–445. IEEE (2019)
26. Niculae, S.: Reinforcement learning vs genetic algorithms in game-theoretic cybersecurity, October 2018. thesiscommons.org/nxzep
27. Oakley, J.: Improving cyber defensive stratagem through apt centric offensive security assessment. In: International Conference on Cyber Warfare and Security, pp. 552-XV. Academic Conferences International Limited (2018)
28. Oltsik, J., Alexander, C., CISM, C.: The life and times of cybersecurity professionals. ESG and ISSA: Research Report (2017)
29. Ošlejšek, R., Toth, D., Eichler, Z., Burská, K.: Towards a unified data storage and generic visualizations in cyber ranges. In: 16th European Conference on Cyber Warfare and Security. p. 298. Academic Conferences and publishing limited (2017)
30. Passerini, Emanuele, Paleari, Roberto, Martignoni, Lorenzo: How good are malware detectors at remediating infected systems? In: Flegel, Ulrich, Bruschi, Danilo (eds.) DIMVA 2009. LNCS, vol. 5587, pp. 21–37. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02918-9_2
31. Rieck, K., Trinius, P., Willems, C., Holz, T.: Automatic analysis of malware behavior using machine learning. J. Comput. Secur. **19**(4), 639–668 (2011)
32. Rossey, L.: Simspace cyber range. In: ACSAC 2015 Panel: Cyber Experimentation of the Future (CEF): Catalyzing a New Generation of Experimental Cyber-security Research (2015)
33. Rossey, L.M., et al.: Lariat: lincoln adaptable real-time information assurance testbed. In: Aerospace Conference, vol. 6, pp. 6–6. IEEE (2002)
34. Sarraute, C., Buffet, O., Hoffmann, J.: POMDPs make better hackers: accounting for uncertainty in penetration testing. In: 26th AAAI Conference on Artificial Intelligence (2012)
35. Silver, D., et al.: Mastering the game of go with deep neural networks and tree search. Nature **529**(7587), 484 (2016)
36. Sommer, R., Paxson, V.: Outside the closed world: on using machine learning for network intrusion detection. In: 2010 IEEE Symposium on Security and Privacy, pp. 305–316. IEEE (2010)
37. Trinius, P., Willems, C., Holz, T., Rieck, K.: A malware instruction set for behavior-based analysis (2009)
38. Van Dijk, M., Juels, A., Oprea, A., Rivest, R.L.: Flipit: The game of "stealthy takeover". J. Cryptol. **26**(4), 655–713 (2013)
39. Wagner, D., Soto, P.: Mimicry attacks on host-based intrusion detection systems. In: 9th ACM Conference on Computer and Communications Security, pp. 255–264. ACM (2002)
40. Wood, B.J., Duggan, R.A.: Red teaming of advanced information assurance concepts. In: DARPA Information Survivability Conference and Exposition, pp. 112–118. IEEE (2000)