GIAC GOLD
Practical Attack Detection using Big Data, Semantic Methods, and Kill Chains

GIAC (GSEC) Gold Certification

Author: Brian Nafziger, brian @ nafziger . net
Advisor: Hamed Khiabani, Ph.D.

Accepted: October 1, 2014

Abstract

Traditional toolsets using atomic syntactic-based detection methods have slowly lost the ability, in and of themselves, to detect and respond to today's well-planned, multi-phased, multi-asset, and multi-day attacks thereby leaving a gap in detecting these attacks.

Modern toolsets must rapidly detect and respond to attacks that are across multiple phases, across thousands of assets, across days of time, and across millions of events. Big Data, Semantic-based detection methods, and the Kill Chain model are, potentially, the tool, methods, and model of choice for detecting modern attacks.

The question is, do they live up to the potential? If true, then the security community must learn, grow, and share these frameworks. The objective is to show that Big Data, Semantic-based methods, and the Kill Chain model begin to fill the gap and then learn, grow, and share a working framework.

This paper will evaluate a framework that strives to aid in the detection of such attacks. A positive evaluation paves the way for the security professional to move one-step closer to early attack detection.

1. Introduction

"A vulnerability is an error or weakness in the design, implementation, or operation of a system. An attack is a means [a sequence of actions] of exploiting some vulnerability in a system. A threat is an adversary that is motivated and capable of exploiting a vulnerability." (Hollingworth, 2003; Schneider, 1999) Attack detection is the ability to detect unauthorized access to, disclosure of, modification of, destruction of, or withholding of information (Benson, n.d.).

Traditional toolsets using syntactic-based (or rule-based) attack detection alone no longer detect modern crafted attacks (Cole, 2013). Syntactic-based or rule-based attack detection methods using atomic indicators, such as hashes and IPs, are trivial to collect, however, by that very attribute they are also trivial for attackers to change and thereby effectively evade detection. (Bianco, 2013) Modern attacks are rapidly evolving, multi-phased, multi-asset, multi-day, and are resulting in catastrophic losses (Verizon, 2013).

Modern toolsets must span large data sets, must utilize semantic-based (or pattern-based) detection, and must utilize modern models such as the Kill Chain model (Cole, 2013; Bianco, 2013). Semantic-based or pattern based attack detection triggers on attacker's tools, tactics, techniques, and procedures, which are difficult to discern but difficult for attackers to change. The combined synergism of Big Data, Semantic Methods, and the Kill Chain model offers the potential to aid in the best attack detection. The potential impacts are many. To the defender, detection now identifies compromise faster resulting in less damage. To an attacker, evading detection becomes more difficult (Bianco, 2013; Bijou, 2013; Cloppert, 2010).

The objective of this paper is to show the practical steps in the creation and execution of the framework – understanding events, mapping events to Semantic Methods such as behavior analysis and mapping Semantic Methods to the Kill Chain Model - all within a Big Data environment (Cole, 2013).

1.1. Big Data

Big Data based attack detection offers hope in detecting crafted attacks across multiple domains, multiple assets and multiple days as seen in today's modern environment (Giura, 2013; Lin, 2013; Yen, 2013; Singh, 2014). Big Data is the ability to process large complex sets of data where traditional data processing fails. "Big data is high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization" (Laney, n.d.). Big Data architecture originated with Google's Map Reduce algorithm which uses parallel processing to split and distribute queries (the map step) and then gather the results (the reduce step) (Dean, J., Ghemawat, S., 2004). Enterprises are producing an unprecedented volume, velocity and variety of real-time machine data from every identity and asset that can only be aggregated, correlated, and analyzed broadly and quickly using big data technologies.

1.2. Semantics

Semantic-based attack detection offers hope in detecting crafted attacks (Cole, 2013). Semantic-based or meaning-based attack detection methods look for attack patterns using mathematics, statistics, artificial intelligence, and disciplines outside of the realms of security (Talabis, 2007). These methods can include variants such as behavior-based methods, stateful-based methods, statistics-based methods, machine learning-based methods, data mining-based methods, or baseline-based methods (Cole, 2013; Giura, 2013; Lin, 2013; Singh, 2014; Yen, 2013). The best methods produce indicators triggering on tools, tactics, techniques, and procedures used by attackers. While these semantic methods are difficult to craft, they are also difficult for attackers to change (Bianco, 2013). In addition, combining semantic methods using a framework of methods known as ensembles allows the practitioner to allow for multiple detection methods with varying level of detection ability that when combined still detect intrusions (Opitz, 1999; Xin, 2013).

1.3. Kill Chains

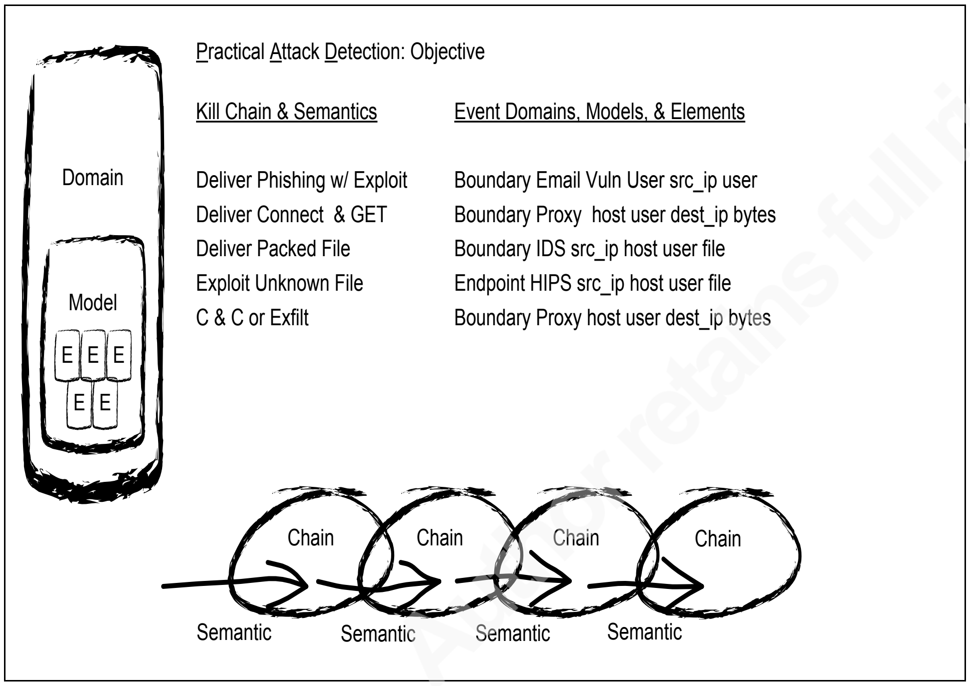
Kill Chain based attack detection offers hope in detecting crafted attacks across multiple phases (Bianco, 2013; Bijou, 2013; Hutchins, 2010; Verizon, 2013). The Kill Chain originated as a military targeting concept defined by linked steps described as a chain for, among several tasks, detecting and engaging the enemy (Tirpak, 2000).

Jeffry Carr originated the phase Cyber Kill Chain (Carr, 2008) and Lockheed Martin later defined the model (Hutchins, 2010). The Intrusion Kill Chain Model reflects the path an attacker uses in an intrusion and consists of reconnaissance, weaponization, delivery, exploitation, installation, command and control, and actions on objectives. In detail, the Intrusion Kill Chain components are reconnaissance - selecting the target; weaponization - creating the exploit; delivery - delivering the exploit; exploitation - detonating the exploit, installation – installing remote access; command and control - establishing persistent access; and actions on objectives – performing the final unauthorized actions. The actions on objectives can include access to, disclosure of, modification of, destruction of, or withholding of information (Benson, n.d.). The model indicators describe discrete parts of an intrusion. Indicators are atomic, such as an IP address; calculated, such as a file hash; or behavioral, such as a collection of atomic and calculated indicators. The Lockheed Martin model focuses on moving detection and mitigation to earlier phases (Hutchins, 2010).

Mandiant describes the attack lifecycle as reconnaissance, initial compromise, establish a foothold, escalate privileges, internal reconnaissance, move laterally, maintain a presence, and complete the mission (Mandiant, 2010; Mandiant 2013). The Mandiant model focuses on post compromise actions (Hutchins, 2010).

2. Building the Framework

The objective of this paper is to build a framework of attack detection using all three components in combination, see Figure 1: Objective. The framework defines events, uses semantic methods such as behavior analysis to detect potential attacks, uses kill chains to detect potential chain traversal, and executes all of the above using Big Data tools. This paper strives to engage in a discussion on the viability of Practical Attack Detection Using Big Data, Semantic Methods, and Kill Chains.


Practical Attack Detection: Objective

Kill Chain & Semantics	Event Domains, Models, & Elements
Deliver Phishing w/ Exploit	Boundary Email Vuln User src_ip user
Deliver Connect & GET	Boundary Proxy host user dest_ip bytes
Deliver Packed File	Boundary IDS src_ip host user file
Exploit Unknown File	Endpoint HIPS src_ip host user file
C & C or Exfilt	Boundary Proxy host user dest_ip bytes

Figure 1: Objective

Conceptually, the framework roughly uses the Data Mining concept of Knowledge Discovery in Data, which consists of data cleaning, integration, selection, transformation, mining, pattern evaluation, and knowledge presentation (Brownlee, 2014). Data cleaning and integration of events from data sources are implicit in Big Data functionality. Data selection and transformation are the processes of mapping event elements using semantic methods. Data mining and pattern evaluation is the process of detecting attacks and the kill chain progression (Hancock, n.d.).

Practically, the framework fundamentally uses the Data Mining concept of feature selection, extraction, and detection (Hancock, n.d.). The framework uses the process of selecting, extracting and detecting potentially relevant features or potential indicators from event elements using an ensemble of semantic-based methods. It populates this set of indicators across kill chain phases and alerts on a potential attack as a progression of indicators across the kill chain.

2.1. Events

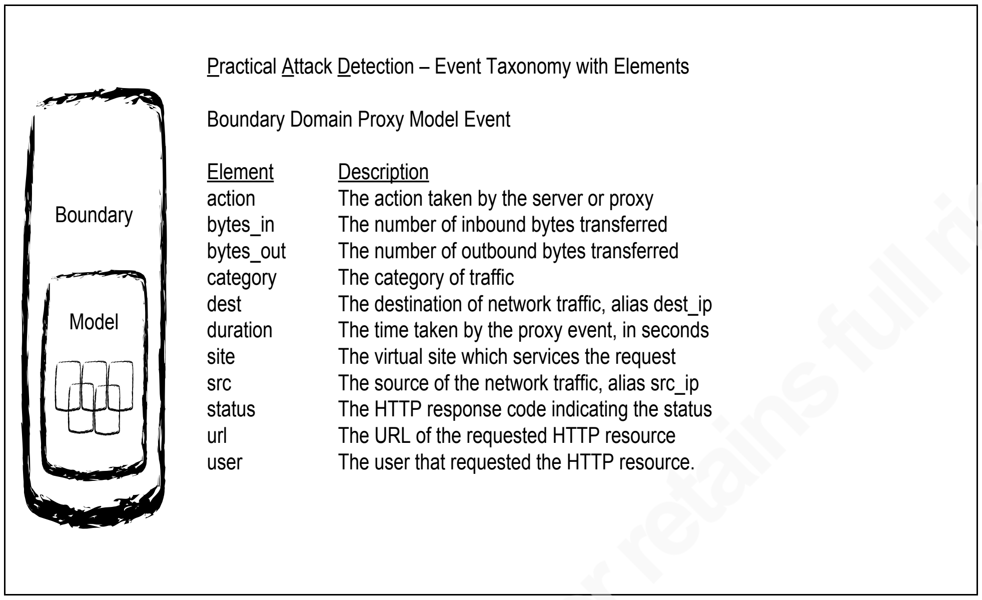
Understanding and organizing events are a fundamental requirement for selection, extraction, and detection of indicators across event elements – Big Data offers the ability to understand and organize these events and elements. Events are any detectable occurrence that has significance across the network, system, or application infrastructure (UCISA, n.d.). Events typically seen in an infrastructure include Access Services (VPN, Wired, and Wireless), Authentication Services (AD, LDAP), Allocation Services (DNS, DHCP), Filtering Services (IDS, IPS, Firewall, Proxy, and Spam), Infrastructure Services (Routers, Flows and Syslog), and Host Services (Windows, UNIX, AV, FW, IDS, and IPS).

Conceptually, the event landscape organizes into domains, models, and elements (Duncan, 2014; Sweden, 2009). Domains organize collections of models that exist in similar operational spaces. Models organize collections of elements that originate from similar operational controls. Elements represent an atomic or calculated unit originating from a singular operational control meant to convey a precise unit of meaning.

Practically, the typical operational security domains include threat, vulnerability, boundary, endpoint, identity, and incident management domains (Robb, 2011). Typical models within the boundary domain include, for example, network firewall, network intrusion detection, network web filtering, and network email filtering data models. Typical elements within the network firewall model include, for example, source IP, destination IP and bytes.

The ability to detect attack commonalities across a variety of domains, models, and elements not only requires an understanding and organization of events but also requires a lingua franca. A lingua franca is a common language with which to converse across a variety of domains, models, and elements.

Using Splunk as the Big Data environment, Splunk offers a taxonomy using models and elements with a common language for elements. The Splunk taxonomy is the Common Information Model. From the prior examples, the lingua franca for source is src_ip and for destination is dest_ip (Splunk, 2014). Begin the process of Practical Attack Detection by building a field manual and document the event taxonomy, as seen below in **Figure 2: Event Taxonomy with Elements**.



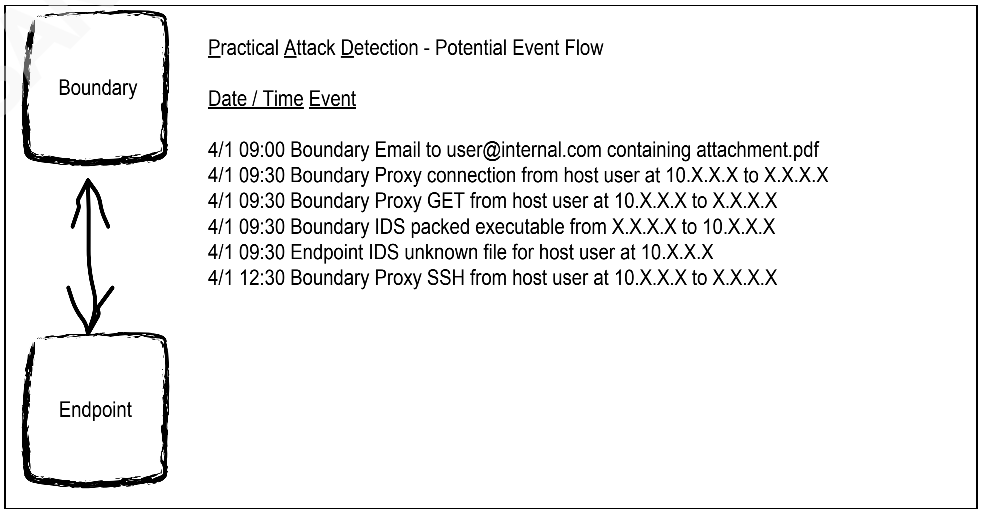
Practical Attack Detection – Event Taxonomy with Elements

Boundary Domain Proxy Model Event

Element	Description
action	The action taken by the server or proxy
bytes_in	The number of inbound bytes transferred
bytes_out	The number of outbound bytes transferred
category	The category of traffic
dest	The destination of network traffic, alias dest_ip
duration	The time taken by the proxy event, in seconds
site	The virtual site which services the request
src	The source of the network traffic, alias src_ip
status	The HTTP response code indicating the status
url	The URL of the requested HTTP resource
user	The user that requested the HTTP resource.

Figure 2: Event Taxonomy with Elements (Splunk, 2014)

Using the event taxonomy, an example of a potential event flow is below in Figure 3: Potential Event Flow. In this event flow, the user gets an email, initiates an outbound HTTP connection, downloads a file from the outbound connection, and finally initiates an outbound SSH connection. Singular events are unlikely to develop an understanding of what transpired, especially when the attack atomic indicators are new. Multiple events begin to develop an understanding of attack pattern. The synergism of all the events will develop the best understanding of multiple attack patterns.



Practical Attack Detection - Potential Event Flow

Date / Time Event

```
4/1 09:00 Boundary Email to user@internal.com containing attachment.pdf
4/1 09:30 Boundary Proxy connection from host user at 10.X.X.X to X.X.X.X
4/1 09:30 Boundary Proxy GET from host user at 10.X.X.X to X.X.X.X
4/1 09:30 Boundary IDS packed executable from X.X.X.X to 10.X.X.X
4/1 09:30 Endpoint IDS unknown file for host user at 10.X.X.X
4/1 12:30 Boundary Proxy SSH from host user at 10.X.X.X to X.X.X.X
```

Figure 3: Potential Event Flow

Brian Nafziger, brian @ nafziger . net

Splunk offers the ability to query the event elements using the Search Processing Language. The language offers a variety of functionality, including but not limited to, searching and manipulating time series events by piping a myriad array of commands and functions to produce alerts, reports, and dashboards (Alspaugh, S., Ganapathi, A., Hearst, M., & Katz, R., 2013). The language also provides many complex features to abstract and manage data. A simple example of a potential email query is below in

Figure 4: Potential Email Query. Further Splunk queries throughout are in simple terms but derived from actual working queries.

```
index=email
source=external recipient=* action=deliver
(attachment="*.doc*" OR attachment="*.xls*" OR attachment="*.ppt*")
```

Figure 4: Potential Email Query

Understanding and organizing event context is also a requirement. Context is an ally in comprehensive attack detection for detecting attacks crossing boundaries into high-risk areas, for rapid risk assessments of events, and for tying together disparate events. The building blocks of context are assets, identities and vulnerabilities (Chuvakin, 2010). For example, events are of greater value with an understanding of details not commonly in events, such as, identification of critical users, identification of critical assets, or identification of assets with critical vulnerabilities.

Splunk offers the ability to build and query context-using databases, using directories, using events or using customized commands (Splunk, 2014). Splunk can then cache and retrieve context on the local file system using lookups (Splunk, 2014). An example of a potential email query with identity context is below in Figure 5: Potential Email Query with Identity Context.

```
$SPLUNK_HOME\etc\apps\search\lookup\INFOSEC-CTX-IDENTITY-DB.csv
email,user
JohnCalvin@gmail.com,jcalvin
ThomasHobbes@gmail.com,thobbes

$SPLUNK_HOME\etc\apps\search\lookups\transforms.conf
```

```
[INFOSEC-CTX-IDENTITY-DB]
filename = INFOSEC-CTX-IDENTITY-DB.csv

index=email
source=external recipient=* action=deliver
(attachment="*.doc*" OR attachment="*.xls*" OR attachment="*.ppt*")
| lookup INFOSEC-CTX-IDENTITY-DB email AS recipient OUTPUT user
```

Figure 5: Potential Email Query with Identity Context

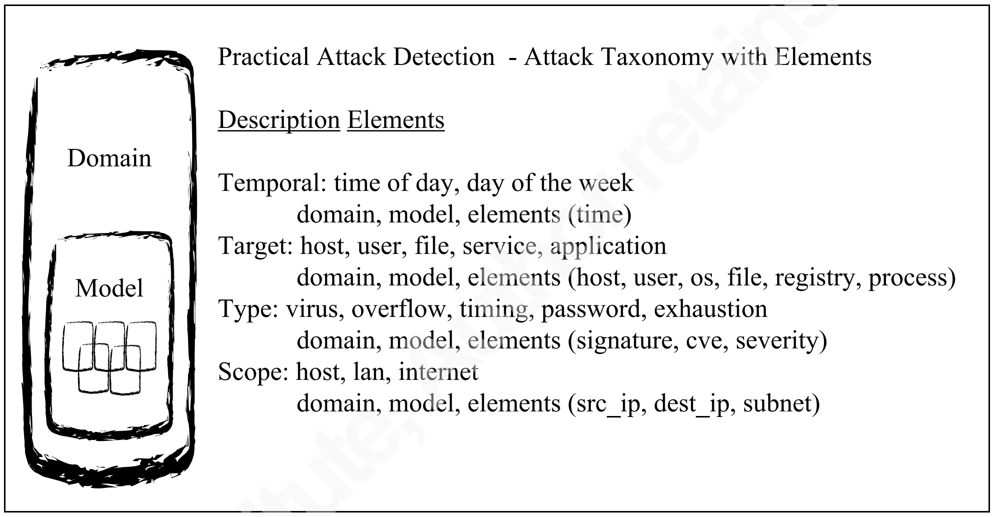
This paper strives to focus solely on *combining* big data, semantics, and kill chains into a framework. The configuration, collection, detection, and analysis of data are a robust topic. The topic also includes further potential data such as content, session, transaction and statistical data. The addition of a full range of data will add valuable insight into attacks. Such data adapts well in the big data, semantic methods, and kill chain architecture. There are several excellent books covering these and many other detailed tasks such as best practices, collecting, monitoring, and low-level analysis (Bejtlich, 2013; Bejtlich, 2006; Bejtlich, 2005; Sanders, 2013).

2.2. Semantics

Understanding and organizing semantic detection methods within events is the next requirement for selection, extraction, and detection of indicators across events. Semantic-based methods detect attacks and can include variants such as behavior-based methods, stateful-based methods, statistics-based methods, machine learning-based methods, data mining-based methods, or baseline-based levels (Cole, 2013; Giura, 2013; Lin, 2013; Singh, 2014; Yen, 2013). An ensemble of methods finds potential attacks, not necessarily confirmed attacks (Opitz, 1999; Xin, 2013). The accumulation of multiple potential attacks across the kill chain model increases the potential of an actual attack.

Conceptually, the components of event organization, domains, models and elements, still apply in discerning attacks and creating semantic detection methods. Therefore, these attacks and semantic detection methods exist across the event taxonomy. However, it is necessary to classify them to ensure appropriate indicator selection, extraction, and detection. Practically, suggested attack indicators include, but are not limited to temporality - time of day, day of the week; target - host, service, application, file, user; type - virus, overflow, timing, password, exhaustion; and scope - host, LAN, internet (Hollingworth, 2003).

Using Splunk as the Big Data environment, map event elements to the aforementioned attack indicators and create an Attack Taxonomy. In most cases, attack elements are implicitly included in the Common Information Model (for instance, temporality or time). Document the attack taxonomy as below in Figure 6: Attack Taxonomy with Elements.



Practical Attack Detection - Attack Taxonomy with Elements

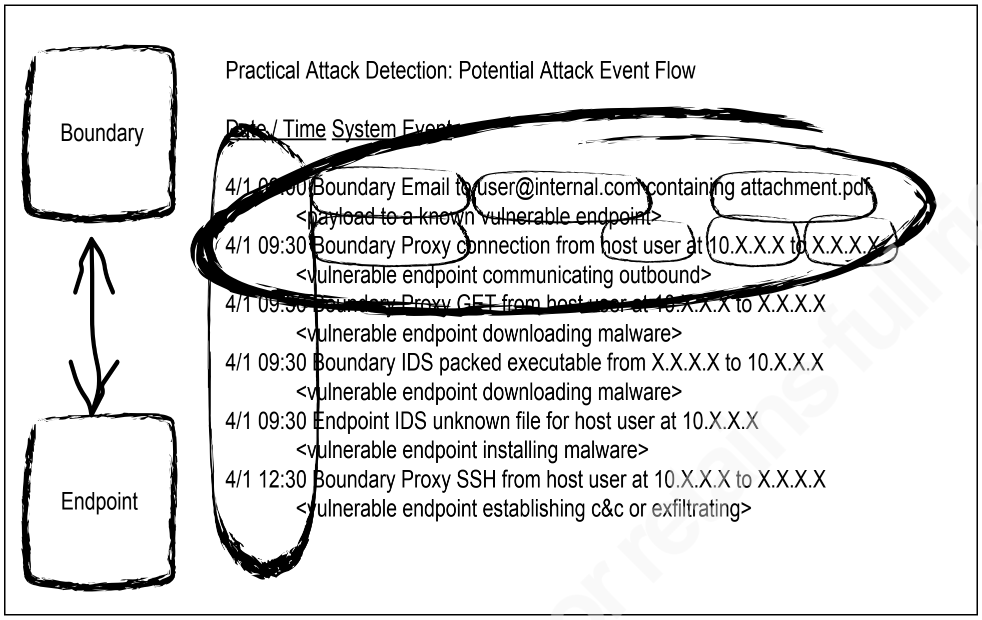
Description Elements

Temporal: time of day, day of the week
    domain, model, elements (time)
Target: host, user, file, service, application
    domain, model, elements (host, user, os, file, registry, process)
Type: virus, overflow, timing, password, exhaustion
    domain, model, elements (signature, cve, severity)
Scope: host, lan, internet
    domain, model, elements (src_ip, dest_ip, subnet)

Figure 6: Attack Taxonomy with Elements (Hollingworth, 2003)

Semantic Detection Method discernment requires research of known attacks or understanding of potentially unknown attacks. Books, journals, blogs, reports, and closed sources (private investigations and forensics) offer attack details for observation. The intent of this process is to define the value of an ensemble of semantic detection methods based on behaviors or other semantic methods.

Using the attack taxonomy, the continuing example of a potential attack event flow with elements is below in Figure 7: Potential Attack Event Flow. The attacker sends a Trojanized email to a vulnerable user. The vulnerable user opens the Trojanized email, and then the Trojan exploits and initiates an outbound connection. From the attack events, one semantic detection method is detecting any potentially exploitable payload sent to a vulnerable user. From the events, the semantic detection method indicators are time, recipient, attachment, host, and user.



Practical Attack Detection: Potential Attack Event Flow

```
Date / Time System Events

4/1 09:30 Boundary Email to user@internal.com containing attachment.pdf
          <payload to a known vulnerable endpoint>
4/1 09:30 Boundary Proxy connection from host user at 10.X.X.X to X.X.X.X
          <vulnerable endpoint communicating outbound>
4/1 09:30 Boundary Proxy GET from host user at 10.X.X.X to X.X.X.X
          <vulnerable endpoint downloading malware>
4/1 09:30 Boundary IDS packed executable from X.X.X.X to 10.X.X.X
          <vulnerable endpoint downloading malware>
4/1 09:30 Endpoint IDS unknown file for host user at 10.X.X.X
          <vulnerable endpoint installing malware>
4/1 12:30 Boundary Proxy SSH from host user at 10.X.X.X to X.X.X.X
          <vulnerable endpoint establishing c&c or exfiltrating>
```

Figure 7: Potential Attack Event Flow

Splunk offers the ability to query attack elements using the Search Processing Language. Using contexts, a complex semantic detection method query could identify incoming emails with attachments sent to vulnerable user hosts. An example of an email to vulnerable host query is below in Figure 8: Potential Email Semantic (in practice as seen later, it is not quite this simple).

```
index=email
source=external recipient=* action=deliver
(attachment="*.doc*" OR attachment="*.xls*" OR attachment="*.ppt*")
| lookup INFOSEC-CTX-IDENTITY-DB email AS recipient OUTPUT user
| lookup INFOSEC-CTX-ASSET-DB user OUTPUT host
| lookup INFOSEC-CTX-VULNERABILITY-DB host OUTPUT signature
| where like (signature,"%")
| table _time user host signature
```

Figure 8: Potential Email Semantic Query

Document the available semantic methods as seen below in Figure 9: Semantic Methods.

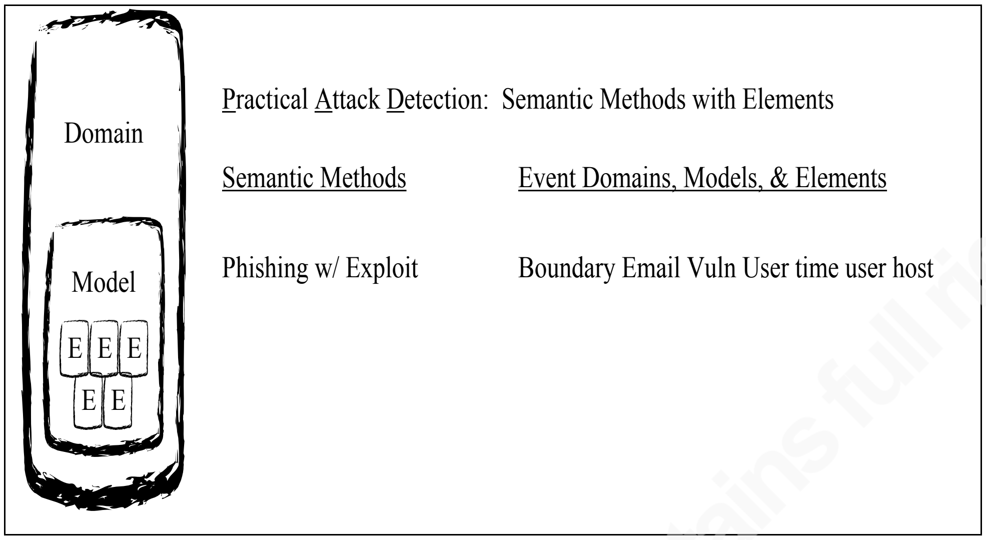
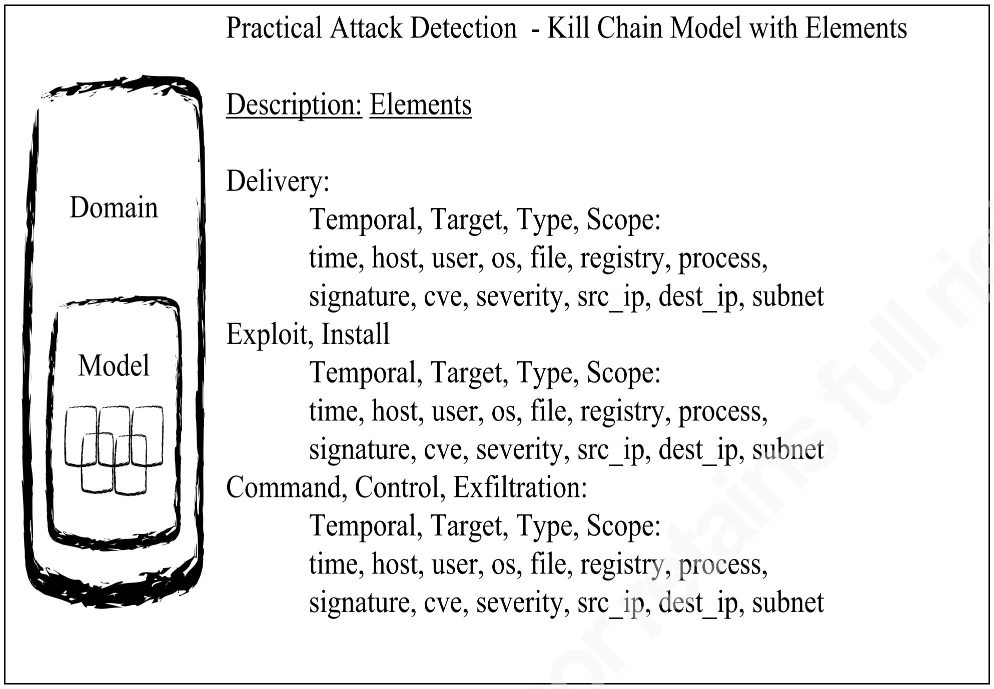

Domain
Model
E E E
E E
Practical Attack Detection: Semantic Methods with Elements
Semantic Methods
Event Domains, Models, & Elements
Phishing w/ Exploit
Boundary Email Vuln User time user host

Figure 9: Semantic Methods with Elements

Once again, this paper strives to focus solely on *combining* big data, semantics, and kill chains into a framework. The analysis of semantic methods as detection mechanisms is also a robust topic.

2.3. Kill Chains

Understanding and organizing kill chains using semantic methods and events is the final requirement for selection, extraction, and detection of indicators across events. Kill chains detect a specific sequence of attack patterns or phases of an attack. Lockheed Martin and Mandiant offer defined models and distilling the aspects of both models offers a simplified kill chain model for use in this paper - delivery, exploit, install, command and control and exfiltration where each layer of the model consists of attack indicators or elements (Hutchins, 2010; Mandiant, 2010; Mandiant 2013). Document the kill chain model as seen below in **Figure 10: Kill Chain Model with Elements**.



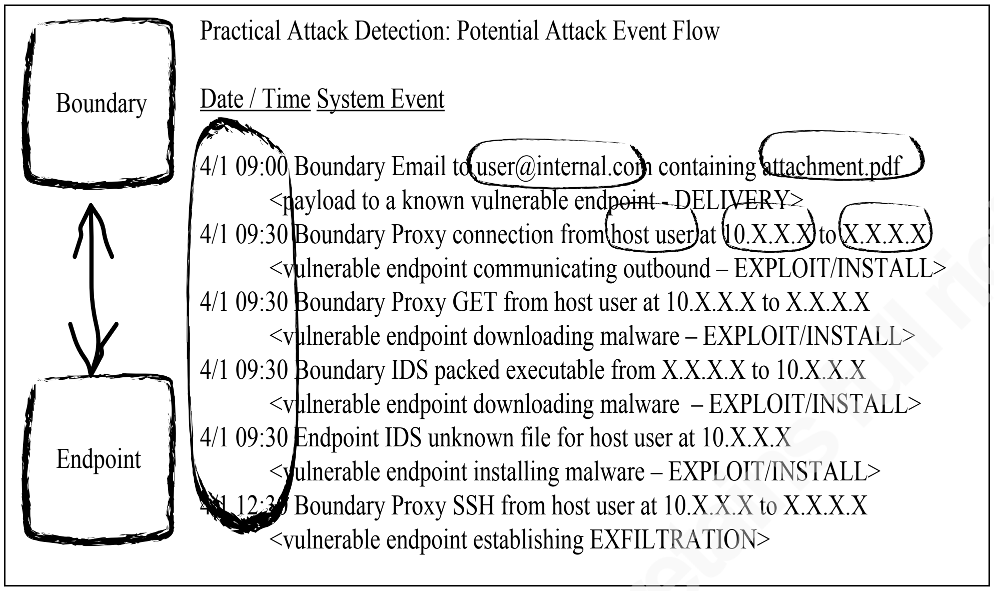
Practical Attack Detection - Kill Chain Model with Elements

Description: Elements

Delivery:
    Temporal, Target, Type, Scope:
    time, host, user, os, file, registry, process,
    signature, cve, severity, src_ip, dest_ip, subnet
Exploit, Install
    Temporal, Target, Type, Scope:
    time, host, user, os, file, registry, process,
    signature, cve, severity, src_ip, dest_ip, subnet
Command, Control, Exfiltration:
    Temporal, Target, Type, Scope:
    time, host, user, os, file, registry, process,
    signature, cve, severity, src_ip, dest_ip, subnet

Figure 10: Kill Chain Model with Elements (Hollingworth, 2003; Hutchins, 2010; Mandiant)

Kill Chain discernment only requires determining the location of each semantic detection method within the attack kill chain. Using the kill chain model, the continuing example of a potential phased attack event flow with elements is below in Figure 11: Potential Phased Attack Event Flow. The attacker sends a Trojanized email to a vulnerable user – delivery. The vulnerable user opens the Trojanized email, and the Trojan exploits and installs malware – exploitation and installation. Finally, the user initiates an outbound connection – command and control or exfiltration.



Practical Attack Detection: Potential Attack Event Flow

```
Date / Time System Event

4/1 09:00 Boundary Email to user@internal.com containing attachment.pdf
          <payload to a known vulnerable endpoint - DELIVERY>
4/1 09:30 Boundary Proxy connection from host user at 10.X.X.X to X.X.X.X
          <vulnerable endpoint communicating outbound – EXPLOIT/INSTALL>
4/1 09:30 Boundary Proxy GET from host user at 10.X.X.X to X.X.X.X
          <vulnerable endpoint downloading malware – EXPLOIT/INSTALL>
4/1 09:30 Boundary IDS packed executable from X.X.X.X to 10.X.X.X
          <vulnerable endpoint downloading malware – EXPLOIT/INSTALL>
4/1 09:30 Endpoint IDS unknown file for host user at 10.X.X.X
          <vulnerable endpoint installing malware – EXPLOIT/INSTALL>
4/1 12:30 Boundary Proxy SSH from host user at 10.X.X.X to X.X.X.X
          <vulnerable endpoint establishing EXFILTRATION>
```

Figure 11: Potential Phased Attack Event Flow

Splunk also offers the ability to store context in a rolling state table both updating and expiring content over time. The Kill Chain is essentially a context with pertinent indicators from the semantic methods such as time, host, user, src, and dest. After reaching a defined threshold of common indicators across the kill chain, an alert and report auto-generate with data from the kill chain, assets, identities, and vulnerabilities (Palantir Cyber, 2013). Other potential reporting details include associated AV, IDS, DNS, and proxy events (Palantir Cyber, 2013). An example of an email to vulnerable user query that generates kill chain indicators, loads previous indicators, and saves still valid indicators into a context is seen below in Figure 12: Potential Email Semantic Kill Chain Query (again, in practice, as seen later, it is not quite this simple).

```
earliest=-20m@m latest=-5m@m
index=email  source=external recipient=* action=deliver
(attachment="*.doc*" OR attachment="*.xls*" OR attachment="*.ppt*")
| lookup INFOSEC-CTX-IDENTITY-DB email AS recipient OUTPUT user
| lookup INFOSEC-CTX-ASSET-DB user OUTPUT host
| lookup INFOSEC-CTX-VULNERABILITY-DB host OUTPUT signature
| where like (signature,"%")
| table _time user host signature

| eval chain = "Delivery" | eval semantic = "Mail Recipient Vulnerable"
| stats min(_time) as firstTime  max(_time) as lastTime  by user host semantic
chain
| table firstTime lastTime user host signature semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime by user host
semantic chain
| table firstTime lastTime user host signature semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 12: Potential Email Semantic Kill Chain Query

Document the kill chain with semantic methods and elements, as seen below in Figure 13: Kill Chain with Semantic Methods and Elements.

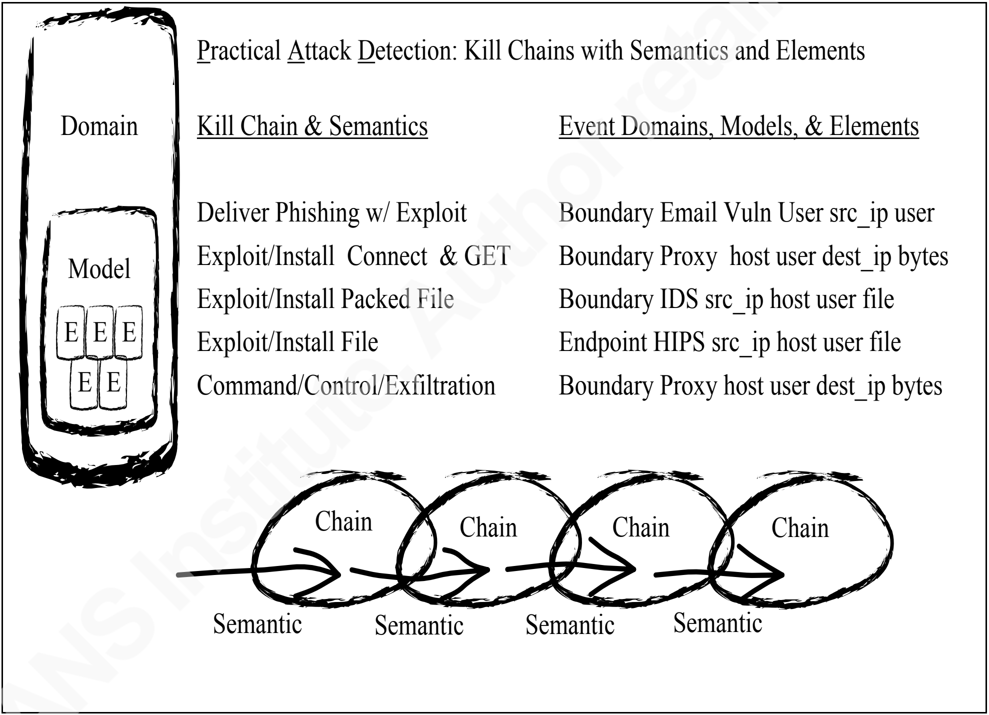

Practical Attack Detection: Kill Chains with Semantics and Elements
Domain
Model
E E E
E E
Kill Chain & Semantics
Event Domains, Models, & Elements
Deliver Phishing w/ Exploit
Boundary Email Vuln User src_ip user
Exploit/Install Connect & GET
Boundary Proxy host user dest_ip bytes
Exploit/Install Packed File
Boundary IDS src_ip host user file
Exploit/Install File
Endpoint HIPS src_ip host user file
Command/Control/Exfiltration
Boundary Proxy host user dest_ip bytes
Chain
Chain
Chain
Chain
Semantic
Semantic
Semantic
Semantic

Figure 13: Kill Chain with Semantic Methods and Elements

2.4. Building Contexts

Making it practical, context is necessary and adds value in identifying events, building semantics and in reporting kill chain events. Create context tables using asset, identity and vulnerability data from a variety of sources. Static contexts created from an external database or dynamic context created from events that have details unavailable in databases. Use the following query templates as the basis for building context. These queries do not account for extraction nuances in databases and events or the higher levels of query abstraction available in Splunk

Create and schedule a current asset context using a remote database query as seen below in Figure 14: Context - Asset Database.

```
| dbquery "server" "SELECT * FROM VENDOR-ASSET-DB"
| table status host user location type manu model
| outputlookup INFOSEC-CTX-ASSET-DB.csv
```

Figure 14: Context - Asset Database

Create and schedule a current identity context using a remote active directory query as seen below in Figure 15: Context - Identity Database.

```
| ldapsearch domain=AD
search="(&(objectclass=user)(!(objectClass=computer)))"
| table status user name department address city state country telephone email
| outputlookup INFOSEC-CTX-IDENTITY-DB.csv
```

Figure 15: Context - Identity Database

Create and schedule a current vulnerability context using a local vulnerability database query as seen below in Figure 16: Context - Vulnerability Database:

```
| inputlookup VENDOR-VULNERABILITY-DB.csv
| table status ip dns host os type severity signature cve cvss
| outputlookup INFOSEC-CTX-VULNERABILITY-DB.csv
```

Figure 16: Context - Vulnerability Database

Create and schedule a rolling dynamic asset context using an event query as seen below in Figure 17: Context - Asset Dynamic. In this example, the context derives from host and DHCP events in real-time.

```
| metasearch earliest=-20m@m latest=-5m@m
| stats min(_time) as firstTime, max(_time) as lastTime by host
| lookup dnslookup clienthost AS host OUTPUT clientip AS ip | mvexpand ip
| append [
search earliest=-20m@m latest=-5m@m
source="*dhcpd.log" GrantLease OR RenewLease
| stats min(_time) as firstTime, max(_time) as lastTime by host ip mac ]
| table firstTime  lastTime host ip mac

| inputlookup append=T INFOSEC-CTX-ASSET-DYNAMIC.csv
| where lastTime > relative_time(now(), "-24h")
| stats min(firstTime) as firstTime max(lastTime) as lastTime by  host ip mac
| table firstTime  lastTime host ip mac
| outputlookup INFOSEC-CTX-ASSET-DYNAMIC.csv
```

Figure 17: Context - Asset Dynamic

Create and schedule a rolling identity context using an event query as seen below in Figure 18: Context - Identity . In this example, the context derives from Windows Logon events in real-time.

```
earliest=-20m@m latest=-5m@m
sourcetype=WinEventLog:Security  EventCode=4624 user!=*$
| stats  min(_time) as firstTime,  max(_time) as lastTime by host user src_host
src_ip
| table firstTime  lastTime host user src_host src_ip

| inputlookup append=T INFOSEC-CTX-IDENTITY-DYNAMIC.csv
| where lastTime > relative_time(now(), "-24h")
| stats min(firstTime) as firstTime max(lastTime) as lastTime by  host user
src_host src_ip
| table firstTime  lastTime host user src_host src_ip
| outputlookup INFOSEC-CTX-IDENTITY-DYNAMIC.csv
```

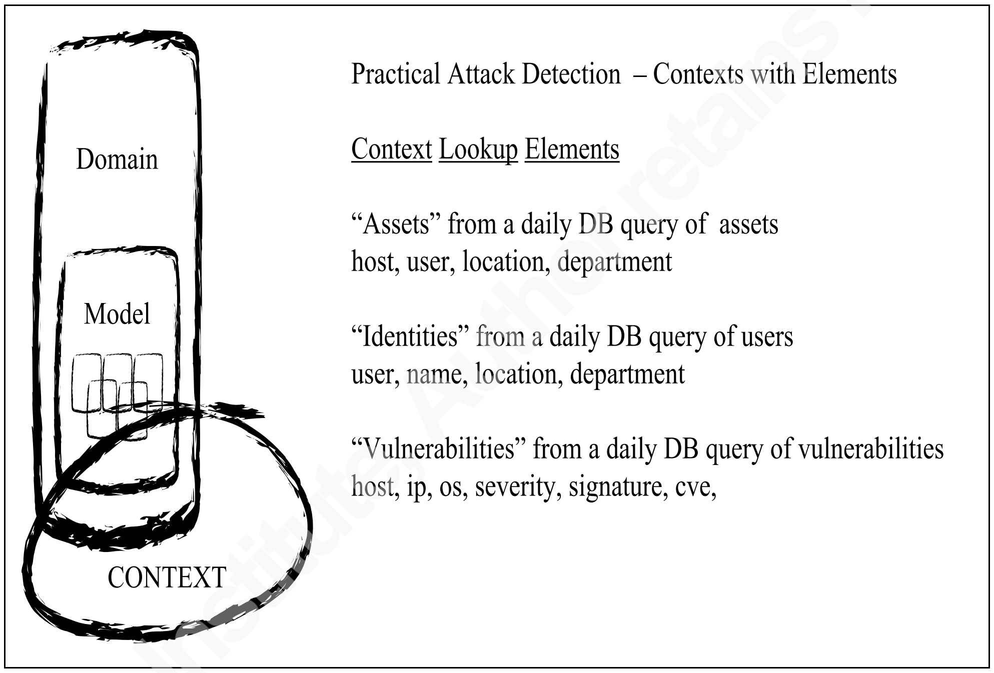
Figure 18: Context - Identity Dynamic

Create and schedule a rolling vulnerability context using an event query as seen below in Figure 19: Context - Vulnerability D. In this example, the context derives from vulnerability events in real-time.

```
earliest=-20m@m latest=-5m@m
index=vulnerabilities | fillnull value=NULL
| stats min(_time) as firstTime max(_time) as lastTime last(status) as status by id
ip dns host os type severity signature cve cvss
| table status id ip dns host os type severity signature cve cvss firstTime lastTime

| inputlookup append=T INFOSEC-CTX-VULNERABILITY.csv
| where lastTime > relative_time(now(), "-30d")
| stats min(firstTime) as firstTime max(lastTime) as lastTime  last(status) as
status by id ip dns host os type severity signature cve cvss
| table status id ip dns host os type severity signature cve cvss firstTime lastTime
| outputlookup INFOSEC-CTX-VULNERABILITY.csv
```

Figure 19: Context - Vulnerability Dynamic

Document the event contexts, as seen below in Figure 20: Contexts with Elements.



Practical Attack Detection – Contexts with Elements

Context Lookup Elements

"Assets" from a daily DB query of assets
host, user, location, department

"Identities" from a daily DB query of users
user, name, location, department

"Vulnerabilities" from a daily DB query of vulnerabilities
host, ip, os, severity, signature, cve,

Figure 20: Contexts with Elements (Chuvakin, 2010)

2.5. Building Delivery Detection

Crafted attacks, as described in Shakarian's *Introduction to Cyber-Warfare* (Shakarian, 2013), often utilize malware delivery of Trojanized PDF, DOC, and PPT email attachments to exploit an existing vulnerability on a user's asset. Within the event taxonomy, emails with listed recipients and attachments should exist and vulnerabilities should exist. Using these indicators, an email based semantic detection method would be to notate any incoming email containing a PDF, DOC, PPT, or URL sent to a known vulnerable asset.

Create and schedule a semantic query as seen below in Figure 21: Delivery Mail Recipient Vulnerable. First, identify incoming emails with potentially Trojanized attachments. Then identify users with vulnerable hosts receiving those attachments. The expanded query below reflects users with multiple assets and multiple vulnerabilities. Finally select and save indicators including time, host, user, and optional os, file, registry, process, signature, cve, severity, src_ip, and dest_ip.

```
earliest=-20m@m latest=-5m@m
index=email  source=external recipient=* action=deliver  attachment=*
| lookup INFOSEC-CTX-IDENTITY-DB email AS recipient OUTPUT user
| lookup INFOSEC-CTX-ASSET-DB user OUTPUT host | mvexpand host
| lookup INFOSEC-CTX-ASSET-DB user, host OUTPUT type | mvexpand type
| lookup INFOSEC-CTX-ASSET-DB user, host, type OUTPUT status |
mvexpand status
| rename status as assetStatus | rename type as assetType
| where like(assetType, "%NOTEBOOK%") AND assetStatus="DEPLOYED"
| lookup INFOSEC-CTX-VULNERABILITY-DB host OUTPUT signature |
mvexpand signature
| lookup INFOSEC-CTX-VULNERABILITY-DB host, signature OUTPUT type |
mvexpand type
| lookup INFOSEC-CTX-VULNERABILITY-DB host, signature, type OUTPUT
status | mvexpand status
| lookup INFOSEC-CTX-VULNERABILITY-DB host, signature, type, status
OUTPUT cvss | mvexpand cvss
| rename status as vulnStatus | rename type as vulnType | rename cvss as
vulnCvss
| where like(signature,"%") AND vulnType="Confirmed" AND
vulnStatus="Active" AND vulnCvss>=7
| stats last(_time) as _time by user host
| table _time user host

| eval chain = "Delivery" | eval semantic = "Mail Recipient Vulnerable"
| stats min(_time) as firstTime max(_time) as lastTime by user host semantic
chain
| table firstTime lastTime user host semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats min(firstTime) as firstTime max(lastTime) as lastTime by user host
semantic chain
| table firstTime lastTime user host semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 21: Delivery Mail Recipient Vulnerable

Another email based semantic detection technique known as gray listing focuses on the propensity of spam for malware delivery using never seen before source IPs (Jakhar, 2008). Techniques include variations such as checking a source IP, sender, recipient table for past usage and cluster analysis of source IPs. Within the event taxonomy, email source IPs, senders and recipients should exist. Using these indicators, a simple email based semantic detection method would be to evaluate and notate new incoming emails against a context of previous source IPs, senders and recipients.

Create and schedule a semantic query with context as seen below in Figure 22: Delivery Mail Sender Unique. First, identify incoming emails. Then identify users and hosts receiving those emails. The expanded query below reflects users with multiple assets. Select and save the earliest time and latest time for the source IP, sender and recipient. Then select only those emails never previously seen using the difference from the earliest time and latest time. Finally select and save indicators including time, user, host and optional src_ip, sender, and recipient.

```
earliest=-20m@m latest=-5m@m
index=email
source=external action=deliver src_ip=* sender=* recipient=*
| eval recipient=split(recipient, " ")| mvexpand recipient
| stats min(_time) as firstTime max(_time) as lastTime by src_ip sender recipient
| table firstTime lastTime src_ip sender recipient

| lookup INFOSEC-CTX-IDENTITY-DB email AS recipient OUTPUT user
| lookup INFOSEC-CTX-IDENTITY-DB email AS recipient OUTPUT user
| lookup INFOSEC-CTX-ASSET-DB user OUTPUT host | mvexpand host
| lookup INFOSEC-CTX-ASSET-DB user, host OUTPUT type | mvexpand type
| lookup INFOSEC-CTX-ASSET-DB user, host, type OUTPUT status | mvexpand
status
| rename status as assetStatus | rename type as assetType
| where (like(assetType, "%NOTEBOOK%")) AND assetStatus="DEPLOYED"
| table firstTime lastTime user host src_ip sender recipient

| inputlookup append=T INFOSEC-CTX-MAIL-SENDER-UNIQUE.csv
| where lastTime > relative_time(now(), "-30d")
| stats min(firstTime) as firstTime max(lastTime) as lastTime by user host src_ip
sender recipient
| table firstTime lastTime user host src_ip sender recipient
| outputlookup INFOSEC-CTX-MAIL-SENDER-UNIQUE.csv

| eval diffTime = lastTime-firstTime
| where lastTime-firstTime <= 300

| eval chain = "Delivery" | eval semantic = "Mail Sender Unique"
| stats min(firstTime) as firstTime max(lastTime) as lastTime by user host
semantic chain
| table firstTime lastTime user host semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime by user host
semantic chain
| table firstTime lastTime user host semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 22: Delivery Mail Sender Unique

2.6. Building Exploitation and Installation Detection

Crafted attacks exploit, install and persist using common load points such as the Run and Run Once registry entries (Russinovich, M., & Cogswell, B., 2014). Within the event taxonomy, host endpoint protection (and Splunk for that matter) offers file integrity and registry monitoring and should include changes to these common load points. Using these indicators, a useful semantic detection method would be to evaluate and notate user, host, and load entries.

Create and schedule a semantic query with context as seen below in Figure 23: Exploit Endpoint Load Unique. First, identify common load point changes. Identify the user and host. Select and save the earliest time and latest time for the host, user, file, and or registry. Then select only those changes never previously seen using the difference from the earliest time and latest time. Finally select and save indicators including time, user, host, and optional file and registry changes.

```
earliest=-20m@m latest=-5m@m
index=endpoint "common load points"
| stats min(_time) as firstTime max(_time) as lastTime by host user file registry
| table firstTime lastTime host user file registry

| inputlookup append=T INFOSEC-CTX-ENDPOINT-CHANGE-UNIQUE.csv
| where lastTime > relative_time(now(), "-30d")
| stats min(firstTime) as firstTime max(lastTime) as lastTime by host user file
registry
| table firstTime lastTime host user file registry
| outputlookup INFOSEC-CTX-ENDPOINT-CHANGE-UNIQUE.csv

| eval diffTime = lastTime-firstTime
| where lastTime-firstTime <= 300
| eval chain = "Exploit" | eval semantic = "Endpoint Load Unique"
| stats min(firstTime) as firstTime  max(firstTime) as lastTime by host user
semantic chain
| table firstTime lastTime user host semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime  by host user
semantic chain
| table firstTime lastTime host user semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 23: Exploit Endpoint Load Unique

Crafted attacks can also leverage existing common malware exploitations (Aziz, A., 2011). Within the event taxonomy, host endpoint protection offers signature, reputation and behavioral based monitoring (Symantec, 2012; Symantec, 2013). This paper focuses on the detection of attacks using Big Data, Semantics and the Kill Chain, however, not to the exclusion of syntactic or signature detection. Using these indicators, a useful detection method would be to evaluate and notate user and host with known or potential threat entries.

Create and schedule a query with context as seen below in Figure 24: Exploit Endpoint Risk Found. First, identify any known or potential risks. Identify the user and host. Select and save the earliest time and latest time for the host, user, and file. Finally select and save indicators including time, user, host, and optional threat details.

```
earliest=-20m@m latest=-5m@m
index=endpoint "Virus found" OR "Risk found" OR "File submission"
| stats min(_time) as firstTime max(_time) as lastTime by host user signature
filename
| table firstTime lastTime host user signature filename

| eval chain = "Exploit" | eval semantic = "Endpoint Risk Found"
| stats min(firstTime) as firstTime  max(lastTime) as lastTime by host user
semantic chain
| table firstTime lastTime host user semantic chain name
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime  by host user
semantic chain
| table firstTime lastTime host user semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 24: Exploit Endpoint Risk Found

2.7. Building Command, Control and Exfiltration Detection

Crafted attacks, as described in Cole's *Advanced Persistent Threat* (Cole, 2013), need to utilize command and control and also have the potential to exfiltrate data. Cole describes the potential to identify command and control and exfiltration using:

1) Length of Connections – typical users make short outbound connections
2) Number of packets – typical connections send a smaller number of packets
3) Amount of data – typical connections send a smaller amount of data
4) Destination IP – typical users connect to known domains

(Cole, Advanced Persistent Threat, 2013, p. 38,134-135,183-187)

Breaking down "Length of Connections," identify abnormally long connections by comparing single user connection duration against the connection duration of all users. Within the event taxonomy, proxy filtering offers user and host based connection monitoring including duration.

Create and schedule a semantic query with context as seen below in Figure 25: Exfiltrate Proxy Long Connect. First, generate connection duration standard deviation across all the proxy events. Then notate any proxy events with a duration greater than three times the standard deviation. Finally select and save indicators including time, user, host, and optional details such as destination.

```
earliest=-65m@m latest=-5m@m
index=proxy
| eventstats  min(_time) as firstTime max(_time) as lastTime avg(duration) as
avg stdev(duration) as stdev
| eval notable=avg + 3*stdev
| where duration > notable
| where _time > relative_time(now(),"-20m")
| lookup dnslookup clientip AS src OUTPUT clienthost AS host
| table _time firstTime lastTime src host user category dest duration notable

| eval chain = "Exfiltrate" | eval semantic = "Proxy Long Connect"
| stats min(firstTime) as firstTime  max(lastTime) as lastTime by host user
semantic chain
| table firstTime lastTime user host semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime  by host user
semantic chain
| table firstTime lastTime host user semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 25: Exfiltrate Proxy Long Connect

Breaking down "Number of Packets," identify an abnormal number of connections by comparing single user connection count against the connection count of all users. Within the event taxonomy, proxy filtering offers user and host based connection monitoring.

Create and schedule a semantic query with context as seen below in Figure 26: Exfiltrate Proxy Frequent Connect. First, generate connection count standard deviation across all the proxy events. Then notate any proxy users with a count greater than three times the standard deviation. Finally select and save indicators including time, user, host, and optional details such as destination.

```
earliest=-65m@m latest=-5m@m
index=proxy
| bucket _time span=15m
| stats count by _time user src dest category
| eventstats min(_time) as firstTime max(_time) as lastTime avg(count) as avg
stdev(count) as stdev
| eval notable=avg + 3*stdev
| where count > notable
| where _time > relative_time(now(),"-20m")
| lookup dnslookup clientip AS src OUTPUT clienthost AS host
| table _time firstTime lastTime src host user category dest count notable

| eval chain = "Exfiltrate" | eval semantic = "Proxy Frequent Connect"
| stats min(firstTime) as firstTime  max(lastTime) as lastTime by host user
semantic chain
| table firstTime lastTime user host semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime  by host user
semantic chain
| table firstTime lastTime host user semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure 26: Exfiltrate Proxy Frequent Connect

Breaking down "Amount of Data," identify an abnormal amount of outbound data by comparing single user connection bytes outbound against the connection bytes outbound of all users. Within the event taxonomy, proxy filtering offers user and host based connection monitoring including bytes outbound.

Create and schedule a semantic query with context as seen below in Figure 27: Exfiltrate Proxy Large Outbound. First, generate connection count standard deviation across all the proxy events. Then notate any proxy users with bytes outbound greater than three times the standard deviation. Finally select and save indicators including time, user, host, and optional details such as destination.

```
earliest=-65m@m latest=-5m@m
index=proxy
| bucket _time span=15m
| stats sum(bytes_out) as bytes_out by _time user src dest category
| eventstats min(_time) as firstTime max(_time) as lastTime avg(bytes_out) as
avg stdev(bytes_out) as stdev
| eval notable=avg + 3*stdev
| where bytes_out > notable
| where _time > relative_time(now(),"-30m")
| lookup dnslookup clientip AS src OUTPUT clienthost AS host
| table _time firstTime lastTime src host user category dest bytes_out notable

| eval chain = "Exfiltrate" | eval semantic = "Proxy Large Outbound"
| stats min(firstTime) as firstTime  max(lastTime) as lastTime by host user
semantic chain
| table firstTime lastTime user host semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
```

Figure 27: Exfiltrate Proxy Large Outbound

Breaking down "Destination IP," identify any outbound connection to an IP that is uncached and untrusted. Within the event taxonomy, proxy filtering offers user and host based connection monitoring including the destination.

Create and schedule a semantic query with context as seen below in Figure28: Exfiltrate Proxy Dest IP28. First, select only those destinations that are IPs. Then determine if the destination is untrusted by using the iplocation command to select untrusted countries. Select and save the earliest time and latest time for the host, user, and destination. Then select only those IPs never previously seen using the difference from the earliest time and latest time. Finally select and save indicators including time, user, host, and optional details such as destination and country.

```
earliest=-20m@m latest=-5m@m
index=proxy
| rex field=dest "\b(?<ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})\b" | search ip=*
| stats min(_time) as firstTime max(_time) as lastTime by src user dest
| lookup dnslookup clientip AS src OUTPUT clienthost as host
| iplocation dest | search Country="UntrustedCountry"
| table firstTime lastTime host user dest Country

| inputlookup append=T INFOSEC-CTX-PROXY-DEST-IP.csv
| where lastTime > relative_time(now(), "-30d")
| stats min(firstTime) as firstTime max(lastTime) as lastTime by host user dest
| table firstTime lastTime host user dest
| outputlookup INFOSEC-CTX-PROXY-DEST-IP.csv

| eval diffTime = lastTime-firstTime
| where lastTime-firstTime <= 300

| eval chain = "Exfiltrate" | eval semantic = "Proxy Dest IP"
| stats min(firstTime) as firstTime max(lastTime) as lastTime by host user
semantic chain
| table firstTime lastTime host user semantic chain
| inputlookup append=t INFOSEC-CHAIN.csv
| where lastTime > relative_time(now(), "-24h")
| stats first(firstTime) as firstTime last(lastTime) as lastTime by host user
semantic chain
| table firstTime lastTime host user semantic chain
| outputlookup INFOSEC-CHAIN.csv
```

Figure28: Exfiltrate Proxy Dest IP

2.8. Triggering and Reporting the Kill Chain

Trigger the Kill Chain by finding the proper sequence of chain events in the Kill Chain context. Splunk offers the ability to track the sequence of events using the transaction command. The transitive transaction property allows events missing data such as the user or host to correlate with the chain with the caveat that the missing data be null (Garner, 2009). It is the transitive transaction properly that makes Splunk a unique and powerful tool.

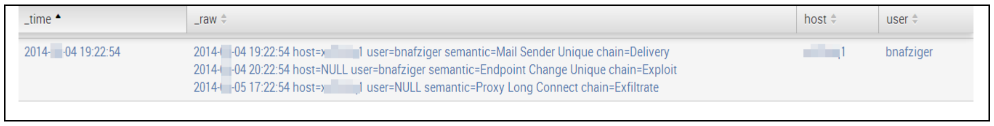
Create and schedule the kill chain query as seen below in Figure **29: Kill Chain Query**29. To enable the transitive property, null the value of fields for data that is not fully known such as user=SYSTEM or host=None. Use several transactions to build each section of the Kill Chain. Select those chains that match the full path. Finally, display the results and any pertinent asset, identity or vulnerability details.

It is quite possible to simulate attacks traversing the kill chain by a combining a variety of actions such receiving unique emails, installing applications that touch load points and visiting unique sites. However, the current framework implementation very quickly triggered on potential production attacks traversing the kill chain. Even so, the ability to inject events using an inject csv file is trivial and is useful for testing transitive transactions. Therefore, the inject lookup is included.

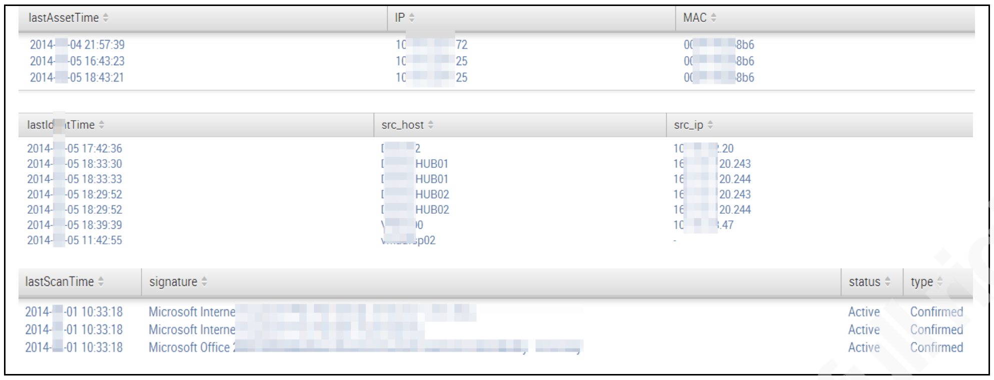
```
$SPLUNK_HOME\etc\apps\search\lookup\INFOSEC-CHAIN-INJECT.csv
firstTime,lastTime,host,user,semantic,chain
NONE,NONE,TARGET,JOHN,Mail Sender Unique,Delivery
NONE,NONE,TARGET,JOHN,Endpoint Change Unique,Exploit
NONE,NONE,TARGET,JOHN,Proxy Long Connect,Exfiltrate

| inputlookup INFOSEC-CHAIN.csv

| COMMENT append inject for testing only, remove for production
| append [ |inputlookup INFOSEC-CHAIN-INJECT.csv

| eval lastTime = if(chain=="Delivery",relative_time(now(),"-23h"),lastTime)
| eval host = if(chain=="Delivery","WORKSTATION",host)
| eval user = if(chain=="Delivery","bnafziger",user)

| eval lastTime = if(chain=="Exploit",relative_time(now(),"-22h"),lastTime)
| eval host = if(chain=="Exploit",NULL,host)
| eval user = if(chain=="Exploit","bnafziger",user)

| eval lastTime = if(chain=="Exfiltrate",relative_time(now(),"-1h"),lastTime)
| eval host = if(chain=="Exfiltrate","WORKSTATION",host)
| eval user = if(chain=="Exfiltrate",NULL,user)
]

| eval user = if(user="None",NULL,user)
| eval user = if(user="SYSTEM",NULL,user)
| eval tuser = if(isnull(user),"NULL",user)
| eval host = if(host="None",NULL,host)
| eval thost = if(isnull(host),"NULL",host)

| eval _time=lastTime
| eval _raw = strftime(lastTime , "%Y-%m-%d %H:%M:%S")
        +" host="+thost+" user="+tuser+" semantic="+semantic+" chain="+chain

| transaction host user
        connected=f mvraw=t delim="\n\n" maxspan=-1 keepevicted="t"
        endswith="Delivery"
| transaction host user
        connected=f mvraw=t delim="\n\n" maxspan=-1 keepevicted="t"
        startswith="Delivery" endswith="Exploit"
| transaction host user
        connected=f mvraw=t delim="\n\n" maxspan=-1 keepevicted="t"
        startswith="Exploit" endswith="Exfiltrate"
| transaction host user
        connected=f mvraw=t delim="\n\n" maxspan=-1
        startswith="Delivery" endswith="Exfiltrate"

| table _time _raw  host user | search Delivery Exploit Exfiltrate

| lookup INFOSEC-CTX-ASSET-DYNAMIC Host as host
        OUTPUT lastTime IP MAC
| eval lastAssetTime = strftime(lastTime , "%Y-%m-%d %H:%M:%S")

| lookup INFOSEC-CTX-IDENTITY-DYNAMIC src_host as host
        OUTPUT lastTime host as src_host src_ip
| eval lastIdentTime = strftime(lastTime , "%Y-%m-%d %H:%M:%S")

| lookup INFOSEC-CTX-VULNERABILITY-DYNAMIC host
        OUTPUT lastTime signature status type
| eval lastScanTime = strftime(lastTime , "%Y-%m-%d %H:%M:%S")

| table _time _raw  host user
        lastAssetTime IP MAC
        lastIdentTime src_host src_ip
        lastScanTime signature status type
```

Figure 29: Kill Chain Query

In the end, numerous events in production triggered potential attacks in the kill chain as seen below in Figure 30: Kill Chain Results30 in varying slices of real events. Though not included below, additional static details, user full name and address, are easy to add using lookups.



_time	_raw	host	user
2014-[redacted]-04 19:22:54	2014-[redacted]-04 19:22:54 host=x[redacted]1 user=bnafziger semantic=Mail Sender Unique chain=Delivery	[redacted]1	bnafziger
	2014-[redacted]-04 20:22:54 host=NULL user=bnafziger semantic=Endpoint Change Unique chain=Exploit		
	2014-[redacted]-05 17:22:54 host=x[redacted]1 user=NULL semantic=Proxy Long Connect chain=Exfiltrate		



lastAssetTime	IP	MAC
2014-[redacted]-04 21:57:39	10[redacted]72	00[redacted]8b6
2014-[redacted]-05 16:43:23	10[redacted]25	00[redacted]8b6
2014-[redacted]-05 18:43:21	10[redacted]25	00[redacted]8b6

lastId[redacted]tTime	src_host	src_ip
2014-[redacted]-05 17:42:36	[[redacted]2	10[redacted].20
2014-[redacted]-05 18:33:30	[[redacted]HUB01	16[redacted]20.243
2014-[redacted]-05 18:33:33	[[redacted]HUB01	16[redacted]20.244
2014-[redacted]-05 18:29:52	[[redacted]HUB02	16[redacted]20.243
2014-[redacted]-05 18:29:52	[[redacted]HUB02	16[redacted]20.244
2014-[redacted]-05 18:39:39	V[redacted]0	10[redacted].47
2014-[redacted]-05 11:42:55	[redacted]p02	-

lastScanTime	signature	status	type
2014-[redacted]-01 10:33:18	Microsoft Interne[redacted]	Active	Confirmed
2014-[redacted]-01 10:33:18	Microsoft Interne[redacted]	Active	Confirmed
2014-[redacted]-01 10:33:18	Microsoft Office [redacted]	Active	Confirmed

Figure 30: Kill Chain Results

3. Conclusion

Practical Threat Detection using Big Data, Semantic Methods, and Kill Chains have previously proven to work individually and consequently at least minimally in practice as seen in this framework; however, the tool, methods and model are challenging to build and tune in a comprehensive and synergistic manner. The initial results at this time are promising and do notate higher risk but not necessarily true positive alerts. Even though the initial results are promising, the framework will require continued efforts to generate true positive alerts. It is the hope that the community of security practitioners continues the efforts.

Several challenges occurred throughout the exercise though no challenge was insurmountable. Two of the challenges were dirty data and latent data. Dirty data, data that did not parse properly, required extra effort to extract to a level of usability. Numerous regular expressions within the queries extracted the data exceptionally well. Latent data, data that slowed in transit, required some delays in the queries to extract all the data. Running queries in a delayed manner, gathered the data exceptionally well.

Several areas of future work exist. More data sources need integrated - firewall, IDS, VPN, etc. More contexts need added - high-risk users, high-risk assets, etc. More semantic methods need defined - threat data, machine learning, r project integration, etc. More semantic method tuning needs completed – standard deviations require bell distributions, etc. More kill chain tuning needs completed - the current kill chains focused on post compromise however further kill chains could include pre compromise.

The current kill chain focused on time, host and user, further tuning could include scope based or other elements. Finally, further tuning could include weighting of semantics.

"We're witnessing the end of the data age, and the first sparks of the age of analysis" (Palantir, 2008).

4. References

Alspaugh, S., Ganapathi, A., Hearst, M., & Katz, R. (2013). Building Blocks for Exploratory Data Analysis Tools. Retrieved July 1, 2014, from http://www.eecs.berkeley.edu/~alspaugh/papers/lsa_idea_2013.pdf

Aziz, A. (2011, October 20). Malware & APT risks for Critical Infrastructures. Retrieved July 1, 2014, from http://www.nerc.com/files/10_Aziz_NERC_Malware_APT_Risk_FireEye.pdf

Benson, C. (n.d.). Security Threats. *Security Threats*. Retrieved June 18, 2014, from http://technet.microsoft.com/en-us/library/cc723507.aspx

Bejtlich, R. (2013). *The Practice of Network Security Monitoring Understanding Incident Detection and Response*. San Francisco: No Starch Press.

Bejtlich, R. (2006). *Extrusion detection: Security monitoring for internal intrusions*. Upper Saddle River, NJ: Addison-Wesley.

Bejtlich, R. (2005). *The Tao of network security monitoring: Beyond intrusion detection*. Boston: Addison-Wesley.

Bianco, D. (2013, March 1). Enterprise Detection & Response: The Pyramid of Pain. Retrieved June 18, 2014, from http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html

Bianco, D. (2013, September 14). Enterprise Security Monitoring. . Retrieved May 1, 2014, from http://www.slideshare.net/bsidesaugusta/david-bianco-enterprise-security-monitoring

Bijou, R. (2013, March 15). Examining the Cyber Kill Chain. . Retrieved May 1, 2014, from http://www.rbijou.com/2013/03/15/examining-the-cyber-kill-chain/

Brownlee, J. (2014, January 6). What is Data Mining and KDD. . Retrieved June 18, 2014, from http://machinelearningmastery.com/what-is-data-mining-and-kdd/

Carr, J. (2008). . Retrieved July 1, 2014, from http://jeffreycarr.blogspot.com/2013/08/the-cyber-kill-chain-trademarked-by.html

Chuvakin, A., Knapp, E. (2010, January). Content Aware Siem Define. . Retrieved July 1, 2014, from http://www.enterprisemanagement360.com/wp-content/files_mf/white_paper/content_aware_siem_defined.pdf

Cloppert, M. (2010, July 1). Intelligence-Drive Response. . Retrieved May 1, 2014, from http://digital-forensics.sans.org/summit-archives/2010/36-cloppert-primary-deck.pdf

Cole, E. (2012). Advanced Persistent Threat: Understanding the Danger and How to Protect your Organization. Waltham, MA: Syngress.

Dean, J., Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. Retrieved July 1, 2014, http://static.googleusercontent.com/media/research.google.com/en/us/archive/mapreduce-osdi04.pdf

Duncan, A. D. (2014, February 1). The ABC of Data Governance: Driving Information Excellence. Retrieved June 18, 2014, from http://www.slideshare.net/AlanDDuncan/the-abc-of-data-governance

Garner, M. (2009, January 17). Splunk for Xitive Xactions. Retrieved July 1, 2014, from http://blogs.splunk.com/2009/01/17/splunk-for-xitive-xactions/

Giura, P., & Wang, W. (2013, December). Using Large Scale Distributed Computing to Unveil Advanced Persistent Threats. Academy of Science and Engineering Science Journal, 1(3), pp. 93-105.

Goldsmith, T. (2014, April 2). Centre for the Protection of National Infrastructure Effective Log Management. . Retrieved May 21, 2014, from http://contextis.com/files/CPNI-CTX_-_Effective_Log_Management.pdf

Hancock, Jr., M. F. (n.d.). Data Mining. . Retrieved June 18, 2014, from http://www.celestech.com/test/PracticalDataMining/DataMiningSlideBank.ppt

Hollingworth, D. (2003, June 26). Towards Threat, Attack, and Vulnerability Taxonomies. . Retrieved June 18, 2014, from http://webhost.laas.fr/TSF/IFIPWG/Workshops&Meetings/44/W1/02-Hollingworth.pdf

Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2010, November 21). Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary

Campaigns and Intrusion Kill Chains. . Retrieved May 1, 2014, from http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf

IT Glossary. Big Data. Retrieved June 18, 2014, from http://www.gartner.com/it-glossary/big-data/

Jakhar, A. (2008). SPAM - The Evolution. Retrieved July 1, 2014, from https://www.blackhat.com/presentations/bh-europe-08/Jakhar/Whitepaper/bh-eu-08-jakhar-WP.pdf Laney, D. (n.d.).

Lin, M. S., Chiu, C. Y., Lee, Y. J., & Pao, H. K. (2013, October). Malicious URL Filtering—A Big Data Application. In 2013 IEEE International Conference on Big Data (pp. 589-596). IEEE.

Mandiant. (2010). . M-Trends: The Advanced Persistent Threat. Retrieved July 1, from https://www.mandiant.com/resources/mandiant-reports/

Mandiant. (2013, February 18). . APT1: Exposing One of China’s Cyber Espionage Units. Retrieved July 1, from http://intelreport.mandiant.com/Mandiant_APT1_Report.pdf

Opitz, D., & Macline, R. (1999, August). Popular Ensemble Methods: An Empirical Study. Retrieved July 1, 2014 from http://jair.org/media/614/live-614-1812-jair.pdf

Palantir. (2008). . Retrieved July 1, 2014, from https://www.palantir.com/2008/10/palantir-government-conference/

Palantir. (2013, August 22). . Retrieved May 1, 2014, from http://www.palantir.com/_ptwp_live_ect0/wp-content/uploads/2013/11/Palantir-Solution-Overview-Cyber-long.pdf

Robb, C. (2011, October 1). The Heart of the Matter A Core Services Taxonomy for State IT Security Programs. . Retrieved June 18, 2014, from http://www.nascio.org/publications/documents/NASCIO_CoreSecuritySevices.pdf

Russinovich, M., & Cogswell, B. (2014, August 18). Autoruns for Windows. Retrieved August 18, 2014, from http://technet.microsoft.com/en-us/sysinternals/bb963902.aspx

Sanders, C., & Smith, J. (2013). *Applied network security monitoring collection, detection, and analysis*. Waltham, MA: Syngress.

Schneider, F., ed. (1999) *Trust in Cyberspace*, National Academy Press

Shakarian, P., & Shakarian, J. (2013). *Introduction to Cyber-Warfare a Multidisciplinary Approach*. Amsterdam [Netherlands: Morgan Kaufmann Publishers, an imprint of Elsevier.

Singh, K., Guntuku, S. C., Thakur, A., & Hota, C. (2014, March 29). Big Data Analytics framework for Peer-to-Peer Botnet detection using Random Forests. . Retrieved May 1, 2014, from http://www.bits-pilani.ac.in/uploads/Hyderabad/hota/kamal-sarath-hota.pdf

Splunk. (2014). Common Information Model. . Retrieved July 1, 2014, from http://docs.splunk.com/Documentation/CIM/latest/User/Overview

Splunk. (2014). DB Connect App. . Retrieved July 1, 2014, from http://www.splunk.com/view/db-connect/SP-CAAAHR6

Splunk. (2014). LDAP Search Add On. . Retrieved July 1, 2014, from http://docs.splunk.com/Documentation/ActiveDirectory/1.2.2/DeployAD/ConfiguretheSA-ldapsearchsupportingaddon

Splunk. (2014). Lookup Search Reference. . Retrieved July 1, 2014, from http://docs.splunk.com/Documentation/Splunk/6.1.3/SearchReference/Lookup

Sweden, E. (2009, March 1). Data Governance Part II. . Retrieved June 18, 2014, from http://www.nascio.org/publications/documents/NASCIO-DataGovernancePTII.pdf

Symantec. (2012, February 15). WS.Reputation.1. Retrieved July 1, 2014, from http://www.symantec.com/security_response/writeup.jsp?docid=2010-051308-1854-99&tabid=2

Symantec. (2013, October 18). Symantec Insight and SONAR. Retrieved July 1, 2014, from http://www.symantec.com/connect/articles/symantec-insight-and-sonar

Talabis, M. (2007). Security Analytics Project. . Retrieved July 1, 2014, from https://www.blackhat.com/presentations/bh-usa-07/Del_Moral_Talabis/Presentation/bh-usa-07-del_moral_talabis.pdf

Tirpak, J. (2000, July 1). Find, Fix, Track, Target, Engage, Assess. . Retrieved June 2, 2014, from http://www.airforcemag.com/magazinearchive/pages/2000/july%202000/0700find.aspx

UCISA. (n.d.). ITIL – A guide to event management. .Retrieved September 1, 2014, from https://www.ucisa.ac.uk/~/media/Files/members/activities/ITIL/service_operation/eventm_management/ITIL_a guide to event management pdf.ashx

Verizon. (2013, April 22). Data Breach Investigations Report. . Retrieved May 1, 2014, from http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2013_en_xg.pdf

Xin, Kai, (2013, October). Good Enough Analytics. Retrieved July 1, 2014, from http://www.slideshare.net/KaiX/good-enough-analyticsfinal

Yen, T. F., Oprea, A., Onarlioglu, K., Leetham, T., Robertson, W., Juels, A., & Kirda, E. (2013, December). Beehive: Large-Scale Log Analysis for Detecting Suspicious Activity in Enterprise Networks. In Proceedings of the 29th Annual Computer Security Applications Conference (pp. 199-208). ACM.