MTR180158
MITRE TECHNICAL REPORT

MITRE

TTP-Based Hunting

Sponsor: USCYBERCOM
Dept. No.: P522
Contract No.: W56KGU-16-C-0010
Project No.: 0718N00A-WF

Annapolis Junction, MD

Roman Daszczyszak
Dan Ellis
Steve Luke
Sean Whitley

March 2019

Executive Summary

Although common, attempts to detect malicious activity through signatures of easily-changed attributes such as Internet Protocol (IP) addresses, domains, or hashes of files, are brittle and quickly become outdated. This approach is often referred to as signature-based or Indicator of Compromise (IOC) detection. Red Team results and incident analysis provide ample evidence that this approach provides some value, but is ineffective against adaptable threats. This is because adversaries easily and frequently change those attributes to avoid detection.

Anomaly-based detection on the other hand, employs statistical analysis, machine learning, and other forms of big data analysis to detect atypical events. This approach has traditionally suffered from high false positive rates, can require significant investment in large scale data collection and processing, and does not always provide enough contextual information around why something was flagged as suspicious, which can make analytic refinement challenging.

A growing body of evidence from industry, MITRE, and government experimentation confirms that collecting and filtering data based on knowledge of adversary tactics, techniques, and procedures (TTPs) is an effective method for detecting malicious activity. This approach is effective because the technology on which adversaries operate (e.g., Microsoft Windows) constrains the number and types of techniques they can use to accomplish their goals post-compromise. There are a relatively small number of these techniques, and they occur on systems owned by the victim organization. All adversaries must either employ these known techniques or expend vast resources to develop novel techniques regardless of their capabilities or strategic mission objectives. This paper expands on existing best practices to detect malicious behaviors expressed as techniques, using a method that is operating system technology agnostic, and describes the step-by-step procedures to implement.

These three detection approaches are not mutually exclusive. Signature-based, anomaly-based, and TTP-based detection are complementary approaches to one another. However, the relative costs and effectiveness of each approach dictate a significant shift in how these approaches are employed. Because of its efficiency and relatively low investment, TTP-driven hunting may yield benefits far greater than the costs.

Based on existing best practices and supported by experimentation on an operational network, we recommend that hunt operations collect and utilize timely, actionable information on the techniques adversaries must employ across all systems of interest. MITRE ATT&CK™ represents a categorized enumeration of those techniques. Hunt analysts should determine data collection requirements to detect those techniques. System owners should deploy, activate, and/or configure sensors to continuously collect the data required to detect those techniques. Cyber platform developers should incorporate as much native sensing capability as possible into their systems to facilitate this approach (e.g., Microsoft Sysmon and Windows Event Logging). Hunt teams should receive education and training on implementing each step of this methodology and how to extract adversary techniques from cyber threat intelligence. Analysts should partner with threat emulators to develop, test and refine analytics and response actions to maximize effectiveness. To be successful, hunt teams should partner with local system owners to baseline benign activity that may trigger hunting analytics in order to tune the hunt approach for that network.

Table of Contents

List of Figures

1 Introduction

This paper builds upon a growing body of evidence from the cybersecurity community to present a robust and successful approach to detecting malicious activity based on an understanding of adversaries' tactics, techniques, and procedures (TTP) in cyberspace. It attempts to show that, by describing adversary behavior *at the right level of abstraction*, appropriate sensors (host and network-based) can be deployed and analytics can be designed to detect adversaries with high accuracy, even across variations in different implementations. The approach presented, TTP-based hunting, is complementary to existing practices such as using indicators of compromise (IOCs) or using statistical analysis of data to detect anomalies. This paper makes recommendations for how hunting teams can implement a TTP-based approach.

1.1 Definition of Hunting

The word "hunting" is an emerging term within cybersecurity for which the exact definition is still evolving. In the 2017 Threat Hunting Survey, the SysAdmin, Audit, Network, and Security (SANS) Institute (Lee & Lee, 2017) defines threat hunting as, "a focused and iterative approach to searching out, identifying and understanding adversaries that have entered the defender's networks." Sqrrl (2016) defines threat hunting as, "… the process of proactively and iteratively searching through networks to detect and isolate advanced threats that evade existing security solutions." Endgame defines hunting as, "the process of proactively looking for signs of malicious activity within enterprise networks without prior knowledge of those signs, then ensuring that the malicious activity is removed from your systems and networks." (Scarfone, 2016, p. 1). For this paper, "hunting" is defined as *the proactive detection and investigation of malicious activity within a network.* Similarly, a "hunt team" is a *group of individuals dedicated to performing a hunt on a given network.*

1.2 Analysis Space

Malicious activity in cyberspace can be considered in three dimensions: time, terrain and behavior. Every event in cyberspace can be represented as a specific behavior (benign, malicious or suspicious) at a specific time, on a specific machine, process, subnet, or other element of cyber terrain. Each of these dimensions is described below. Figure 1 provides a visualization of these three dimensions with example values for behavior and terrain types.

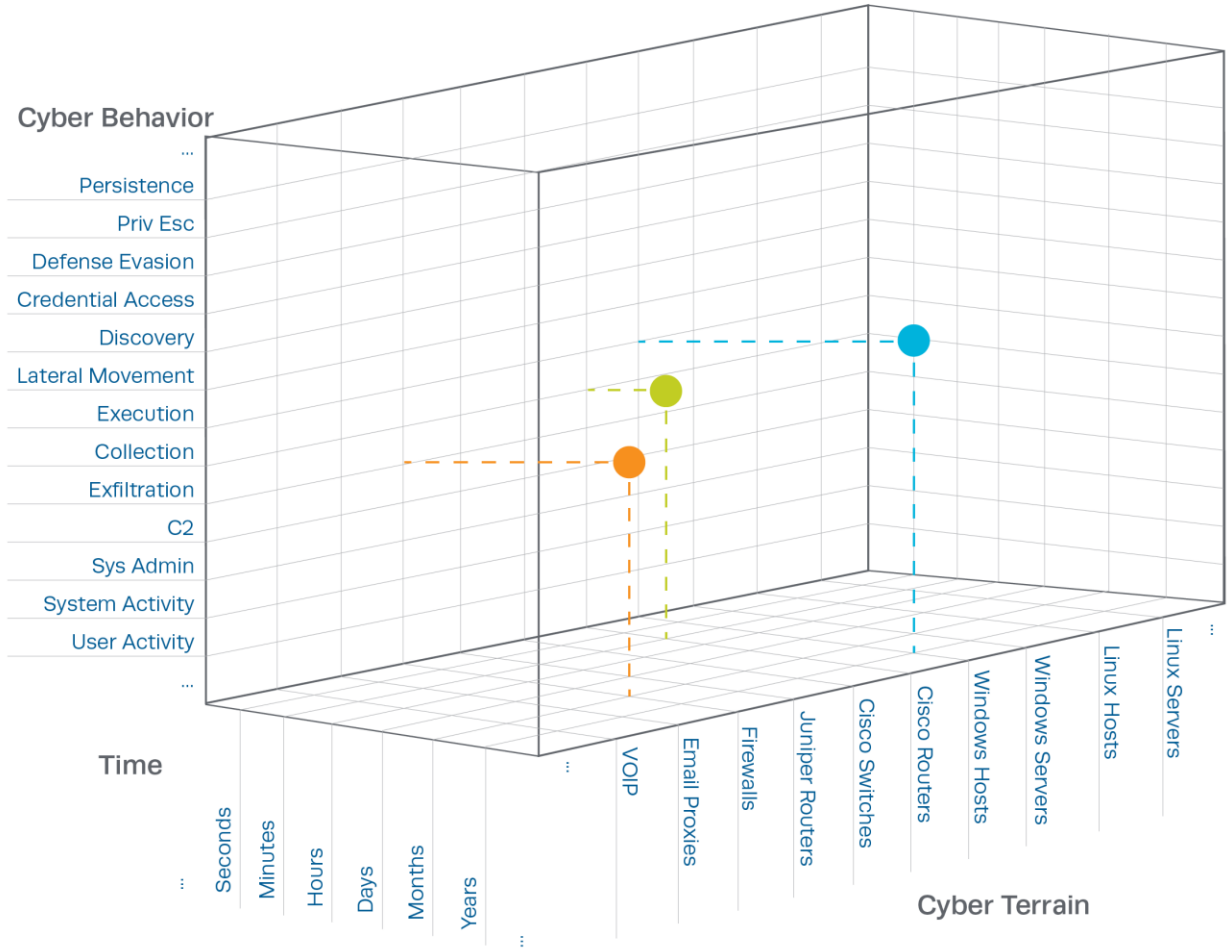

Figure 1 Analysis Space

The time dimension is relatively straightforward. Most evidence of malicious behavior is transient in nature (process content and activity, for example) and must be collected during the intrusion as it cannot be obtained after the fact. Data from continuous monitoring is generally preferred over forensic data collection because hunting for malicious behavior starts with a large, unknown window of time and most forensic data (and its related data collection capabilities) only cover a narrow slice of the time domain for these transient events. It is far more likely that malicious behaviors will fall outside of this slice, thus these tools are less effective for hunting. If used as part of an ongoing investigation, forensic tools can complement other data sources but are much less effective as primary data sources for detection.

Cyber "terrain" can refer to the broad range of hosts, network segments, or other areas where the adversary may be operating. For the purposes of this methodology, terrain is restricted to where defenders have authority to operate and a responsibility to defend – within an enterprise or enclave monitored by a Security Operations Center (SOC) or hunt team. Particular focus should be paid to areas that might be highly targeted by an adversary (e.g., crown jewels) (MITRE, 2018); areas that the adversaries may need to traverse to complete their objective (Internet access points, Trusted Internet Connections, Domain Controllers, etc.); or areas that, if damaged, will

hamper defensive forces in countering the intrusion (SOC analysis systems, perimeter and host sensors, log collection architecture, etc.).

Behavior refers to malicious activities in cyberspace. Data should be collected to observe those activities. For example, if the adversary can launch malware from ordinarily benign processes, defenders should capture data on process launches and the process's parent information from hosts within the terrain of interest. Another example is encryption of malicious command and control (C2) communications across the network. Collecting network communication information from the host may allow defenders to potentially mitigate this visibility gap.

1.3 Survey of Detection Methods and Data Types

Detection approaches include sweeping for IOCs and network security monitoring (NSM); anomaly detection; and TTP-based detection – with the majority of current data collection efforts focused on network sensors and perimeter proxies as opposed to host-based event data. Each of these approaches has benefits and limitations.

1.3.1 Indicators of Compromise

Prior to 2016, threat hunting processes appear to have been primarily organized around searching for IOCs; which include static characteristics of malware, such as hashes, filenames, libraries, strings; or gathering and analyzing disk and memory forensics artifacts.

A signature written to detect IP addresses, domains, file hashes, or filenames associated with malicious activity, without triggering on benign instances, is often very brittle to polymorphism, metamorphism and other implementation modifications which are relatively cheap for an adversary to use. David Bianco captured this through his *Pyramid of Pain* (FireEye, 2014). Defining those brittle signatures and indicators often requires extensive resources, through reverse engineering and static analysis, and are often dependent on detection through some other means (often after having been successfully used on by adversaries in other breaches and independently detected, reported, and disseminated). Thus, indicator sweeping fails to identify novel or changing threats that don't match known indicators, and only provides detection capabilities after the fact.

1.3.2 Anomaly-Based Detection

Anomaly-based detection employs statistical analysis, machine learning, and other forms of big data analysis to detect atypical events. This approach has traditionally suffered from high false positive rates, can require significant investment in large scale data collection and processing, and does not always provide enough contextual information around why something was flagged as suspicious, which can make analytic refinement challenging.

The benign activity of software, system administrators, software developers and everyday users across enterprise networks is often so variable across time, users, and network space, that

defining “normal” behavior is often a futile exercise.[1] The volume of data required to be processed for anomaly and statistical analysis can be prohibitive to collect and retain. There must be sufficient data collected, from a sufficient number of data sources and locations within an environment, to enable trend and statistical analysis. However, what is sufficient can vary greatly and is often unknowable in advance, making this type of detection hard to utilize and measure effectively.

1.3.3 TTP-Based Detection

Rather than characterizing and searching for tools and artifacts, a more robust approach is to characterize and search for the techniques adversaries must use to achieve their goals. These techniques do not change frequently, and are common across adversaries due to the constraints of the target technology. The MITRE ATT&CK™ framework[2] is an effective way to characterize those techniques. ATT&CK categorizes reported adversary TTPs from public and open cyber threat intelligence and aligns them by tactic category within the phases of the Cyber Attack Lifecycle[3].

[1] https://www.utdallas.edu/~muratk/courses/dmsec_files/oakland10-ml.pdf
[2] https://attack.mitre.org/ and Strom, et. al (2018)
[3] https://www.mitre.org/capabilities/cybersecurity/threat-based-defense

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	CredentialAccess	Discovery	Lateral Movement	Collection	Exfiltration	Command and Control
Valid Accounts	Trap	Scheduled Task	Process Injection	XSL Script Processing	Network Sniffing	System Time Discovery	Windows Remote Management	Video Capture	Scheduled Transfer	Web Service
Trusted Relationship	LSASS Driver		Extra Window Memory Injection	Process Injection	Two-Factor Authentication Interception	System Service Discovery	Third-party Software	Screen Capture	Exfiltration Over Physical Medium	Uncommonly Used Port
Supply Chain Compromise	Local Job Scheduling		Bypass User Account Control	Extra Window Memory Injection	Private Keys	System Owner/User Discovery	Taint Shared Content	Man in the Browser	Exfiltration Over Command and Control Channel	Standard Non-Application Layer Protocol
Spearphishing via Service	Launchctl		Access Token Manipulation	Access Token Manipulation	Password Filter DLL		SSH Hijacking	Input Capture		Standard Application Layer Protocol
Spearphishing Link	XSL Script Processing		Valid Accounts	Valid Accounts	LLMNR/NBT-NS Poisoning		Shared Webroot	Email Collection	Data Staged	
Spearphishing Attachment	Windows Remote Management		Plist Modification		Keychain	System Network Configuration Discovery	Replication Through Removable Media	Data from Removable Media	Data Transfer Size Limits	Remote Access Tools
Replication Through Removable Media	User Execution		Image File Execution Options Injection		Kerberoasting	Security Software Discovery		Data from Network Shared Drive	Data Encrypted	Port Knocking
Exploit Public-Facing Application	Trusted Developer Utilities		DLL Search Order Hijacking		Input Prompt	Remote System Discovery	Remote File Copy		Data Compressed	Multilayer Encryption
Hardware Additions	Third-party Software		Web Shell	Web Service	Input Capture	Query Registry	Remote Desktop Protocol	Data from Information Repositories	Automated Exfiltration	Multiband Communication
Drive-by Compromise	Space after Filename		Setuid and Setgid	Trusted Developer Utilities	Hooking	Process Discovery	Pass the Ticket	Automated Collection	Exfiltration Over Network Medium	Multi-Stage Channels
	Source		Service Registry Permissions Weakness	Timestomp	Forced Authentication	Permission Groups Discovery	Pass the Hash	Audio Capture	Exfiltration Over Alternative Protocol	Multi-hop Proxy
	Signed Script Proxy Execution		Port Monitors	Template Injection	Exploitation for Credential Access	Peripheral Device Discovery	Logon Scripts	Data from Local System		Fallback Channels
	Service Execution		Path Interception	Space after Filename	Credentials in Files	Password Policy Discovery	Exploitation of Remote Services	Clipboard Data		Domain Fronting
	Scripting		New Service	Software Packing	Credential Dumping	Network Share Discovery	Application Deployment Software			Data Obfuscation
	Rundll32		Launch Daemon	SIP and Trust Provider Hijacking	Brute Force	Network Service Scanning				Data Encoding
	Regsvr32		Hooking	Signed Binary Proxy Execution	Bash History	File and Directory Discovery	Windows Admin Shares			Custom Cryptographic Protocol
	Regsvcs/Regasm		File System Permissions Weakness	Rundll32	Account Manipulation	Browser Bookmark Discovery	Remote Services			Connection Proxy
	PowerShell		Dylib Hijacking	Rootkit	Securityd Memory	Application Window Discovery	Distributed Component Object Model			Communication Through Removable Media
	Mshta		Application Shimming	Regsvr32	Credentials in Registry	System Network Connections Discovery	AppleScript			Standard Cryptographic Protocol
	Installutil		AppInit DLLs	Redundant Access		System Information Discovery				Remote File Copy
	Graphical User Interface		AppCert DLLs	Process Hollowing		Account Discovery				Custom Command and Control Protocol
	Exploitation for Client Execution		Accessibility Features	Process Doppelgänging						Commonly Used Port
	Execution through API	Winlogon Helper DLL	Sudo Caching	Port Knocking						
	Dynamic Data Exchange	Windows Management Instrumentation Event Subscription	Sudo	Obfuscated Files or Information						
	Control Panel Items		SID-History Injection							
	Compiled HTML File		Exploitation for Privilege Escalation	Network Share Connection Removal						
	Command-Line Interface	Security Support Provider		Modify Registry						
	CMSTP	Screensaver		Masquerading						
	AppleScript	Registry Run Keys / Startup Folder		LC_MAIN Hijacking						
	Windows Management Instrumentation	Shared Applications		Launchctl						
	Signed Binary Proxy Execution	Re-opened Applications		Install.tlb						
	Execution through Module Load	Port Knocking		Install Root Certificate						
		Office Application Startup		Indirect Command Execution						
		Netsh Helper DLL		Component Firmware						
		Modify Existing Service		Indicator Removal from Tools						
		Logon Scripts		Indicator Blocking						
		Login Item		HISTCONTROL						
		LC_LOAD_DYLIB Addition		Hidden Window						
		Launch Agent		Hidden Users						
		Kernel Modules and Extensions		Hidden Files and Directories						
		Hidden Files and Directories		Gatekeeper Bypass						
		External Remote Services		File System Logical Offsets						
		Create Account		File Permissions Modification						
		Component Object Model Hijacking		File Deletion						
		Change Default File Association		Exploitation for Defense Evasion						
		Bootkit		Disabling Security Tools						
		BITS Jobs		Deobfuscate/Decode Files or Information						
		Authentication Package		Control Panel Items						
		Account Manipulation		Component Object Model Hijacking						
		.bash_profile and .bashrc		Compiled HTML File						
		Time Providers		Code Signing						
		System Firmware		CMSTP						
		Shortcut Modification		Clear Command History						
		Redundant Access		BITS Jobs						
		Hypervisor		Signed Script Proxy Execution						
		Component Firmware		Scripting						
		Browser Extensions		NTFS File Attributes						
				Mshta						
				Indicator Removal on Host						
				DLL Side-Loading						
				DCShadow						

Figure 2 MITRE ATT&CK Matrix

Adversary behavior-focused models like ATT&CK have been found to be very useful in defensive operations, helping to identify new adversary behaviors, helping prioritize detection for techniques utilized by multiple adversaries, and, in conjunction with data modeling, allows for identification of visibility and defensive capability gaps that orient around the threat to an organization. It allows defenders to frame detection hypotheses, focused on the adversary's actions and phases of their operations, that lend themselves to specific, implementable analytics within the defender's analysis platform (e.g. a security information and event management (SIEM) or other data analysis system).

A good data model for hunting will relate what objects and actions an analyst wishes to capture to the key data (fields, values, attributes) needed from the environment's sensors. It ties the data of what the sensor can observe to the actions and events the analytics are meant to identify and detect (see section 2.2.3 *Determine Data Requirements* for more information). One example of data modeling useful for hunting is the Cyber Analytic Repository (CAR) data model, which attempts to describe adversary actions in terms of the data required to identify those actions, irrespective of a specific tool or product (MITRE, 2015b).

The methodology proposed in this paper will utilize the ATT&CK framework, in conjunction with the CAR data model, as an example of a generic adversary model that attempts to identify all possible adversary behaviors to analyze and detect. Other frameworks may be used if they satisfy the requirements of identifying specific adversary TTPs that can be decomposed into actionable analytics within an analysis platform.

1.3.4 Network-Based Data

Traditionally, continuous activity monitoring has primarily focused on collecting and analyzing network traffic, usually focused at perimeter boundaries, as part of a Network Security Monitoring (NSM)-focused defensive operation. The historical focus on network perimeter monitoring developed in part from the convenience and cost efficiency of placing a limited set of sensors at a relatively small number of heavily controlled network gateways.

Network perimeter sensors provide little to no insight into adversary activity outside of the initial successful breach and data exfiltration stages, including especially lateral movement and privilege escalation within the compromised environment. Network sensors deployed within an environment may assist in mitigating some of these shortfalls but it can be difficult to deploy enough network sensors to comprehensively monitor any but the smallest enterprises. Used correctly, however, internal network sensing remains an important component of an enterprises defenses and can complement host-based sensing for TTP-based detection.

1.3.5 Host-Based Data

Host event data collection has generally been identified as the most desired data source for hunting (Lee & Lee, 2017, p. 16) but "many respondents feel that endpoint data is more obscure and harder to obtain" (Lee & Lee, 2017, p. 17). Recent advances in operating system capabilities (e.g. Windows 10 event tracing and event forwarding; auditd and the integrity measurement architecture (IMA) on Linux) show that host-based data of sufficient granularity and abstraction is increasingly available. There is a pervasive notion in the community that endpoint data is too

voluminous to collect and analyze. However, research into malicious file detection conducted by Invincia Labs (Berlin, Saxe, & Slater, 2015) suggests that 100-200 megabytes of audit data from Windows workstations per day was sufficient to detect 85% of the malware executed on a system. While this experiment was not conducted via a threat hunting effort but rather a machine learning one, it suggests an achievable target for scoping the volume of data collection and storage requirements.

1.4 Comparison of Published Methodologies

According to SANS (2017), few organizations utilize an existing hunting methodology, citing a lack of published or accessible methodologies. Through literature review, this appears to still be correct. Sqrrl has published a hunt methodology, described in the *A Framework for Hunting* whitepaper released in 2016, and a maturity model. Endgame has published a hunting process, titled *The Hunt Cycle*. (Scarfone, 2016, p.9) Beyond those published models, it appears that most organizations' hunting methods are defined internally (27.1%) or consist of ad-hoc processes without a documented methodology (45.1%). (Lee & Lee, 2017)

Sqrrl's methodology is organized around four phases: 1) Create hypotheses; 2) Investigate via Tools and Techniques; 3) Uncover New Patterns and TTPs; and 4) Inform and Enrich Analytics. The Sqrl methodology authors describe, in general terms, that one should start hunting with a hypothesis, based around proposed adversary activity one wishes to hunt for or around a portion of the environment one suspects the adversary may be operating within. It also provides a maturity model that would allow organizations to assess and develop their hunting capabilities. MITRE's work on TTP Hunt, using the MITRE ATT&CK framework, complements this methodology with additional implementation details.

Endgame's methodology, as described in *The Hunter's Handbook* eBook, is also organized around four phases: 1) Survey (a selected portion of the environment); 2) Secure (the hunted environment); 3) Detect (the adversary); and 4) Respond (and remediate the intrusion). The initial focus on gaining a better understanding of the environment, via Survey, is relatively novel in the literature, yet reflects general understanding among experienced hunting practitioners that the results of an organization's early hunting attempts will be focused around a better understanding of the environment rather than actually identifying an undetected adversary. Endgame then suggests securing the hunting environment by locking down lateral movement capabilities. Similarly to the Sqrrl methodology, this paper complements the Endgame approach with additional recommendations on organizing hunting hypotheses and information on adversary behaviors.

The methodology described in this paper differs from these methodologies by attempting to create a general hunting methodology focused on identifying adversary behavior, structured within an adversary model, that does not depend on specific tools or products but rather describes what data is necessary, what types of data should be available from sensors, and how to utilize that data with analytics to conduct a hunt.

2 Methodology

2.1 Overview

The approach to hunting described below has two components: *Characterization* of malicious activity, and hunt *Execution*. These components should be ongoing activities, continuously updated based on new information about adversaries and terrain. The flow of updates is visualized in Figure 3, the *V Diagram* below. There are three layers of related activities, focusing on the malicious activity, analytic processes, and data processes. Examples given in the following sections are described using a notional hunt team, composed of individual analysts and a team lead, to illustrate key points.

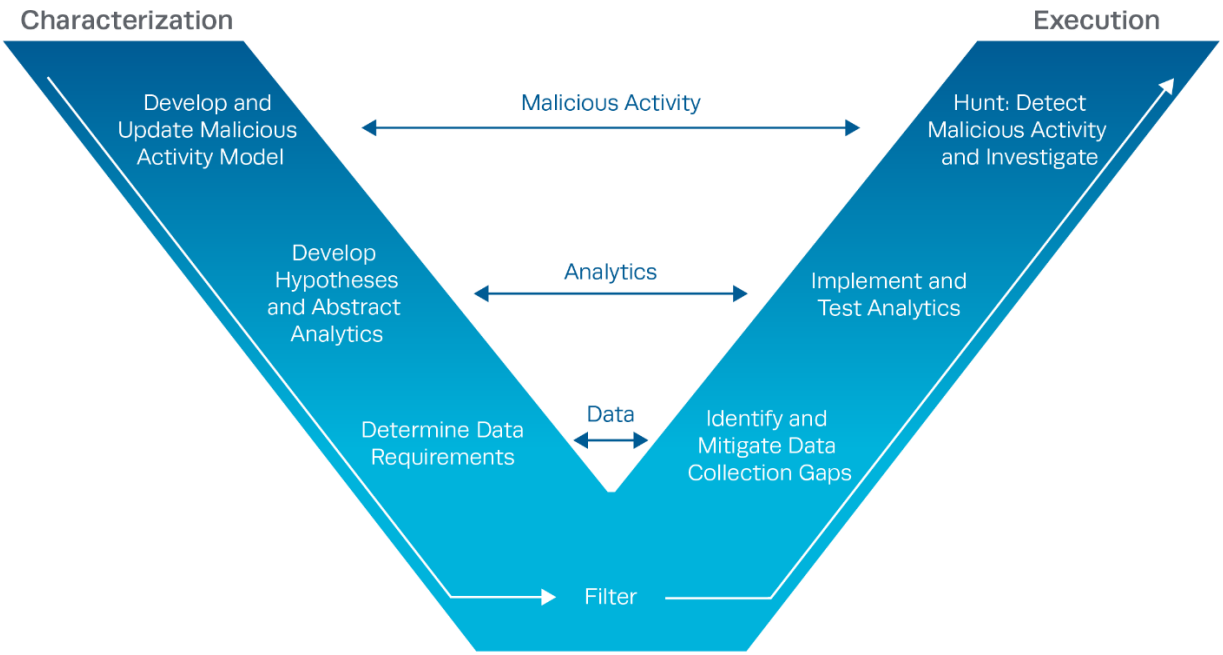

Figure 3 TTP Hunting Methodology "V" Diagram

Characterization of malicious activity starts with developing or updating the generic adversary model of behavior to identify all TTPs that an adversary may use – regardless of which adversary group, environment, or targeted network. For each TTP identified in the model, an analyst proposes one or more detection hypotheses that are formulated as abstract analytics. These hypotheses and abstract analytics are used to determine what data is necessary to collect. For each hunting operation, the hunt team should filter these data collection requirements and analytics based on the specifics of the terrain and situation of that hunt.

Execution employs the filtered data requirements and data model to conduct a gap analysis of sensors and data sources within the environment. If necessary, additional sensors (network or host-based) may be deployed at this stage to address visibility gaps. Once data is flowing into the analysis system, the analyst leverages the data model to implement analytics within the analysis system. The hunt team then executes the hunt by selecting specific analytics strongly associated with malicious behavior to try and obtain an initial detection. Analytic tuning and triaging

suspicious and correlated events to positively identify the presence of an adversary follows this initial detection.

Each of these steps are described in greater detail below.

2.2 Characterization of Malicious Activity (Left Side of the "V")

2.2.1 Gather Data and Develop Malicious Activity Model

Through cyber threat intelligence collection, threat information sharing by other organizations (e.g., FireEye reports, MITRE ATT&CK framework), and research efforts, the defensive operations community collects information on how adversaries behave across various terrain types (e.g., Windows Enterprise Networks, ICS/SCADA systems, infrastructure devices, mobile devices, Internet of Things (IoT) devices). It is important during this analysis to consider which aspects of adversarial behavior are transient, or easy for the adversary to change or mask, and which aspects of behavior are likely to remain constant or prove difficult for the adversary to change (e.g., TTPs). The focus is on information that can be converted into TTP-based analytics rather than brittle indications of compromise such as file hashes, IP addresses or domain names (i.e. focus on the top of David Bianco's *Pyramid of Pain*). This information needs to be organized in such a way as to facilitate filtering by dimensions in the analysis space (time, terrain, behavior), by the adversary, or the phase of the adversary's operation.

A common question asked at this stage is how to prioritize TTPs for analytic development. There are many possible effective methods and research to date has not highlighted one method to be better than another. Some methods to prioritize include:

- Based on TTP usage by adversaries. A rough approximation would be to count distinct references on techniques and built a heatmap on what's most prevalent across groups/software, either based on the examples already in ATT&CK or based on locally collected cyber threat intelligence and/or past incidents. For all of these usages, keep in mind that previous reporting is probably only against a subset of usage, and it should be used with caution and an understanding of the biases in the data (e.g., the data in ATT&CK itself is only based on openly-reported incidents, while data from past incidents may not include attacks undetected at the time).
- Based on currently available data. Start with which data types are available and work to build out to incorporate new sources. This is a very practical approach as collecting new data might require modifications to operational systems or purchasing new tools.
- Based on the adversary's lifecycle. Focus first on early stages of the adversary's lifecycle with initial access/execution/discovery tactics. Detecting activity at this stage may have larger benefit than detecting later stages when responsive action might be too late.
- Based on technique bottleneck. Start with what needs to happen and that most adversaries are likely doing (e.g., credential dumping, remote system discovery).
- Based on differentiation between malicious behavior and benign behavior in the operational network. There may be some TTPs used by adversaries which are known to be very unlikely to be used by the organization's intended users and system administrators. Indications of those behaviors should have very low false-alarm rates.

- Based on a combination of the above approaches.

2.2.2 Develop Hypotheses and Abstract Analytics

Based on this knowledge of adversarial behavior, the analyst proposes a hypothesis to detect that behavior in the form of an abstract analytic. For example, knowing that adversaries sometimes move files between systems using Server Message Block (SMB) protocol, and then execute them using scheduled tasks (*schtasks*), an abstract analytic might be to detect when a *schtasks* execution occurs as a result of a file moved through an SMB session. During the development of a hypothesis, analysts should be careful to avoid creating an analytic that is too specific to a particular instantiation of a technique by a particular tool. Ideally, hypotheses and analytics will be based on the behavioral invariants of a technique.

2.2.3 Determine Data Requirements

In order to hunt effectively, one needs to have data that adequately captures the activity of the adversary from data sources and sensors properly located within the terrain) to successfully observe it. Specific data requirements for hunting fall into two broad categories: collection requirements and modeling requirements.

To identify collection requirements, a list of required data and data sources should be created based on the set of abstract analytics developed. For example, in the analytic described above, data collection needs might include capturing network traffic and host logs associated with SMB, and contextual data associated with any invocation of *schtasks* on each desktop in the enterprise (e.g. which file is executed, the date/time for execution and the user associated). As a result; comprehensive data requirements can be aggregated across all analytics; linking each data requirement and the terrain type, adversary and lifecycle phase associated with that requirement.

Sensor and data source selection play a key role. It is helpful to consider the merits of which sensor or data source to select based on the amount of contextual information they provide the analyst balanced against the volume of data generated by each data source. It is generally true that more context equates to more volume (network bandwidth utilized for collection; storage, indexing, and analysis resources for processing), so it is unlikely to be possible to capture all possible data (full content collection from hosts and network devices) to support a hunt. By starting with an understanding of adversarial techniques and abstract analytics, an organization can reduce its data collection load by tailoring the collection strategy for the desired analytics. However, context is critical to effectively triage suspicious events and separate truly malicious activity from suspicious but ultimately benign activity. Therefore, it is important to collect sufficient data to enable analysts to make connections between analytic hits.

The diagram in Figure 4 illustrates the relationship between the amount of contextual information present from a data source, the generalized amount of data generated by the data source, and whether the data source is primarily host-based or network-based. The higher the data source is on the chart, the more likely it is to be able to capture the context needed for hunting. A successful hunt will involve correlating information from multiple data sources in

multiple locations (host and network) to create a comprehensive understanding of the activities taking place on the network.

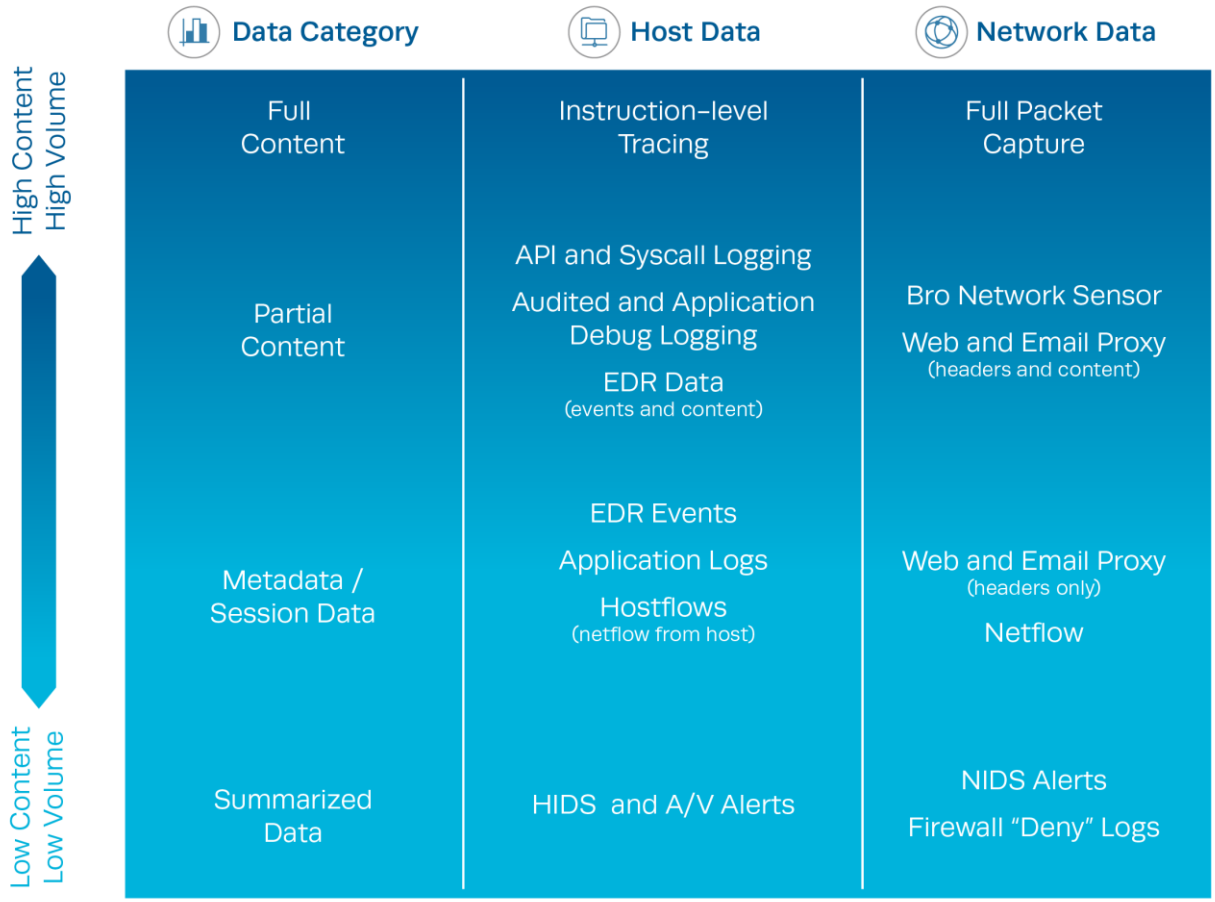

Figure 4 Context vs. Volume of Host and Network Data

Sensors that provide continuous activity, event, or content data are preferred over signature and alert-based ones, as the nature of hunting requires examining activity that was not initially detected as malicious. Many traditional sensors, such as signature-based host or network intrusion detection systems (HIDS/NIDS), focus on detecting very specific, discrete information present in an attack (usually some highly identifiable malware attribute). These types of sensors are generally not useful for hunting, as they are focused on automated detection of well-known malicious activity – there's no hunting involved – and the data generated by these sensors is generally limited to specific alerts. They do not provide contextual event and activity data around any suspicious activity being investigated. Host sensor selection will likely involve some combination of endpoint detection and response (EDR) agents, application logs, and operating system event logs.

Data from host, network, and application proxy sensors must be specific enough to be able to ascertain what occurred on a host or network (on that part of the terrain) but not so specific that the volume of data generated is too large to feasibly collect, aggregate, and analyze. For instance, network flow data may have limited value if it lacks corresponding application layer information.

Similarly, host-based sensors that may initially appear adequate might later be found inadequate due to an inability to represent fine-grained process detail (e.g., parent process, execution path, command line arguments, etc.). The goal is to, at a minimum, be able to link related events by causality to identify each major step of the adversary's actions. Events do not happen in isolation from one another, so the data should support identifying what came before as well as what happened next. These causal relationships can be succinctly represented, in Figure 5, by a simple state diagram, identifying major system components (processes, files, network flows, etc.) and how they relate to each other.

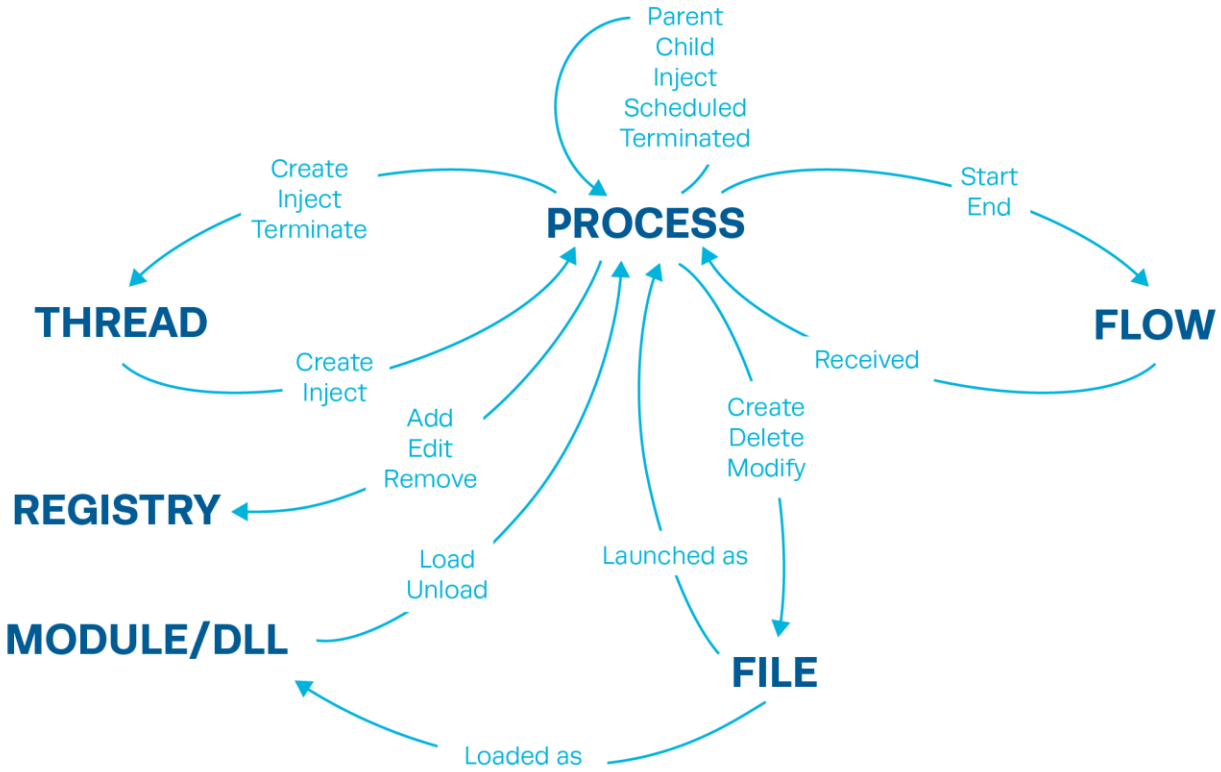

Figure 5 System Activity Relationships (based on CAR data model)

To help narrow the scope of what data is necessary, it helps to leverage a common data model. Good data model requirements for hunting will involve relating what system activity one wishes to capture to the key data (fields, values, attributes) needed from sensors. It ties the data of what the sensor can observe to the behavior and events your analytics are meant to identify and detect and aids analyst reasoning.

For example, the CAR data model (MITRE, 2015b) uses an object:action pair to describe some specific adversary behavior with the data fields linked to that object:action pair that correspond to the behavior. Figure 6 shows the process object within CAR, describing process activity that would be useful in detecting and tracking an adversary – in this case, process creation and termination events – and the data fields required. The hunt team can use the model to identify what information the sensors and data sources need to capture to enable analytics and the hunting mission.

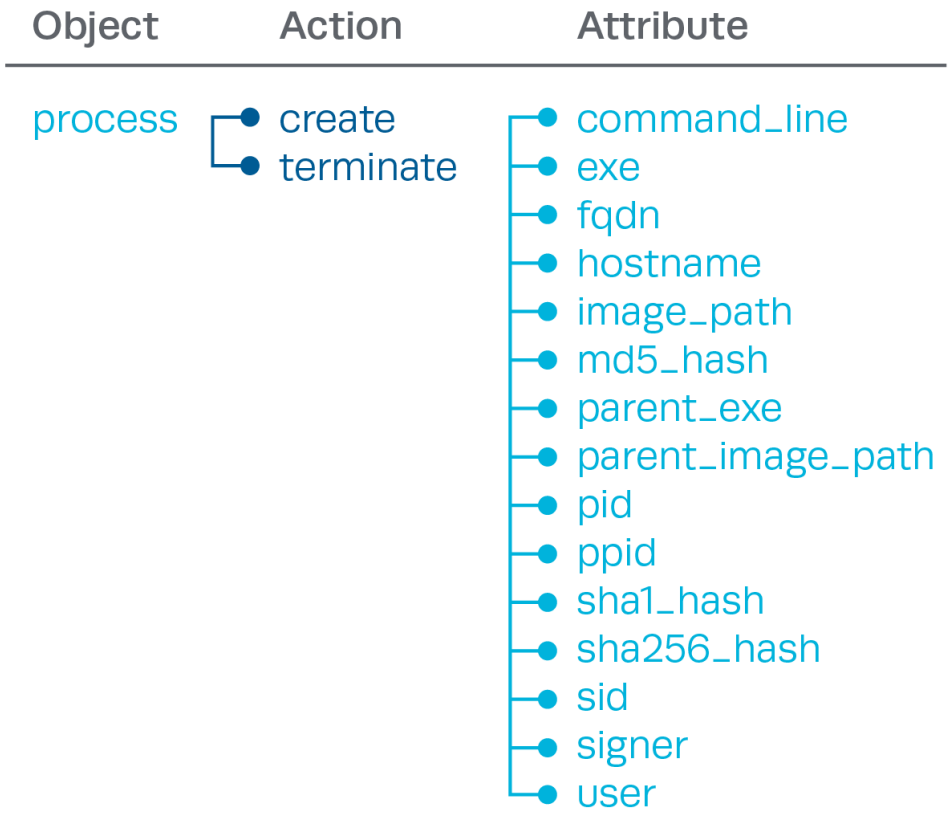

Figure 6 CAR Data Model

2.3 Filter

Upon completion of the *Characterization* phase, the team has a generic adversary model of all known TTPs; proposed hypotheses and abstract analytics defined to detect those TTPs; and data requirements and a data model necessary to enable those abstract analytics. Now the team needs to filter what they plan to analyze to hunt the adversary – essentially this is where the team initially constrains the analysis space to focus on what time, terrain, or behavior will enable them to start hunting the adversary.

Filtering on time is relatively straightforward – the team may have information that indicates an adversary was operating within the target environment at a certain time, so the initial window should be bounded around that time period. Another possibility is to start from the present and retrospectively look backwards for a finite period (e.g., two weeks prior to the start of the hunt activity). Often, the time window is automatically constrained by the retention period of the data storage and analysis system (e.g., the SIEM contains logs that cover a rolling 30-day period).

Once the terrain in which to hunt is known, the team can filter based on the types of systems and data available. For example, in an enterprise environment that is primarily Windows systems with some Linux servers, data requirements can be reduced to only data sources relevant to Windows and Linux systems. In other cases, such as building automation systems, data generated by both Industral Control Systems (ICS) devices and Windows devices would be

required to be collected. Given that the data requirements are tied to analytics that are then tied to TTPs, this automatically reduces the number of analytics necessary.

The team can filter on behavior – specifically selecting which adversary TTPs to detect within the environment. There are numerous ways to accomplish this but two general approaches to use are: filter based on the likelihood that the TTP will be easily identified as malicious relative to similar but benign behavior, or on the likelihood of a specific adversary group known to target the environment using the TTP.

Filtering on ease of detection requires knowledge about what activities are common within an environment and likely involve trial and error to reduce false positives. In a large enterprise network, there is significant variation in benign behaviors exhibited by users and system administrators in performing their duties, so what may be typical, benign activity for one user might be very unusual for another. Repeated hunting operations will help identify which behaviors are uncommon, and therefore more useful for detection within that environment.

Filtering on which adversary groups are targeting an environment may be useful, with some caveats. It is unlikely that an organization or environment will be targeted by a single adversary group, so filtering down to just known behavior of that group may cause a hunt to miss the presence of another adversary that has successfully compromised the environment. Additionally, adversaries can, and often do, adapt as new TTPs are identified. Filtering only on TTPs that were previously identified as associated with the adversary may allow the adversary to escape detection. Therefore, this type of filtering is more likely to be beneficial in *prioritizing* which TTPs to search for first, but *should not be used to deselect* TTPs from being considered during a hunt.

These methods of initial filtering will also be useful during the hunt for tuning and refining analytical results.

2.4 Execution Phase (Right Side of the 'V')

2.4.1 Identify and Mitigate Collection Gaps

2.4.1.1 Confirm Existing Data Sources – Presence and Validity

When first embarking on a hunt, and periodically throughout the hunt, analysts should assess how well existing data collection meets the requirements. For example, analysts might need to determine if the data are present, valid (free from configuration errors and adversary tampering), and collected across the terrain of interest continuously. One method to check that the data is present is simple frequency analysis of relevant event codes over time to detect periods of time when collection of that event may have been disrupted. Another way to perform a validity check is to compare results from different data sources to ensure consistency (e.g., host-based network connections corresponding with flow data from a network sensor). Frequency analysis of event counts by IP address or hostname can be used to identify coverage gaps across the terrain.

2.4.1.2 Deploy Required New Sensors to Fill Gaps with Existing Collected Data

One common problem with hunting missions is the lack of available data to observe the adversary activity from a part of the terrain that lacks sensor coverage. This could be an area of the network that doesn't have a network sensor positioned on it; a host that does not have adequate logging configured; or some other visibility gap. Hunt teams should assess what coverage is available within the environment and supplement that coverage with necessary configuration changes, centralized data collection, and deployment of additional sensors and capabilities to mitigate those visibility gaps. If, at any time before or during a hunt, significant gaps are detected between the desired and actual data collection, the team should assess how to handle each gap. When possible, new sensors should be deployed to fill the gap . The team should bear in mind that some sensors are less costly or easier to deploy than others. For example, *Windows* audit logging capabilities are often already present and can be activated with configuration changes, whereas EDR tools may require acquisition, deployment, and calibration to start collecting the right data. The hunt team should also consider operations security (OPSEC) in the deployment of new sensors, and balance the value of the additional data collected with the visibility of that sensor to the adversary. The deployment of new sensors might impact business or mission functions. The hunt team should be prepared to communicate effectively about the pros and cons of each aspect of their hunt plan relative to OPSEC, mission impact, and probability of successful hunting.

Note that sensor deployment and data collection that starts post-compromise may be less effective in comparison to continuous, ongoing monitoring due to the issues in covering the time domain mentioned above. Additionally, sensors deployed on already compromised hosts may not be able to observe activity effectively due to anti-monitoring and anti-forensics capabilities of the adversary's tools (i.e., *Defense Evasion* tactic in ATT&CK). However, it is unlikely that the adversary can subvert every host, network device, or sensor, if the sensoring coverage is comprehensive enough. In the case of an adversary having already gained a presence in an environment before adequate defensive sensing was implemented it can be more effective to search for the side-effects of an evasive technique or search within data sources that would be unaffected by that technique (e.g., searching for one-sided network connections between two hosts may indicate missing data from a compromised host's sensing).

2.4.1.3 Alternatives to New Sensors

If deploying new sensors is not possible or practical, the team should assess if other data collected can be used to fill the gap, perhaps with lower confidence or granularity of visibility. This can be done by mapping data sources to the analytics they enable. This mapping allows the team to assess the impact on the hunt operation due to the lack of a particular data source and adjust their analytics to adapt.

Knowing the blind spots with respect to terrain and time - which adversarial techniques are not visible due to a data gap and therefore have reduced analytic coverage – can help the team determine how to proceed and to communicate to the network owner(s) about the impact. If certain adversarial techniques are no longer visible due to the gap, the hunt team may need to adjust its overall analytic approach as they seek initial detections and connections between detected adversarial behavior. This could include modifying which behaviors are included in

initial detection analytics, or increasing tolerance for missing evidence in linking two suspicious events.

If certain areas of the cyber terrain are not covered by existing data collection, increased scrutiny can be placed on links between covered terrain and those blind spots (e.g., network connections made from covered systems to those without coverage). If certain windows of time lack data collection, links between events on either side of that window will be more tenuous. At a minimum, the hunt team should be aware of the visibility gaps and their impact on hunting results, and should communicate them to network owners.

2.4.2 Implement and Test Analytics

The abstract analytic, the data model, and available data sources can now inform the creation of an analytic within the team's analysis system. The form of the analytic may vary depending on the specific system used. For example, if the team is using Splunk, the analytic will be in the form of one or more Splunk queries.

The analytic should be written to specifically identify the behavior noted in the TTP and leverage the data model as much as possible. The risk of writing analytics without modeling the data first is that the analytic could be too specific to that environments devices' and configurations making it harder to re-apply to other configurations. For example, if the analyst models data on process creation from Linux hosts and Windows hosts using a 'process creation' alias, the implemented analytic can refer to 'process creation' without having to specify specific Windows event IDs or Linux events within the analytic itself. The analytic becomes more useful across terrain types and data types – ideally, one analytic to run and query regardless of terrain. This ideal may not be practical for all analytics but serves as a goal for implementation guidance.

Due to the varied nature of operating systems, various events may not be applicable across all operating systems. For example, capturing registry changes in Windows can be a way to detect a number of different adversary TTPs, however there is no corollary in UNIX/Linux. Despite this, there is still value in writing analytics with a data model in mind, rather than for specific tools or logs. By abstracting the analytic, it can be fed from multiple sources of information for the same kind of data. This can provide redundancy or be used in an environment where sensor deployment is not uniform or consistent (for example, across a large corporation with data from multiple subsidiaries).

It is important to note that analytics developed at this stage are not set in stone. Analytics development is an iterative process that requires frequent tuning and reevaluation of logic. Changes in the environment may cause certain analytics to be retuned, or new adversary TTPs may need to be compensated for. See (Strom) for more details on this process.

2.4.3 Hunt: Detect Malicious Activity and Investigate

Hunting is an iterative process that requires creativity and flexibility. It is enabled by a core sequence of steps that provide a foundation for that flexibility. The flow chart in Figure 7 below describes that core sequence. It begins with collected data and knowledge of malicious TTPs and illustrates fundamental processes to leverage that knowledge to filter the data efficiently and find

the malicious activity. Once that activity is sufficiently understood, costs can be imposed on the adversary. Each step in this process is described in greater detail in the following sections.

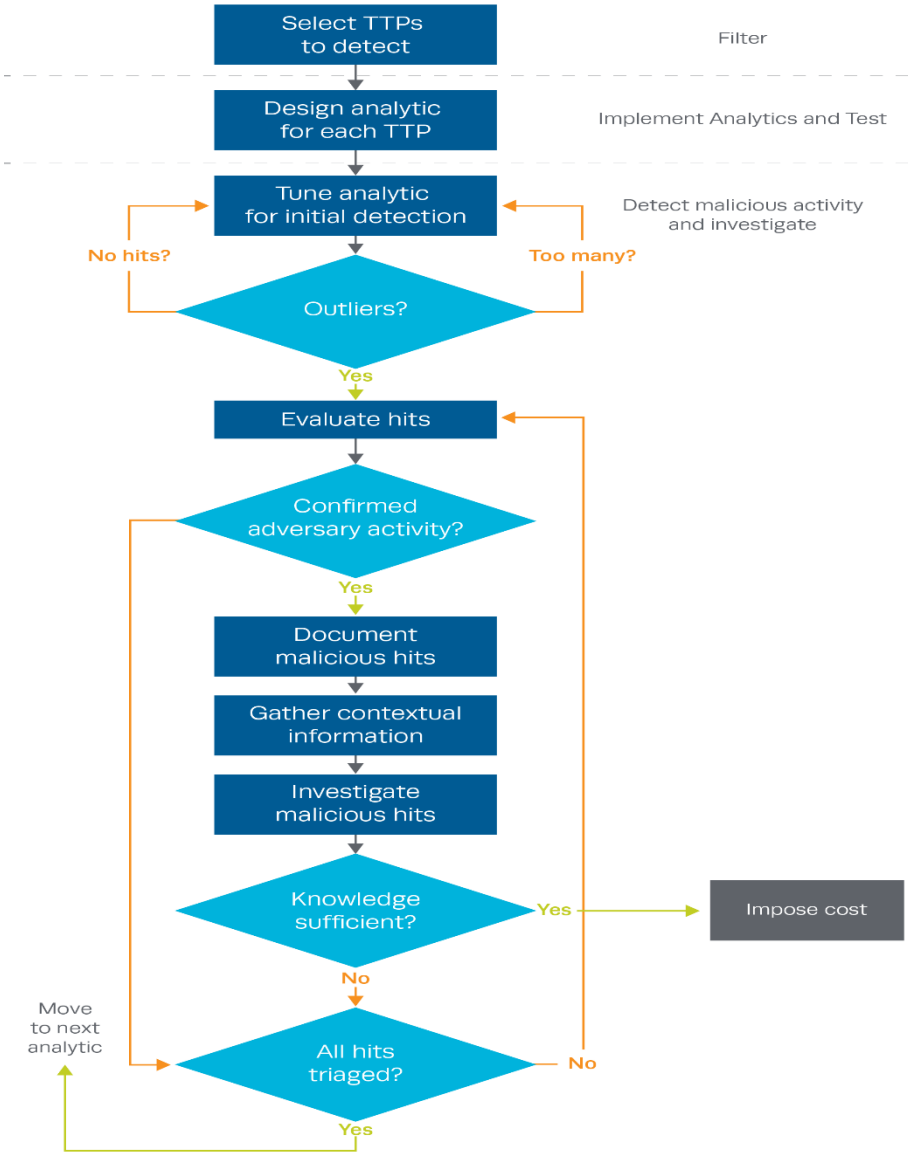

Figure 7 General Hunt Process Flow

2.4.3.1 Tune analytic(s) for initial detection

The first challenge for an analyst is to tune the analytic efficiently to the subset of hits with malicious activity. Narrowing the space across which results are queried will reduce the total number of events to be analyzed. Broadening it may result in greater total events, but it may also reveal patterns that would otherwise elude notice. It may be useful to count the number of occurrences of events for a unit of terrain over a specified time period (e.g., on each machine, over the course of one day). This results in three-dimensional data which can be represented as a heat map where x and y axes are time and terrain, respectively, and the color of each square corresponds to the count of that list of behaviors for that terrain in the timeframe specified.

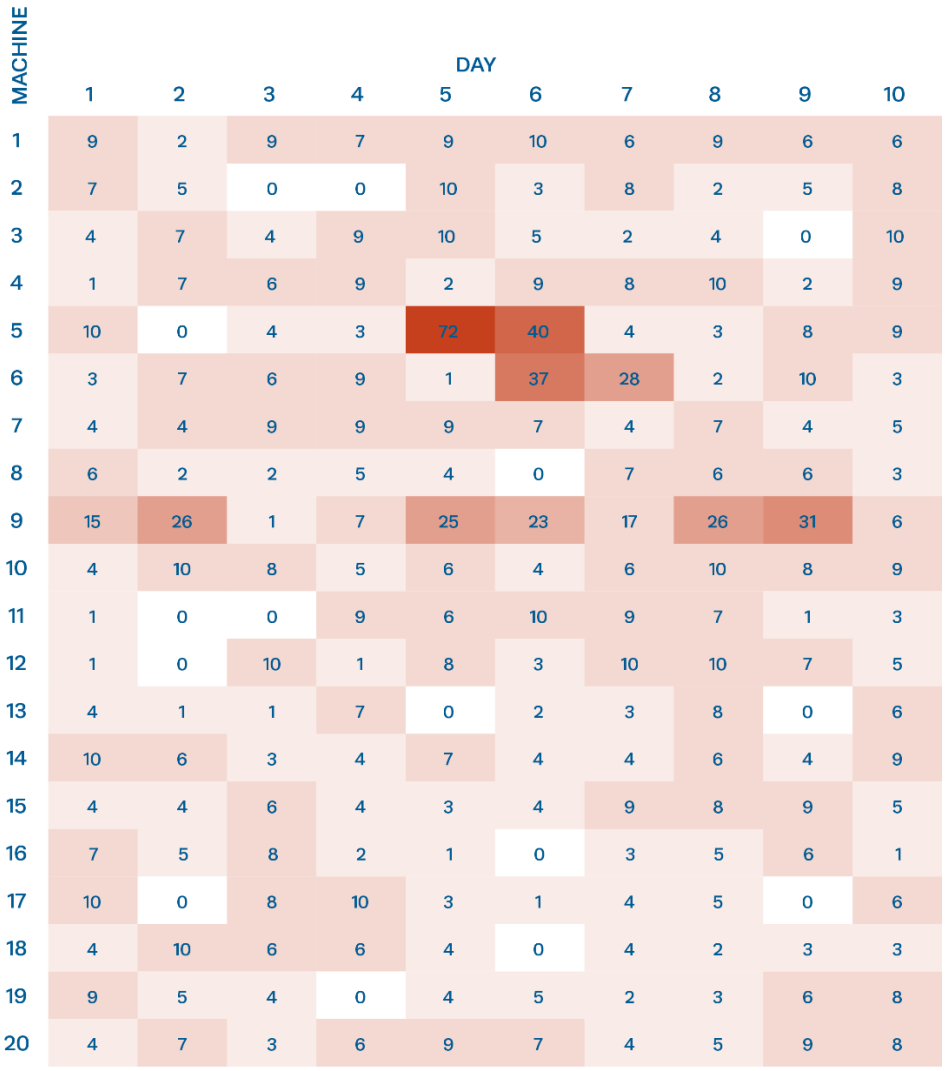

Figure 8 Heat Map

The heat map in Figure 8 represents the number of instances of a set of behaviors (those associated with adversary behavior, but not prevalent as benign in this network) that occurred on each machine for each day. This heatmap enables the analyst to quickly focus on Machine 5, day 5 as a lead to pursue.

Switching the axes of this heatmap around also yields interesting results. For example, switching the Terrain and Behavior axes produces a map detailing the prevalence of certain behaviors occurring throughout the network. This could be used by an analyst to identify instances where specific behaviors are increasing (or decreasing) and may even reveal the overall flow of an attack. For example, during Day 1 of an attack there was a surge of behaviors related to the *Discovery* tactic. This was followed by a surge in events related to *Lateral Movement* in Day 2 and finally a surge in *Exfiltration* events in Day 3.

Each of these behavioral analytics will have a false positive rate. We recommend beginning with analytics with a relatively low false positive rate individually, however this is difficult to know in advance of an event. Terrain specific knowledge could be used here to inform the choice of analytics. For example, the hunt team could exclude the Sysinternals tool *psexec* from the list of commands they are searching for in a network with frequent and benign use of the *psexec* command.

Constraining the analysis space in terms of behaviors can be challenging. For example, one could anticipate that users rarely if ever use Remote Desktop Protocol (RDP) as part of their daily work. Upon analyzing the logs, however, the team may find that there are in fact several users who do so every day. To compensate for this, the hunt team may need to adjust certain aspects of the behavior they are looking for. Adjustments could include excluding behaviors that are frequently seen in benign usage (and thus constitute a large number of false alarms), or reducing the number of behaviors searched for by the analytic (e.g. removing an executable that is observed to be running across the entirety of the environment).

The analysis space can also be tuned based on known good behavior. This decision could be informed by open source research into standard behaviors things like applications and protocols or by asking the network owner or administrator if there is known-good activity on the network that is likely to be observed using a given analytic. After several rounds of recalibrating the analytic, it can be beneficial to ask the network owner or system administrator about the observed behavior and if they can identify it as benign.

To further refine the search for malicious activity, the hunt team can modify the unit of aggregation in time or terrain until this heat map shows significant outliers relative to ordinary background activity. For example, one could begin with a time unit of one day and a terrain unit of one machine. Depending on the situation, a time unit of one hour or one week might be more effective at separating data of interest from false alarms. Similarly, grouping by subnet or username might be more effective than grouping by machine for the terrain dimension.

Constraining the analysis in the time dimension to shorter durations might help detect adversarial activity that is pervasive across the terrain but is concentrated in time (e.g., a massive initial infection, reconnaissance phase or exfiltration). Conversely, limiting the analysis to smaller units of terrain could help highlight more targeted adversarial activity across a long period of time.

Constraining the analysis in the terrain dimension can also be productive for the analyst. Analyzing everything across the operational environment is impractical, so the hunt team could identify the networks, devices, applications and processes that are most critical to execution of an organization's mission. Another means of constraining the terrain dimension is to assign different members of the team to focus on different segments of the terrain.

An additional approach to focusing analytic efforts is to prioritize investigating chains of connected analytic hits over individual detections. Suspicious behavior that shares a common process lineage with other suspicious behavior is likely more interesting and worth pursuing first.

Occasionally analysts may find themselves in a position where an analytic fails to return a useful result. This does not necessarily mean the logic behind the analytic is flawed or that there is no malicious activity. It may be the result of over-tuning of the analytic to be too specific. Depending on the situation, any of the three dimensions could be relaxed to reveal activity. For example, the time period could have been too narrow, the adversary might not have reached that portion of the network yet, or the adversary is not utilizing the technique the analytic is attempting to reveal. Whatever the case, incrementally expanding the scope of what the analysts are looking for can help reveal additional information without overloading the analyst with too much data.

2.4.3.2 Evaluate Hits

Once the number of events (or "hits") generated by a given analytic is reduced to a number small enough to devote some hunt team resources to pursuing each, the hunt team needs to resolve each of the hits returned. Events belonging to an outlier group are not necessarily malicious, so each one needs to be evaluated in depth. The methods used for evaluating results do not necessarily follow a prescribed order; the analyst decides which methods to pursue based on available information, experience, and expertise.

Once suspicious activity has been identified, widening the aperture (across time or numbers of devices) to generate a broader data set can help provide the context needed to determine if the activity is malicious. For example, suspicious activity on one machine might actually be benign if the same activity occurs on all the machines in that network, and has occurred for a considerable amount of time. In this example, asking the network administrator and/or user to identify this activity could help determine if the behavior is malicious.

Some events will require deeper inspection to make a determination regarding maliciousness. What form this takes depends heavily on the event in question, but two examples are parsing out the full command line from a process creation and extracting data communicated over a network connection. As a hunt team's processes mature, these cases will ideally decrease as the team becomes more familiar with the kind of data required by the analysts.

Contextual information is often needed to determine if an event is malicious or not. Adversaries do not perform actions in isolation and thus the traces of activity they leave behind do not exist in isolation either. There will be a chain of causality to follow that can be used to connect seemingly disparate events. Therefore, if the analyst can draw a direct connection between the event under investigation and another event or piece of intelligence that is known to be malicious, the certainty that this event is also malicious increases significantly. For example, a command prompt was observed running an executable that, while unusual, is not in itself malicious. However, upon examination, the parent process of that command prompt is discovered to have been spawned by a previously identified malicious executable. Additionally, the user account responsible was also previously identified as executing malicious programs. For these reasons, an analyst could reasonably deduce that the event currently in question should be considered malicious as well.

2.4.3.3 Possible Causes of False (Benign) Hits

Not all activity identified by an analytic is necessarily attributable to an external adversary. Three possibilities to consider are that the activity is legitimate, if uncommon or unusual; the activity is caused by the hunt team itself; or the activity may be an insider threat.

Analytic findings could be the result of legitimate, explainable activity. For example, system administrator activity is one of the most common sources of false positive events. Administrators frequently perform activity that resembles many techniques found in the ATT&CK model such as *Lateral Movement*, *Account Manipulation*, *Scripting*, and *Data Compression*. Administrators are also often responsible for deploying new software to the environment, which can cause unexpected events from both host-based and network data. Ideally, these kinds of planned changes to the environment will be coordinated with the hunt team so that they are prepared when the deployments happen but that is not always feasible, so the analysts need to prepare for this eventuality. These kinds of issues may require the analyst to inquire about the activities of administrators and/or individual users to deconflict results.

Software developers can also introduce unexpected behavior to the environment. For example, a web development team may be standing up and tearing down web servers multiple times a day and running performance tests against them, causing huge spikes in traffic to specific addresses. Alternatively, a team doing research into new adversary detection methods could cause instability with endpoint sensors and as part of their testing may create artifacts of some of the same attacks that they are trying to detect.

System or service misconfigurations can cause false positives in hunting analytics, and often go unnoticed until the hunting activity uncovers them. For example, tools can have inconsistent configurations, servers may have the incorrect auditing policies being enforced, and Domain Name System (DNS) servers can be misconfigured. The analyst needs to be prepared to identify instances where such is the case and to notify the party responsible so it can be addressed. This kind of issue could have the unintended side effect of changing the baseline of the environment, so any anomaly-based analytics in use may need to account for that fact.

Analysts must be cognizant of the possibility of detecting their own hunting activities or sensors rather than the adversary, because various methods used by adversaries can be used by analysts to collect and aggregate data. For example, some teams may use *PowerShell* scripts running as administrators to collect data from endpoints or they might run a vulnerability scan to scan for misconfigurations or vulnerabilities on the network. For this reason, teams should be aware of their footprint in the environment so if they are the cause of what looks like an event it can be quickly dismissed. This use case also exemplifies the need for communication within the team so team members are aware of what each other are doing and can quickly deconflict results. Additionally, this could occur when there are multiple defensive teams operating within the same environment. An example of such a situation would be where an external team comes in to augment existing manpower. If the two teams are not properly coordinating activities with each other, they run the risk of both duplicating effort and tracking each other rather than the adversary.

In rare cases, the activity may be related to an insider threat. In these cases, the hunt team might need to involve law enforcement and/or the organiation's counterintelligence or insider threat tracking group. Behavior that may initially look like an insider threat, however, may come from a variety of motives, ranging from ignorance of accepted policies, to an over-enthusiastic can-do attitude, or even pressure from management to willfully break from policy. Technical and management responses to each of these possibilities are outside the scope of this document and need to be addressed on a case-by-case basis but should be deconflicted to ensure such activity is properly handled.

2.4.3.4 Document Malicious Hits

If the detected event is determined to be malicious then the it should be captured in such a way that the information can be shared between team members as well as other parties with an interest in the investigation (e.g., management, other defensive teams). There are numerous ways that this information can be captured, here are some examples:

- Adversary Timeline - The Adversary Timeline is a simply a list of observed activity in chronological order. The list should contain more than just the event that was observed, but also contextual information like the user (if any) and host/IP address responsible. By adding this additional information, analysts can gain a greater appreciation of how the events are related. Once enough events have been identified the team should consider trying to group the raw events into segments of activity. Doing so will help the team gain context of the activity which may aid in understanding the overall adversary campaign.
- Host List - A list that contains relevant information regarding the various hosts that have been identified as being related to confirmed malicious activity. Some of the information that a team would want to capture is:
  - Hostnames
  - Users
  - Owners
  - IP Addresses
  - Why this host is on the list
- User List - A list that contains information on users that have been confirmed as performing malicious activity. Additionally, consider adding users whose credentials may have been compromised, even if those credentials have not been tied to malicious activity. This may also include relevant information about the user that the hunt team may find useful like:
  - Contact Information
  - Supervisor
  - Location
  - Role
  - Assigned Machines
- Malware List - A condensed list of the malware that has so far been found within the environment. Any utilities or built in programs that are being used by the adversary can also be tracked here. Some of the information that should be captured here is:
  - Malware/Program Name
  - Any Aliases

- General Description
- Other Pertinent Details about the Malware

- Activity Graph - A map that describes the chain of activity between the various hosts identified. The purpose is to provide a visual representation of the malicious activity occurring on the network. The important details to capture on the graph are:
  - Hosts that have been confirmed as having malicious activity take place on them. For this purpose, the hostname, rather than the IP address, is more useful as a given computer could have multiple IP addresses assigned to it for many reasons. However, there will likely be instances where an IP address is all that is available to use (such as an external C2 server).
  - Network connections made between each of the hosts to show where the adversary pivoted in their operation. Capturing every network connection made between each host is unrealistic, so only a select few should be rendered. Initial malicious connections between two hosts are important to note, as this information helps establish how the adversary is moving around the network. As part of the connection information, it is important to capture the time/date of the connection as well as the protocol or method used.
  - User credentials used (if any) are important to note as well. If legitimate user credentials are being used then noting that can help inform directions that the hunt team needs to investigate further in. For example, if a user is observed making a malicious RDP connection to a host but no information regarding what that user did on that host has been found yet, that is something that should be investigated. Conversely, if a user's credentials are being used maliciously to navigate the network then the hunt team needs to trace back those connections to try and find the moment where those were compromised.

2.4.3.5 Gather Contextual Information

Contextual information can be extremely important, as outlined above, and for that reason collecting it is of utmost importance. Not only does it aid in understanding events that have been identified as malicious, but it can be used to drive direction for further investigation. Often, the most valuable information is that which can help to establish a chain of causality: what caused the event in question, and what did the event cause in turn? By capturing these pieces of information, the team can focus their efforts on events that are directly tied to a known bad event. Events that precede a known malicious event should be considered very suspicious and events that were caused by it should be considered malicious. The following paragraphs, while by no means exhaustive, highlight some of the things that an analyst should capture in relation to a given event. They provide a starting point for developing the team's own methods of connecting known malicious events to understand what happened.

Related Processes

Identifying related processes can be an invaluable tool. Through these relationships it is relatively easy to establish chains of activity. The most important pieces of information to capture in this regard are "child" and "parent" process name/image paths, process ID's, and command lines. Additionally, the full command line of processes should be captured if possible, as it often contains invaluable information about the event. The arguments contained within

show how exactly that executable is being used and may also reveal additional information like any files that may have been used/modified or network connections that should be investigated further.

Network Information

Any network-related information that can be tied to a given event is also very important to capture as it will potentially reveal whether the event is part of a broader campaign and how it fits into the bigger picture. Without this context, an analyst is left with isolated series of activity with no direct ties to events happening on other hosts. The primary pieces of information that an analyst needs to capture relating to network activity are any IP addresses, ports, and any details regarding the content of the communication itself. The last item in that list is difficult to define as it may vary considerably based on protocol and available information. For example, if the analyst observes a Secure Copy Process (SCP) create to a remote address, then the analyst will have information regarding the file being transmitted. If, however, the analyst's visibility is limited to just netflow events, then the nature of the file being transmitted may be impossible to discern. Resolving any IP addresses identified to hostnames will also be beneficial for further investigations as well as coordinating with other team members.

System Files

Even in "file-less" attacks, adversaries will almost certainly interact with files on a system at some level. For example, adversaries may exfiltrate a user's documents or run an executable that, while an appropriate process for a typical Windows operation, is being run from an unusual directory. As an investigation progresses it is important to keep track of pieces of information that are tied to relevant files. Ideally, these would be captured in a standardized data model, however some items that can be tracked are the file name, the file path of the executable, a hash of the file (especially if it is a binary or executable file), and any timestamp information. Some pertinent types of files include email attachments preceding other observed activity, creations/deletions/modifications of files around the time of other events, and any files that are directly observed as being part of an event itself (e.g., any found within the command line of a malicious process start, or observed being transmitted over a network connection).

User Information

User information can provide additional context regarding the adversary's activity. Not only can it reveal related information from the same data source, but it can be used to pivot across many of the host-based objects found in the data model. It can be used to identify additional processes being run by the same user, to look for files that that user was responsible for editing, as well as establish boundaries of activity by looking at log-in and log-out times and seeing how those log ins were accomplished. Other compromised hosts can also be identified by looking for the same activity. If the activity appears to be the same on both hosts, further investigation is likely warranted.

2.4.3.6 Investigate Malicious Hits

To pursue a malicious hit, the hunt team should investigate both backwards and forwards to find the activity which caused the hit (ideally back to the initial infection), as well as subsequent activity to determine the scope and scale of the adversary's actions.

In most cases, to begin pursuing the adversary, we recommend working backwards to find the causes of the detected event. This will help determine the full scope of the activity, attribute the events to a specific adversary group, and gain the most useful knowledge for planning decisive response action. Ideally, the hunt team will have the required data collection and analytic capability to determine each link in the causal chain of events leading to this initially-detected event.

For example, on a Windows operating system, the responsible process could be found through identifying the parent process, *schtasks* command that scheduled this process start, the user event that triggered process start, or other methods as enumerated in ATT&CK's *Execution* Tactic. To trace the chain of causal execution across network traffic, the analyst might look for *Lateral Movement* methods like *Remote File Copy*, *Exploitation of Remote Services*, or other methods.

If no causal events are found, the analyst will need to relax the requirement for finding evidence of each link in the causal chain. The analyst should consider the range of processes, systems, etc. that could have resulted in the event under consideration. For example, recent network connections, other activity by the same user or machine in the recent past, or other machines exhibiting identical behavior (e.g., same command line or network traffic).

In parallel with, or after sufficient information has been obtained regarding causally preceding events, the hunt team should investigate caused or related subsequent activity. Similar to the investigation of preceding events, analysts should look first for evidence of directly-caused activity such as child processes, file creations, or opened network connections. When needed, the analyst should expand the investigation to include other machines exhibiting identical behavior and other suspicious files, processes, or activity on the same system. As the investigation proceeds, analysts can consider the direct descendants of known-malicious activity to be malicious, while considering processes with a common parent as only suspicious pending further investigation and context.

Throughout these pursuit investigations, analysts should continually refine the characterization of findings. As they gather more information, they should update a common knowledge repository (e.g., textual reporting, graph of activity) about the currently known chain of events, to include information regarding whether they are indicative of a specific set of adversaries, whether this activity is indicative of a certain stage in the Cyber Attack Lifecycle, and adversary intention. As new information is added to a shared repository, the team should also regularly determine what gaps in knowledge and/or visibility should be filled next and who and/or what could help fill them.

2.4.3.7 Identify Similar Behavior Across the Network

Looking for similar behavior across the network may reveal other instances of compromise that were initially missed. What exactly an analyst might look for is dependent on the event that is currently being looked in to, but some examples include:

- After successfully identifying an executable being used maliciously with specific arguments, those arguments are used to identify other instances of that executable being used even if it is under a different name.

- A connection is made over a specific port, which is then followed by the writing of a file that has been discovered to be the 2nd stage payload for the adversary's malware.
- A malicious instance of encrypted file compression is observed on one host is observed. While this may not be unusual for the environment overall, there were several other instances of the same user using the exact same syntax across multiple machines.

2.4.3.8 Respond

Throughout, the team must be mindful of the courses of actions possible for responding to the intrusion under consideration by the network owners, and tailor the investigation accordingly. There may be different choices made depending on whether the intent is to determine the full scale and scope of the intrusion versus quickly attributing the activity to an adversary group. As a result, the team may alternately prioritize finding the source of the activity, finding the subsequently-targeted systems, or performing deep forensic analysis to better understand the characteristics of the activity or artifacts likely to aid in attribution.

Over time, the knowledge gained by the hunt will be sufficient to make decisions on courses of action (e.g., quarantine, movement of the adversary to a deception environment, placement of honey credentials or misinformation, or perimeter blocking). This may occur when the full extent of the adversarial activity is known, or when the defensive team's knowledge and ability to effectively defend and respond can render the adversary's attack ineffective. The hunt team must strike the right balance between waiting too long to act, and acting prematurely. Too much emphasis on learning the full extent of the activity may hamper timely responsive action. Acting before sufficient knowledge is gained could result in tipping one's hand to the adversary without having significant impact on their presence in the network, or their ability to accomplish their objectives. This is a strategic decision which should incorporate an understanding of the adversary's activity, but also their intent and capabilities as well as the potential or actual impact to the defended environments. This is a ripe area for future research.

2.5 Report

There are several reporting requirements that should be addressed as part of planning for, conducting, and concluding a hunting operation. When planning a hunt, communication and reporting channels will need to be created for all stakeholders. These will include everyone in the hunt team's management chain and the network owner, especially for hunting on an environment that is not owned by the hunt team's organization. Key items to report are the general timeline for the hunt, what phase of the timeline the team is in, any confirmed presence of an adversary, systems affected (both systems that are being investigated as well as known compromised systems) and what damage or risk is currently posed by the adversary. In many cases, this will be an assumption based on available data. Avoid excessive speculation on the adversary's intent and capabilities, but instead focus on what facts have been uncovered from ongoing analysis. Be sure to establish regular update cycles, so stakeholders know when to expect new information.

Reporting also entails knowing the purpose(s) of the hunt – if the hunt is for remediation, reporting should be tailored towards informing stakeholders and remediation personnel where to pre-stage remediation capabilities and what the full scope and scale of the intrusion is to enable

decisive action. Communication channels separate from the environment being hunted on should established to avoid alerting the adversary and allowing them to react to hunting efforts.

3 Best Practices and Recommendations

To implement this methodology, organizations should follow these recommendations in operations, intelligence, workforce development, and capabilities. This methodology also has significant implications for industry partners.

3.1 Implications for Operations

Operations to detect adversarial presence in cyberspace should be oriented around a clear and up-to-date understanding of the TTPs employed by adversaries on the types of technologies being defended. The de-facto industry standard for representing these TTPs is MITRE's ATT&CK Framework. Operators should gain and maintain understanding of those TTPs and continually reassess their sensor and analytic posture. Analytic development should be viewed as a continuous process of design, testing, tuning and employment. As new adversary techniques are discovered, new analytics should be developed. Those analytics must be tested for both recall and precision. Recall testing can be done in a test environment (e.g., a VM or test enclave) to ensure that when the adversarial technique is present, the analytic will fire. Precision testing requires realistic background activity to assess the false alarm rate for the analytic. In addition to testing for recall and precision, analytics should be tested for robustness to evasion. Threat Emulation (or Red Teaming) can be employed to help assess how well the analytic performs throughout this iterative process by providing known instances of the malicious technique using various instantiation mechanisms. Once an analytic has been developed, tested and tuned to maximize recall, precision and robustness, it can be deployed to the operational environment for continuous monitoring in conjunction with existing analytics.

In the course of analytic development, testing and deployment, additional sensing requirements may emerge to support the analytic. These requirements should be captured, documented and fulfilled in the operational environment to support the employment of those analytics. Specifically, it is recommended to begin by configuring host-based collection of the following types of events along with their meta-data[4]:

- Process creation and termination
- Log-on events (remote and local)
- File creation, modification and time-stomping events
- Driver and module loading and unloading events
- Registry modifications
- Service and thread creations and deletions
- Network activity and associated process data

Where possible, network-based collection should be used to support host-based collection, and centralized to support correlation and analysis across devices and areas of operation.

[4] https://car.mitre.org/data_model/

Over time, defensive analysts will use this data and the technique-oriented analytics to tune their overall analytic approach to differentiated malicious activity from benign on their particular systems.

3.2 Implications for Intelligence and Threat Information

There are many excellent sources of information about adversarial activity available online and through cyber threat intelligence subscriptions. This is crucial information to start and continually refine defensive approaches. Organizations should devote resources to researching and maintaining an up-to-date understanding of adversarial TTPs through open source and/or subscription feeds. If an organization has the opportunity to gain insight into adversarial activity directly, keep TTPs in mind during analysis in addition to extracting IOCs such as file hashes, IPs, and domain names. Just as many organizations currently utilize IOCs for blocking future activity, TTPs observed should be included in future analytic and data collection efforts to ensure they are detected even if the adversary modifies their IOCs.

3.3 Implications for Workforce Development

Analysts should be trained in adversarial TTPs and how to develop robust analytics based on that knowledge. Red Teams should be trained on methods to implement known adversarial techniques in a variety of procedures to evade brittle analytics. Malware and threat analysts should learn how to find, extract, and describe adversarial TTPs in addition to IOCs as they study incidents and malware.

3.4 Implications for Capabilities

There are many built-in, open source, and commercial tools available to enable TTP-Based Hunting. By taking an Adversarial TTP-Based Hunting approach, an organization can assess any tool according to how well it helps provide visibility across adversarial techniques. Some of the key attributes to consider when choosing a set of tools for monitoring, prevention, analysis and response include:

- How many adversarial techniques can be detected or mitigated by this capability? Keep in mind that two tools which each provide a lot of visibility, but across the same techniques, might not be as valuable as two tools which provide visibility across a complementary set of techniques.
- How well does the capability match the skill level and resources available to the organization's hunt team? Some capabilities make deployment, analysis and response easier than others. In some cases, there is a trade-off between up-front costs and the costs of operation and maintenance. Often, there is a trade-off between ease of setup and operations, with flexibility of employment.
- How flexible is the tool to the addition of new analytics or data? Some capabilities make it easy to employ the collection and analysis envisioned by the vendor, but difficult to add new analytics.
- Does the capability send data outside the organization's environment? Some capabilities collect data and send it to a central or cloud-based analytic engine. This approach enables

the organization to benefit rapidly from insight gained at other organizations protected by that vendor, but also requires trust in the vendor to safeguard the data.

- Is the capability a monolithic stack of sensors, analytics, visualizations and reporting, or does it specialize in one aspect? Full stacks of capability are often easier to set up and maintain, but might not interoperate with other capabilities of interest (e.g., other sensors or analytic platforms).

3.5 Implications for Industry

There are several areas in which commercial and industrial partners in the defensive cyber operations community can enable TTP-based hunting, relating to platform development, data generation, interoperability, data analysis, and threat information sharing.

Platform developers should, where possible, create built-in capabilities for generating the event data needed by defenders to support TTP-based hunting and other adversary behavior-focused hunting strategies. These capabilities should focus on generating events related to the major system activities (see Figure 5, System Activity Relationships) including data elements that allow each event to be directly related to parent or child events. Some systems already have native event data generation capabilities of varying maturity. For example, Microsoft's Windows 10 enables event logging for process creation events, network connections, and other core system activities. The Auditd and Integrity Measurement Architecture capabilities for Linux also allow for relevant data to be generated. These efforts should be extended to allow for more granular specification of events of interest to be generated and filtered. Other platform developers, particularly in the mobile device and Internet of Things markets, cloud service providers, and network device manufacturers should look to integrate generating this data natively as well. For platforms that do not, and will not, have native data generation capabilities, commercial vendors in the endpoint detection and response (EDR) market can fill this gap with their own sensing capability. In addition EDR solutions can provide additional value by providing independent verification of native reporting and cross-platform sensing, correlation, and analysis capabilities.

In order to maintain visibility and increase interoperability, sensor and platform development should favor establishing open APIs to promote the ability to quickly integrate a new capability into a hunt team's monitoring. No single product has the capability to detect and monitor all adversary behavior, and, as adversaries find new behaviors and TTPs, defenders must be able to incorporate new methods, analysis capabilities, and sensors to respond to these changes. In addition to open APIs, vendors should work towards a common data model that supports tracing causality and enhances analyst reasoning between disparate sensors, data types, and analytic platforms. As defensive organizations adopt more threat- and TTP-based hunting strategies, they will need diverse capabilities that span products, platforms, and environments. Interoperability of these diverse capabilities will likely open APIs using standardized cross-platform data models to provide more comprehensive detection and response than single vendor or proprietary solutions can offer.

Leveraging a common data model and the MITRE ATT&CK framework allows for analysts to collaborate on TTP-focused analytics and share these, and other cyber threat information, across the larger defensive cyberspace operations community. Industry partners should

participate in the development and refinement of a common data model and shared analytical repository. The MITRE Cyber Analytics Repository may provide a starting point for these efforts, with the hope that a larger government, research, and industry partnership is created to facilitate these efforts.

Acknowledgements

This document would not have been possible without the support and contributions of many people within and outside of MITRE including, but not limited to the following: Frank Duff, Emily Hopkins, John Horn, Peter Kaloroumakis, Armand Kinslow, Willie Kupersanin, Frank Lewis, Steve Lindquist, Joe Mansour, Julie Steinke, Blake Strom and the ATT&CK Team, Maura Tennor, Cody Thomas, Dave Wilburn, Jamie Williams, and Todd Wittbold. The concepts of this paper were also influenced by thought leaders in the community including David Bianco, Roberto Rodriguez, and Paul Ewing.

4 References/Bibliography

Alsmadi, I., Karabatis, G., & Aleroud, A. (*Editors*). (2017). *Information Fusion for Cyber-Security Analytics*. Studies in Computational Intelligence 691. Springer International Publishing, Switzerland.

Atkinson, J. & Winchester, R. (2017). A process is no one: Hunting for token manipulation. Retrieved from https://specterops.io/assets/resources/A_Process_is_No_One.pdf

Berlin, K., Saxe, J., & Slater, D. (2015). Malicious behavior detection using Windows audit logs. Retrieved from https://arxiv.org/pdf/1506.04200.pdf

Cole, E. (2016). Threat Hunting: Open Season on the Adversary: A SANS Survey. Retrieved from https://www.sans.org/reading-room/whitepapers/analyst/threat-hunting-open-season-adversary-36882

Director of National Intelligence. (2017). Cyber Threat Framework. Retrieved from https://www.dni.gov/index.php/cyber-threat-framework

FireEye (2014). The Pyramid of Pain: Intel-Driven Detection & Response to Increase Your Adversary's Cost of Operations. Retrieved from: http://rvasec.com/slides/2014/Bianco_Pyramid%20of%20Pain.pdf

Gartner (2017). How to Hunt for Security Threats

Lee, M. & Rascagneres, P. (2018). Who wasn't responsible for Olympic Destroyer? (blog). Retrieved from http://blog.talosintelligence.com/2018/02/who-wasnt-responsible-for-olympic.html

Lee, R, & Lee, R.M. (2017). The Hunter Strikes Back: The SANS 2017 Threat Hunting Survey. Retrieved from https://www.sans.org/reading-room/whitepapers/analyst/hunter-strikes-back-2017-threat-hunting-survey-37760

MITRE (2015a). Adversarial Tactics, Techniques & Common Knowledge (ATT&CK). Retrieved from https://attack.mitre.org

MITRE (2015b). Cyber Analytics Repository (CAR). Retrieved from https://car.mitre.org

MITRE (2018). Crown Jewels Analysis. Retrieved from: https://www.mitre.org/publications/systems-engineering-guide/enterprise-engineering/systems-engineering-for-mission-assurance/crown-jewels-analysis

Scarfone, K. (2016). The Hunter's Handbook – Endgame's guide to adversary hunting. Retrieved from https://www.endgame.com/resource/white-paper/hunters-handbook-endgames-guide-adversary-hunting

Soria-Machado, M., Abolins, D., Boldea, C., & Socha, K. (2017). CERT-EU Security Whitepaper 17-002: Detecting Lateral Movements in Windows Infrastructure. Retrieved from: http://cert.europa.eu/static/WhitePapers/CERT-EU_SWP_17-002_Lateral_Movements.pdf

Sqrrl (2016). A Framework for Cyber Threat Hunting. Retrieved from: http://sqrrl.com/media/Framework-for-Threat-Hunting-Whitepaper.pdf

Sqrrl (2018). Huntpedia: Your Threat Hunting Knowledge Compendium. Retrieved from: http://info.sqrrl.com/huntpedia

Strom, Blake, et al. (2017). Finding Cyber Threats with ATT&CK-based Analytics. https://www.mitre.org/sites/default/files/publications/16-3713-finding-cyber-threats%20with%20att%26ck-based-analytics.pdf

Strom, Blake, et al. (2018). MITRE ATT&CK: Design and Philosophy. https://www.mitre.org/sites/default/files/publications/pr-18-0944-11-mitre-attack-design-and-philosophy.pdf

Tanenbaum, A., & Bos, H. (2017). *Modern Operating Systems*. Fourth Edition. Pearson Prentice-Hall, Pearson Eduction, Inc., Upper Saddle River, New Jersey.