A Formal Understanding about APT Infection

***Abstract*—Nowadays APT (Advanced Persistent Threat) breaches are becoming inevitable, because determined threat actors will always find an insidious way through the gap. Perimeter protection can do little once a foothold has been established by an APT. Modern systems are essentially black boxes and provide very limited visibility of their internal data relationships. Thus, it is very difficult to understand and investigate the infection and spread of an APT. As opposed to attack-centric approaches, this paper focuses on data relations and proposes some formal methods for considering the spatial and causal graphs of infected data by an APT attack. They are called Infection Spread (IS) and Infection Graph (IG). Our approaches can well describe the data dependencies of cyber system components and help people have a qualitative and quantitative understanding of how an APT might walk laterally. It can also provide great values for post-breach assessment once an APT is identified.**

I. Advanced Persistent Threat

Everyone now realizes what security professionals have long been aware of: there is no such thing as perfect security. Security breaches are becoming inevitable, because determined threat actors will always find an insidious way through the gap.

The traditional dangers that security teams have been facing for years are being replaced by a far more hazardous form of attack: the Advanced Persistent Threat (APT). An APT is a network attack in which an unauthorized person gains access to a network and stays there undetected for long time. APT attacks target organizations in sectors with high-value information, such as national defense, and the financial industry. The intention of an APT attack is to steal data rather than to cause damage to the organization [1,2,3,4,5,6].

As illustrated in figure 1, nearly every APT follows four phases: **Reconnaissance, Initial Entry, Escalation of Continuous Privileges, and Exploitation** [7]. An investigation

Figure 1. Cyber Kill Chain

into the organization's weaknesses, which often includes domain queries and port and vulnerability scans. Discovered exposures are exploited and a foothold in the target network is established using sophisticated technical methods or social engineering techniques. Following initial penetration, hackers walk laterally to acquire more rights and gain control over additional systems and install back doors for future access. Once a control has been established, the APT will be able to continuously infect, compromise and exploit more data.

From defense perspective, a defender system need well understand how the APT will continually spread and walk laterally before it reaches the target data; and then IT team can adopt corresponding deployment and protection strategies to stop the APT infection. To achieve this goal, the defender need a good understanding of their data relations.

As of today, modern enterprise IT systems are essentially black boxes and provide very limited visibility of their internal data relations. This greatly limits the potential to understand APTs in depth. A general lack of understanding of complex systems interferes with efforts to diagnose advanced attacks that span multiple applications and systems. Very few of research work has been done to address this concern [29].

II. Breach Visibility & Current Problems

In late November to early December 2013, Target Corporation announced that around 40 million credit and debit cards data was stolen. It is the second largest credit and debit card breach after the TJX Companies data breach where almost 46 million cards were affected.The cost of the data breach could be up to total $1B dollars [8].

The Community Health Systems(CHS) breach exposing 4.5 million patients' data in 29 states on August, 2014 is expected to be costly–the total bill could be somewhere between $75 million and $150 million, according to a calculation at Forbes[8].

In February 2015, Anthem suffered a data breach of nearly 80 million medical records. The company's cyber insurance policy is likely to be exhausted. The financial consequences could reach beyond the $100 million mark [8].

The Heartbleed bug is considered to be one of the most catastrophic vulnerabilities because it enabled anyone to read the memory of systems protected by vulnerable versions of OpenSSL. At Heartbleed disclosure on April 7, 2014, around 17% or half a million of the Internet's secure web servers certified by trusted authorities were believed to have been vulnerable to the attack. When this vulnerability was reported, some organizations were thinking that no confidential data was able to be read out of their servers [9,10,11,12].

From 2014 Threat Report that Mandiant Inc published, "The median number of days attackers were present on a victim network before they were discovered was 229 days" and "33% of the organizations had discovered the intrusion themselves." [13]

These days, the community's been asking below two questions:

- Is there a formal study to help IT understand the infection and spread of an APT after an initial entry being exploited?

- Is there a good methodology to guide an organization how to distribute its data to avoid data breaches?

We think that the lack of understanding the data relations within a complex system contributes the most outstanding reason for not able to answering the above questions.

As of today, most of industry organizations have been highly relying on experienced security professionals, or spend lots of money to hire professional services from third parties, e.g., FireEye Inc. This is apparently not sufficient neither efficient to address all the security concerns.

On the other hand, in academic area, most of research works have been attack-centric only, focusing on malware or viruses scripts or binary detection; very few of studies are related to data relation analysis [26, 27].

Investigating security vulnerabilities, and the possible damage inflicted on a system, tends to be a very complicated process. Therefore, it is strongly encouraged for having a good formal approach that provide a succinct way to understand the data relations.

In this paper, we address the above issues from data relation perspective. In section 3 and 4, we first introduce and discuss the spatial and causal relations for an enterprise's data. Then we propose some formal methods called Infection Spread(IS) and Infection Graph (IG), which can be used to depict the infection path of an APT. We think the data-centric approach could greatly help people understand the data dependencies for infected data, and it can well support root cause analysis and post-breach assessment once an adversary activity is identified. In section 5, as a case study, we use the IS and IG methods to formally describe the Heartbleed-based APTs behavior particularly for CHS data breach case, and then we give a qualitative and quantitative analysis and discussion.

In section 6 and 7, we compare our work with those attack-centric research studies that have been done for decades, and give some discussions about how to use our methods to help an enterprise do the data separation so as to avoid single point of breach (SPOB).

III. Data Relation and Data Space

A. Data Relations

A formal understanding of data relations of a system is important to address APT spreading problems after initial entry. In this section, we first introduce two important concepts, which are Spatial Relation and Causal Relation. We think that any data in an organization holds both spatial and causal attributes.

Definition: Data

Data being discussed herein in this paper is but not limited to, cache lines in CPU or multi-cores, user credentials or keys for VPN services, a process's bss, stack or heap, an OS kernel data, meta-data for a cluster, an VM, an orchestration platform for a cloud, or a disk-array file sytem, or a whole cloud SAN storage.

Definition: Spatial Relation (SR)

Two data holds a spatial relationship if they are managed by the exactly same security policies. Depending on whether the data are persistently close to each other, e.g., database table entries, or just happen to be located together temporally, e.g., cache lines, we call them Persistent SR (PSR) and Temporal SR(TSR) correspondingly. Spatial relation means that having access to one data will have the capability to access its all spatial related data.

We define $aSRb$ if a and b has a spatial relation. $aSRb$ is symmetric. If $aSRb$, then $bSRa$. For example, different cache lines in CPU caches, threads data, bss sections, or heap data in the same process holds TSR, while different tables inside one database or different files in one hard disk, are holding an PSR.

Definition: Causal Relation (CR)

Two data holds a causal relationship if having access to one data will lead to obtaining the access privileges to another data. For example, user/password data has the CR relation with a users data; kernel space and application space has the CR relation, and SQL roots info has the CR with the data inside the database. We define $aCRb$ if a and b has a causal relation. Apparently, Causal relation is NOT symmetric. $aCRb$ does not mean $bCRa$. For instance, a meta-data usually is causal related to the data that it manages. But the otherwise is not true.

Definition: Concurrent Relation(CN)

Two data are concurrent if they are neither spatial related nor causal related during their lifecycles. We define $aCNb$ if a and b has a concurrent relation.

B. Data Space

Based on the data relations for a set of data, we can define the Data Space to represent the whole data of an organization. As illustrated in figure 2, a data space consists of a set of

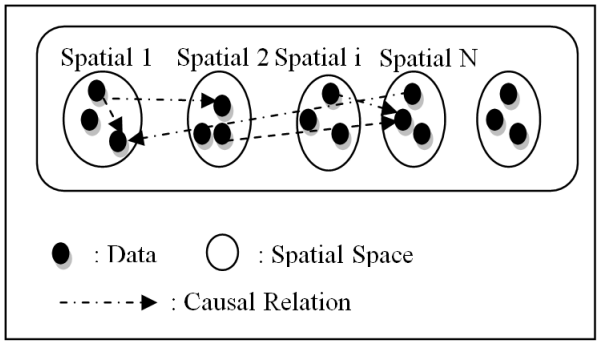

Figure 2. Data Space

spatial spaces. Each spatial space consists of a set of data. We define the relations between two spatial spaces as follows:

Related Data Spaces: Two spatial spaces A and B are related if there at least exist one data a in space A ; and a has at least one causal relation with one data in space B. We can formally define it as:

$$\exists a \in A, \quad and \quad \exists b \in B,$$

We have:

$$aCRb, \quad or \quad bCRa$$

Unrelated Data Spaces: Two spatial spaces A and B are unrelated if there does NOT have any data in space A and B that holds a causal relation. In other words,

$$\nexists a \in A, \quad or \quad \nexists b \in B,$$

We have:

$$aCRb, \quad or \quad bCRa$$

For data in a spatial space, it can hold a causal relation with another data either in the same or in a different spatial space. Any data in a data space has two attributes/tags:
spatial attribute and causal attributes.
Spatial Attribute represents which spatial space that one data is located at. One data at least belongs to one spatial domain. It can cross multiple spatial spaces. For instance, a distributed database.
Data in the same spatial relation shares the same security domain, et al., the same VLAN, file system, Windows Group/Domain, or firewall policies.
Causal Attribute is the information about that one data has a causal relation with other data. And these related data could be in the same or different spatial domain.
In the following sections, we will use the spatial and causal relations to formally describe the spread of an APT.

IV. Infection Spread and Infection Graph

A. Malicious Operations

We argue that an action of malicious operations is either a read or a write action on data, which could be the stack or heap of a thread/process, root/password info of a kernel or a vm, or a database. Malware usually is stealthy, intended to steal one of the above data or spy on computer users for an extended period without their knowledge.
For malicious operations, we define the following notations:
Malicious Read (MR): Any READ behavior that aims to steal data out of an organization.
Malicious Write(MW): Any WRITE behavior that could do harm on the data set including taking a system component down, or setting up a foothold for maintaining presence.
Formally, we define MO(Malicious Operations) as:

$$MO = \{MR, MW\}$$

Definition: We define **m** $\otimes$ **d** as a binary relation between an MO and an DATA, such that m is able to execute an MR or/and MW operations on d. We call it m infect d, or "d is infected by m.
Malicious Spanning: A malicious operation has a capability called "Malicious Spanning". And the spanning domain is its currently located spatial space. For example, if a malicious read is able to breach the heap of a process, it can meantime read other parts of the heap data; if a database is breached, all the tables are able to be read with a brutal force based scanning.

B. Infection Spread

Definition: We define "**a**$\rightarrow$ **b**" as a binary relation between two data, such that if one data is infected, another data could also be infected. We call this relation **Infection Spread (IS)**. For infection spread, there exist **Spatial Infection Spread** and **Causal Infection Spread**.

Theorem 1. ***If aSRb, a$\rightarrow$b spatially if a is infected.***

Proof. If a and b holds a SR relation; and a is infected by a malicious operation m,where $m \in MO$, and m$\otimes$a; because a and b are adjacent to each other and share the same security policies, then m will be able to also read/write b, thus m$\otimes$b. b is infected. □

Similarly, we can derive that, if one data in a spatial space is infected, the whole space is infected. An infection is able to span over to the whole spatial domain.

Theorem 2. ***If aCRb, a$\rightarrow$ b causally if a is infected.***

Proof. Suppose a is a privileged meta-data, and a and b holds a **CR** relation. If a is infected with a malicious operation m, where $m \in MO$, and $m \otimes a$, then with the privileged information derived from a, $\exists n \in MO$, $n \otimes b$. Then b is breached. For example, a malware can use VPN to attack data b after having obtained private key data from a; APTs usually use the causal infection to install new backdoors which is different than the ones installed initially and maintain the continued presence. □

Moreover, we argue that infection relationship $\rightarrow$ holds reflexive, transitive and conditional symmetric properties.

$\forall a, a \rightarrow a(reflexivity)$
$\forall a, b, c, if\, a \rightarrow b, and\, b \rightarrow c,\, then\, a \rightarrow c(transitivity)$;
$\forall a, b, if\, a \rightarrow b,\, and\, aSRb, then\, b \rightarrow a.$

In other words, spatial infection spread is symmetric. However, causal infection spread is not.

For the data in a data center, we have the following corollaries.
Corollary 1: If a and b are thread data on the same process, $a \rightarrow b$ spatially if a is infected.
Corollary 2: A whole process data could be infected if a particular thread data is infected.
Corollary 3: If a is a kernel data and b is a process data on top of the kernel, $a \rightarrow b$causally if a is infected.
Corollary 4: A whole virtual machine data could be infected if a kernel is infected.
Corollary 5: If a is an orchestration data and b is a vm data, $a \rightarrow b$ causally if a is infected.
Corollary 6: A whole cloud data could be infected if the orchestration management is infected.

C. Infection Graph

Infection Graph is defined as an ordered pair $IG = (DATA, IS)$,

- Data: a data set which contains data of thread, process, kernel, VM, or a whole cloud data.
- IS: a set of ordered pairs of data; and each of which represents an Infection Spread Path, e.g., $a \rightarrow b$

We consider the following attributes for an infection graph.

1) **Infection Order** of an IG is $|DATA|$ (the total number of being infected data).
2) **Infection Size** is $|IS|$, the amount of infection paths.
3) **Infection Degree** of data a is the number of edges that connect to it.
4) **Infection Distance**: Given a data path from a to b, an infection distance is defined by the middle stages between a and b. Each stage represents an independent breach.
5) **Infection Probability**: Assume an infection distance is n from data a to b. It means that an APT, after successfully built a foothold at the data a position, it still need walk through n steps in order to reach the target data b. If for every step, the probability of being breached is $p_i (i = 1, 2, 3, \ldots, n)$, we can simply have the Infection Probability being defined as follows:

Theorem 3. ***The probability that an APT can successfully breach a target data is:***

$$IP(a, b) = p_1 \times p_2 \times p_3 \ldots \times p_n = \prod_{i=1}^{n} p_i$$

where
a: the foothold position that an APT established through initial entry.
b: the target data.
n: the infection distance.
p_i: the infection probability of ith step.

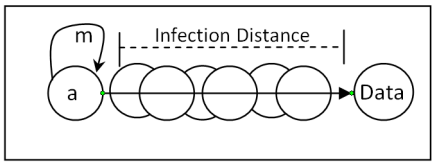

Figure 3. Infection Distance

Discussion:
The above five properties of an infection graph are important for measuring an APT qualitatively and quantitatively. For instance, how persistent and how hard an APT would be.
Infection Degree is a good meter for measuring how redundant an APT is. For example, an APT may use multiple zero-day vulnerabilities to breach a system, instead of only one. an APT breach path still exists even if one path get removed.
The Infection Distance and the Infection Probability can be well used to evaluate the "difficulty" of an APT, and provide a good guidance for an IT to deploy data to avoid the intrusion of APTs.

1) When more n steps are needed to reach b from a; the overall $IP(a, b)$ probability is smaller; and thus it is becoming more difficult for this APT to breach the data b.

2) If for each intermediate step, the smaller of p_i, probability is, the the overall probability $IP(a, b)$ is proportional smaller correspondingly; an APT is becoming harder to breach the target data.

For simplicity purpose, in this paper, we use $(1/e)$ to express the infection probability for each step, thus the whole infection possibility from a to b is $(1/e^n)$. Then, we can draw a diagram to illustrate the probability distributions of an IP in figure 4.

We can tell, after an infection distance is longer than 3,

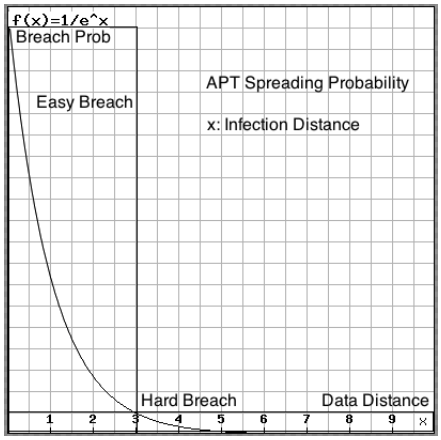

Figure 4. Infection Probability

the probability of the system being breached is shrinking exponentially. and when the distance is 5, the probability is becoming nearly trivial close to zero. This insightful findings could be well used for how to design a defense system. For example, increase the defense chains, so as to make the APT gives up its breach attempts.
With the notations of infection distance and infection possibility, we can define the hardness of a data breach as below.
Definition: Easy Data Breach
An APT attack is an easy data breach if the infection distance between the initial entry and the target data is less than 3.
Definition: Hard Data Breach
An APT attack is a hard data breach if the infection distance between the initial entry and the target data is more than 3.
As we know, deploying an APT itself is a very expensive behavior; and has to be very economic effective. An individual or/and a nation sponsored actor need to design an APT very

sophisticated in order to achieve the attack goal. In other words, **An APT defender can either/both increase the n or/and decrease the p in our theorem 3 to enforce an APT's behavior fall through the Hard Breach zone in our figure 4.**

Next, we will start to define some basic building blocks of infection graphs. We find that any advanced APT spread can be depicted by the composition of these basic patterns.

Initial Infection:

Figure 5 shows an Infection Graph $IG = (\{a\}, \{NULL\})$,

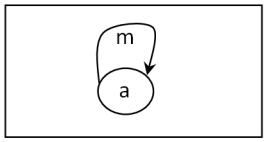

Figure 5. Initial Infection

where a is infected by an vulnerability m, $m \otimes a$. This is the simplest infection graph. Usually, an initial infection is mapped to an APT initial entry during its cyber kill chain.

Spatial Infection:

Figure 6 is a spatial infection evolved from the initial

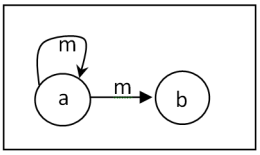

Figure 6. Spatial Infection

infection. For example, an APT walked latterly. After data a is infected by m, data b is infected by the same m operation. $IG = (\{a, b\}, \{(a \rightarrow b)\})$, where $m \otimes a$, $and\, m \otimes b$. An spatial infection usually is the result of an APT spanning behavior.

Causal Infection:

Figure 7 is a causal infection graph. data a is first infected

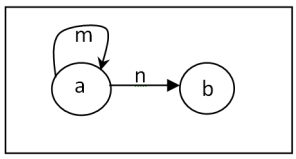

Figure 7. Causal Infection

by m, and then m is able to extract the access privilege information of b from a. After that, the APT is able to pick a corresponding malicious operation to attack b. $IG = (\{a, b\}, (a \rightarrow b))$, where $m \otimes a, and\, n \otimes b$. a and b holds a causal infection relationship. In most cases, a causal infection of being managed data could happen after a meta-data is breached.

With the above three basic building blocks, we conclude that we can draw some more complicated APT spread patterns.

Single Route Infection:

Suppose there existed an APT attack and it first set up a foothold at data a, and then walked through neighbor data b and c. And c is the target data. We formally depict this APT behavior as figure 8 and formal description as: $IG = (\{a, b, c\}, \{(a \rightarrow b), (b \rightarrow c)\}$, where $m \otimes a, m \otimes b$, $and\, n \otimes c$. For the spread, $(a, a), (a, b)$ are spatial infections, while (b, c) is a causal infection.

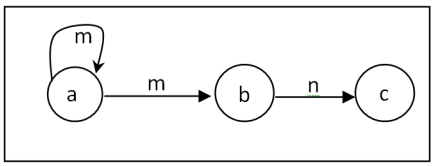

Figure 8. Single Route Infection

Hop Point Infection:

Modern APTs, especially, those supported by national governments, have been using hop points architecture to attack its target. A Hop Point sits in the middle between CC(Command Control) server and the target data so as to improve the APT persistence. Hop points are most frequently compromised systems that APT uses as camouflage for attackers [12, 13]. For figure 9, the APT behavior is that it first set up a foothold at one node, and uses that node as an CC, and then use two different zero-day vulnerabilities to compromise two hop points. These two hop points then separately breach the target data. And the confidential data will be stolen back to CC via either path. Formally, in the

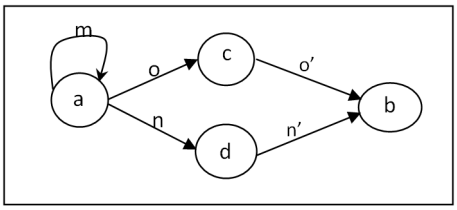

Figure 9. Hop Point Infection

figure 9, c and d are middle points, playing as hop points, while a is the initial exploit node, b is the APT target. $IG = (\{a, b, c, d\}, \{(a \rightarrow c), (a \rightarrow d), (c \rightarrow b), (d \rightarrow b)\})$, $where$
$m \otimes a, o \otimes c, n \otimes d$, ó$\otimes b\, and$ ń$\otimes b$

Compared to the single route pattern, hop point infection topology is more redundant and persistent. b is infected by two different vulnerabilities, and thus still under compromise if either one is found and get removed. Also, Hop Points infection size is double than single route, and thus can have multiple routes to reach target b(infection degree is 2), having redundant capabilities.

V. Evaluation:Heartbleed Exploitation and Infection

The root cause for CHS data breach was Heartbleed. A test server was hacked in April, 2014 by using the Heartbleed as initial entry; and then the attacker gradually walked through CHS intranet and took millions of medical data out [14,15].
In this section we use the infection graph methodology to formally describe the data breach that CHS experienced in last April. The APT used Heartbleed vulnerability as the initial entry [14,15]
The panic about Heartbleed from security experts is mainly because this bug left no traces of anything abnormal happening to the logs. And thus had no way of knowing who had exploited the flaw and what data had been stolen after the initial exploitation. Figure 10 is the final infection graph we

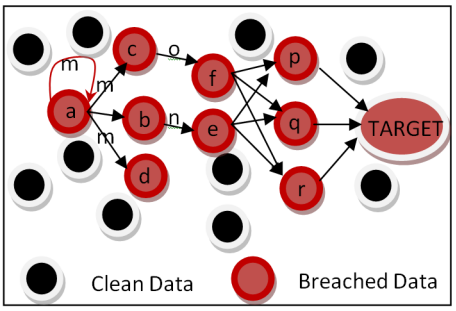

Figure 10. HeartBleed Infection Graph

created. The data set being involved in this case include:

1) a: heartbeat message
2) b: payload
3) c: user/password credentials
4) e: vpn data
5) f: ssh data
6) m: malicious function of dtls1_process_heartbeat(SSL *s)
7) o: malicious operation for ssh login
8) n: malicious operation for vpn

Stage 1: Initial Compromise
At the beginning, we have $m \otimes a$ as the Initial Compromise. And the hearbeat message will keep trying steal data [9]. The initial infection graph is: $Heartbleed = (a, NULL)$, where a is infected by m of dtls1_process_heartbeat; a is the heap data of https process openssl.
Stage 2: Scanning/Spear Phishing
Hackers successfully established a foothold after obtaining the private key. From figure 10, we can realize that, since a, b, c, d are set of data within the same process. From Lemma 1, we have $a \to b; a \to c; a \to d$. And all the infections hold the spatial attributes.
Stage 3: Establish Footholds
After the private key or some other credential data were breached, hackers were able to establish footholds and moved to another lifecycle of the infection. For CHSs breach graph, a new data node e: VPN Service is added into the graph, and a malicious operation n is correspondingly defined as using the private key extracted from b to login to VPN.
Meantime, APTs usually are not satisfied establishing only one foothold. They will try to make multiple ones whenever there is a chance. Hackers also fetched the compromised ssh user/password information. And then they are able to breach into ssh, further develop the APTs presence. Therefore, we add one new data node f: SSH service/data onto the infection graph, and the malicious operation o, which is identified as using user/password info extracted from c to remote SSH.
Stage 4: Walk laterally and Maintain Presence
Generally speaking, systems that the hackers initially compromised do not contain the data that they want [13]. For CHS medical record breach, the first node being breached was a testing server. The APT moved laterally within CHS network to other servers that either contain the target data or allow them to access it.
Suppose that, with one more step, the APT actor obtained the reading permissions of the CHS internal SQL database and was able to read all the medical data with standard queries.We assume the APT actor used both VPN and SSH footholds to walk through the internal networks and obtained 3 users credentials that have the right access privileges to target database.
Stage 5: Reach Target
The goal of an APT is to gradually reach the target data and steal it. The data usually includes intellectual property, business contracts, policy papers or military reports. Once APT actor find targetted data on compromised systems, the data will be packed and be sent out of the organization via a foothold, which was established previously. For Heartbleed example here, after the actor compromised p, q and r, they are able to reach the target data via an operation of SQL query. and the data will be sent back to one of the foothold f or e, and then sent back to the remote C&C server, which was controlled by the APT actor. We can obtain the final stage formal graph as below:
$Heartbleed = (\{a, b, c, d, e, f, p, q, r, target\},$
$\{(a \to b), (a \to c), (a \to d), (b \to e), (c \to f)$
$(f \to p), (f \to q), (f \to r), (e \to p), (e \to q),$
$(e \to r), (p \to target), (q \to target), (r \to target)\}).$

Discussion:
For the Infection Graph above that is derived from the intial Heartbleed attack, we can measure some of its important attributes as follows:

1) Infection Order: 10
2) Infection Degree : 3
3) Infection Size: 14
4) Infection Distance: 3

We conclude that the infection cost of Heartbleed is not as expensive as people thought. An attacker could easily use Heartbleed as an initial vulnerability and then breach a target data within 5 steps, which is an easy breach from our definition.

During the period that the APT tried to walk toward the target, 10 set of data were breached. Please note that, after a foothold was established, APTs usually walked very purposely and tried to be as close as possible to the target. For that purposes, APTs mostly tried use causal relations to gain more privileges or setup new footholds, rather than use spatial relation for infection at the beginning. We argue that a 5 stage APT attack usually is very difficult for maintaining its presence unless the C&C server is only 3 steps far. Generally, we believe that an 3 stage APT attack is the best tactic.

VI. Related Work

In the past decade has been done lots of research work to address various security issues. But very few of them is related to study the infection of an APT, which is still new to the community. Since APT actors are usually sponsored by financial sectors or even foreign national states, they are usually very sophisticated and advanced [2,3,4,5,6,13].

Most of research works on malware analysis and detection are either signature based, or behavior based [14-25]. While many progresses have been made, either of these approaches is facing big challenges. For instance, in order to simulate a complicated malware, it could take 5 minutes for us to evaluate its malicious behaviors; some viruses are encrypted and thus are very difficult for binary detector to decide and analysis whether it is bad or not.

Philips and Swiler [26] proposed the concept of attack graphs that tries to help IT professionals create an graph based on a set of known attack actions. And the analysis system explicitly required as input a database of common attacks, broken into atomic steps, specific network configuration and topology information, and an attacker profile. The attack information is matched with the network configuration information and an attacker profile in order to create an attack graph.

Sheyner, Haines, Jha and Wing have done many research on how to use model checking theory and technologies to automatically create the attack graphs based on a set of attack arsenals [26, 27, 28]. While Wing's approach is more general compared to the one that Philips and Swiler had proposed, all of their works shared some common limitations: Their approaches took an "attack-centric" view of the system [27]. And all their approaches need a set of pre-defined attack profiles/asenals in order to create a very large and complicated attack graphs. However, for modern APT attacks, it is almost impossible for security professionals to know which zero-day vulnerability that an intruder would use, or which vulnerability an IT system has.

Overall, most of related research paid little attention on the data relations, which is our research focus. We think that a general lack of understanding of complex data relations does interferes with efforts to understand advanced attacks that span multiple applications and systems.

VII. Contributions and Discussion

Our most contribution in this paper is that we formally define the data relations in an organization from a security perspective, and conclude that the spatial and causal relations among data contribute most for an APT's infection and spread. With the initial, spatial and causal three basic building blocks that we proposed, any complicated APT spread behaviors can be formally studied and composed for a qualitative and quantitative analysis.

We think that our formal studies can be well used for describing the data dependencies of cyber system components and provide a good visibility of internal data relations and then be able to better understand how an APT might walk laterally. Our methodology can also provide a good guidance to security teams for how to store and distribute business senstive data in an organization to avoid Single Point of Breach.

Rule 1: Avoid storing data in one single spatial space.
Given that whenever an initial node gets exploited then the whole spatial space could be spanned and breached, a good alleviation is to split data onto different spatial zones so as to avoid SPOB. The purpose is to limit the spanning capability of an APT. The infection distance and cost for the APT will go much higher if an APT wants to walk cross different spatial domains to get a full set of the target data. For example, when crossing different VLANs, firewall/IDS checks are usually applied.

Rule 2: Split meta-data to multiple spatial domains.
Meta-data is data of data. Losing meta-data usually means that the data being managed by the meta-data is under high risk. We should avoid storing important meta-data in one single spatial domain so as to avoid single point of breach. For instance, we should not store the user/password data together with its database in the same layer-2 domain, or same unix server. It is strongly encouraged that two data with a causal relation should be split at different spatial domains.

Rule 3: Strong authentication and authorization when a data flow is to cross different spatial spaces.
Modern systems are opaque and thus hard to monitor and control. Thats why DARPAs Transparent Computing (TC) program is looking for research efforts to provide high- fidelity visibility into data component interactions during system. The spatial and causal relations among data could provide a good basis so that a defender system can add strong authentication and authorization when a data flow is trying to cross over a spatial space.

Our future work is to design and implement a prototype APT detector based on our infection graphs; Also, we are interested in the infection network topology issues. We think that APTs maintaining presence and walk laterally are very interesting topics. We plan to use Clos network to measure how an APT attack can survive and maintain presence in dynamically changing systems.

References

[1] http://en.wikipedia.org/wiki/Advanced_Persistent_Threat
[2] Mandiant Corp, APT1: Exposing One of China's Cyber Espionage Units

[3] http://en.wikipedia.org/wiki/Network_security
[4] McAfee, Combating Advanced Persistent Threats.
[5] Trend Micro, Detecting APT Activity with NetworkTraffic Analysis
[6] Dell , Lifecycle of an Advanced Persistent Threat
[7] Eric M. Hutchins, et. al., Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains (2010), Lockheed Martin Corporation.
[8] http://en.wikipedia.org/wiki/Data_breach
[9] http://en.wikipedia.org/wiki/Heartbleed
[10] http://heartbleed.com/
[11] http://en.wikipedia.org/wiki/OpenSSL
[12] http://www.nationalcybersecurityinstitute.org/lessons- from-anthem-attack/
[13] 2014 Threat Report: Thrends Beyond the Breach, Mandiant Inc.
[14] What Healthcare Can Learn From CHS Data Breach, www.informationweek.com/healthcare/security-and-privacy/what-healthcare-can-learn-from-chs-data-breach/
[15] https://www.trustedsec.com/august-2014/chs- hackedheartbleed-exclusive-trustedsec/
[16] Jim Aldridge, Targeted Intrusion Remediation: Lessons From The Front Lines, BlackHat, 2012
[17] BAECHER, P., KOETTER, M., HOLZ, T., DORNSEIF, M., AND FREILING, F. The Nepenthes Platform: An Efficient Approach To Collect Malware. In Recent Advances in Intrusion Detection (RAID) (2006).
[18] BAILEY, M., OBERHEIDE, J., ANDERSEN, J., MAO, Z., JAHANIAN, F., AND NAZARIO, J. Automated Classification and Analysis of Internet Malware. In Symposium on Recent Advances in Intrusion Detection (RAID) (2007).
[19] BAYER, U., KRUEGEL, C., AND KIRDA, E. TTAnalyze: A Tool for Analyzing Malware. In Annual Conference of the European Institute for Computer Antivirus Research (EICAR) (2006).
[20] BRUSCHI, D., MARTIGNONI, L., AND MONGA, M. Detecting Self-Mutating Malware Using Control Flow Graph Matching. In Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA) (2006).
[21] CHRISTODORESCU, M., JHA, S., SESHIA, S., SONG, D., AND BRYANT, R. Semantics-Aware Malware Detection. In IEEE Symposium on Security and Privacy (2005).
[22] COZZIE, A., STRATTON, F., XUE, H., AND KING, S. Digging For Data Structures . In Symposium on Operating Systems Design and Implementation (OSDI) (2008).
[23] EGELE, M., KRUEGEL, C., KIRDA, E., YIN, H., AND SONG, D. Dynamic Spyware Analysis. In Usenix Annual Technical Conference (2007).
[24] Kangkook Jee, ShadowReplica: Efficient Parallelization of Dynamic Data Flow Tracking
[25] G. Edward Suh, Jaewook Lee, Secure Program Execution via Dynamic Information Flow
[26] C. Phillips and L. Swiler. A graph-based system for network vulnerability analysis. In ACM New Security Paradigms Workshop, pages 71-79, 1998.
[27] O. sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing. Automated generation and analysis of attack graphs. In Proceedings of IEEE Symposium on Security an Privacy, May 2002.
[28] S. Jha, O. Sheyner, and J. Wing. Two formal analyses of attack graphs. In Proceedings of the 15th IEEE Computer Security Foundations Workshop, Nova Scotia, Canada, June 2002.
[29] Angelos Keromytis, Transparent Computing Proposal, Innovation Information Office, DARPA.