# Forensic Analysis of Advanced Persistent Threat Attacks in Cloud Environments

Changwei Liu[1], Anoop Singhal[2], Duminda Wijesekera[1,2]

[1]Department of Computer Science, George Mason University, Fairfax VA 22030 USA
[2]National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg MD 20899 USA
[1]{cliu6,dwijesek}@gmu.edu ,[2]{anoop.singhal}@nist.gov

**Abstract:** Due to the increasing cyber-activities and the use of diverse devices offered on cloud environments, cloud forensic investigations must deal with data in diverse formats and large quantities from different devices. The process of forensic investigation in a cloud environment involves filtering away noisy data and using expert knowledge to make up the missing attack steps from advanced persistent threats (APT) attacks that have a long time span. Under such circumstance, obtaining timely and convincing forensic results is a challenge.

We show how MITRE's ATT&CK framework and Lockheed Martin's cyber kill chain can be used to identify forensically valuable data, aggregate and correlate them to construct attack steps. ATT&CK is a globally accessible knowledge base of adversary tactics and techniques that is based on real-world observations. APT attacks on cloud systems consist of a successful reconnaissance, command and control communication, privilege escalation, lateral movement through the network, exfiltration of confidential information that are also the key phases of a cyber kill chain. In this paper we investigate using cyber kill chains to organize the evidence and construct the attack steps. By using an experimental network, we show how our methodology can be used to identify evidence, aggregate them and feed them to a Prolog-based tool to re-construct attack steps for the purpose of performing forensic analysis.

**Keywords**: Cloud attacks, forensic analysis, ATT&CK, cyber kill chain, attack steps

## 1. Introduction

Cyber forensics is the application of science to the identification, collection, examination, and analysis, of data while preserving the integrity of the information and maintaining a strict chain of custody [21]. Due to a continued increase in cyber-activities and the diversity of devices that use services provided in cloud environments, the scope of post attack forensic investigations, especially the ones involving attacks on cloud environments have expanded in two dimensions: (1) Expansion of the attack surfaces created by cloud devices that may not have undergone rigorous security check; (2) The need to analyze data that comes in diverse formats and quantities from new attackable interfaces. Of special concern is that instances of servers running on virtual machines in the cloud are monitored by hypervisors that lack warnings, procedures and tools for forensic investigations. Because existing forensic techniques have not been designed for cloud environments, it is challenging to use existing tools to perform forensic analysis in a cloud environment, especially for those advanced persistent threat attacks (APTs) that could last for so long(such as a year) that the timestamps of evidence are not the indicators of the same attack. The process of investigating such cloud attacks involves filtering away noisy data and using expert knowledge or experience to speculate the attacker strategy. This situation creates increasing obstacles in their abilities to obtaining pertinent forensic results.

Researchers have proposed methodologies to collect evidence and correlate them for forensic analysis of cloud attacks, which include developing tools to collect data from hypervisors or virtual machines (VM)s [5,6], leveraging graphical framework to reconstruct the attack scenarios [2, 7] etc. However, cited research is based on strong assumptions that the forensic data was manually aggregated and preprocessed to evidence representing pre-attack conditions and post-conditions, and uses forensic investigators' expertise to build an attack step when the corresponding evidence is incomplete or compromised.

Recently, ATT&CK-like frameworks have been used in understanding and fighting cyber-attacks. Lockheed Martin's intrusion kill chain (also called *cyber kill chain*) describes seven phases including reconnaissance, weaponization, delivery, exploitation, installation, command and control, actions on objectives. Because most APT attacks consist of a successful reconnaissance, command and control communication, privilege escalation, lateral movement through the network, exfiltration of confidential information etc., the cyber kill chain has been used to analyze security logs, develop attack detection and defense systems, and aggregate evidence for APT attack forensic analysis [1,3]. MITRE ATT&CK is a globally accessible knowledge base of adversary tactics and techniques based on real-world observations [11] to emulate cyber-attacks. It has been used in recent years to create a taxonomy of possible attacks in the enterprise IT environment to allow defenders to understand what attacks are being used in the wild and provide methods to detect attacks including certain APTs. Inspired by these works, we propose to leverage MITRE's Adversarial Tactics and Common Knowledge (ATT&CK) and Lockheed Martin's cyber kill chain to identify the evidence of cloud APT attacks, aggregate it and correlate it to construct the attack steps, as the enhancement to our previous work [2,7] of performing forensics in a cloud environment, which had to depend on the investigator's experience and knowledge to identify evidence and building the attack step when the corresponding evidence is incomplete or compromised. Though existing research uses the two tools to detect cyber-attacks or aggregate/correlate evidence [1,3, 19, 20], to the best of our knowledge, there is no published research work that combines both for attack evidence identification and correlation. That is the main contribution in this paper. We use sample attacks to show how both frameworks can be used to identify forensic data in a cloud environment, covert it to pre-attack and after-attack evidence, and feed it to a logic-based forensic tool to construct attack steps.

The rest of the paper is organized as follows. Section 2 describes the background knowledge and related work. Section 3 shows the experimental attacks and the collected evidence. Section 4 describes how we use the two frameworks to identify, correlate evidence, and implement an existing Prolog based forensic tool to construct attack steps. Finally, we conclude the paper in Section 5.

## 2. Background Knowledge and Related Work

We describe Lockheed Martin's cyber-kill chain model, MITRE ATT&CK framework and related work in this section.

### 2.1 The Cyber-Kill Chain Model

As shown as the bottom part in Figure 1, there are seven attacking phases in Lockheed Martin's kill-chain model, representing the steps composed of a successful network attack. In "reconnaissance" phase, the adversaries identify the targets by researching what targets can meet their attacking objectives, and correspondingly collect information that could enable them to launch attacks; in "weaponization" phase, the adversaries prepare their operations by coupling malware and exploits into a deliverable payload, selecting backdoors/implants and appropriate command and control infrastructures for operations; in "delivery" phase, the adversaries convey the malware to the target to launch the attack operations; in "exploitation" phase, the adversaries trigger exploits to gain access to the targeted victim; in "installation" phase, the adversaries install a persistent backdoor or an implant in the victim to maintain the access for an extended period; in "command and control(C2)" period, the adversaries remotely control the backdoor or implant to open a command channel so that the adversaries could control the victim; in the last phase, "actions on objectives", the adversaries achieve their attacking objectives that include collecting user credentials, escalating their privileges, destroying the system, overwriting or corrupting data or modifying data [8]. According to Milajerdi at el.[1], most APT attacks are accomplished through steps conforming to the cyber kill chain, and

aim to obtain and exfiltrate highly confidential information.

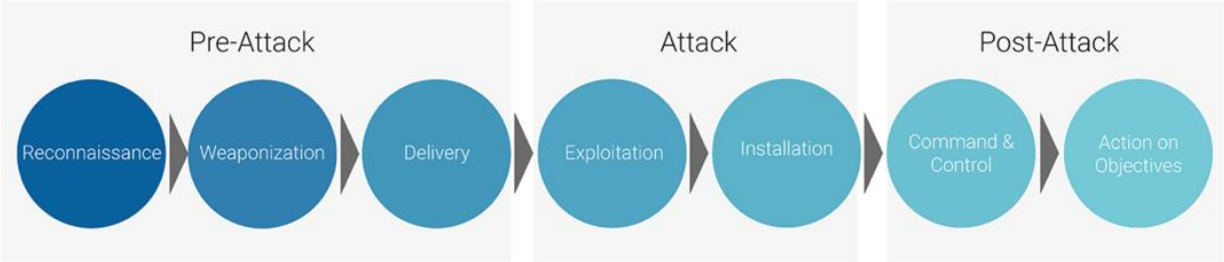



Figure 1. The cyber intrusion kill chain from Lockheed Martin

Cyber Reboot (an American company) reexamined the seven phases and argued that there are three fundamental phases to most network attacks. They are pre-attack, attack and post-attack phases as shown at the top of Figure 1. In the pre-attack phase, the attacker is tasked with attacking objectives and starts to perform reconnaissance of the target; in the attack phase, the attack is executed, enabling the attacker to break through the target's defense and set up communication between the attacked target and the attacker; in the post-attack phase, a further exploitation or access of the targeted victim occurs, which allows the attacker to escalate privilege, destroy the victim system, steal confidential information and etc.

### 2.2 The Adversarial Tactics and Common Knowledge (ATT&CK) Framework

The ATT&CK framework proposed by MITRE [19] is a behavioral model based on real-world observations. As shown in Figure 2, it uses the cyber kill chain to emulate cyber-attacks by emulating the tactics and techniques the attackers could use to achieve their goals. Figure 3 is a snippet of MITRE ATT&CK matrix of enterprise networks, from which we can see that, unlike other threat models that were built by analyzing available threat/vulnerability reports, ATT&CK describes behaviors commonly employed by real adversaries. All attacking techniques are real-world examples of malware or from threats used by red teams. Besides, this framework has public descriptions of attack techniques and how they are leveraged and why the network defenders should pay attention. Therefore, it is useful for network defenders

and forensic investigators to decide what they should monitor and how to specifically investigate to reconstruct the attack steps and mitigate the attack risks.

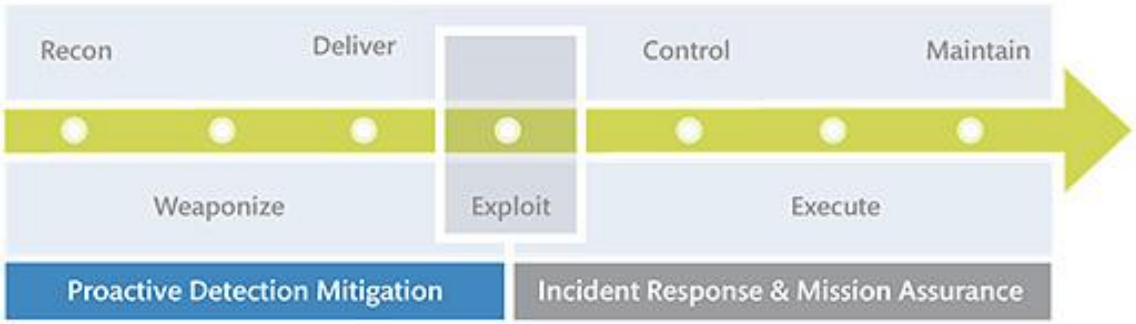



Figure 2. MITRE ATT&CK framework

| Initial Access | Execution | Persistence | Privilege Escalation | Defense Evasion | Credential Access | Discovery | Lateral Movement | Collection | Command and Control | Exfiltration | Impact |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Drive-by Compromise | AppleScript | .bash_profile and .bashrc | Access Token Manipulation | Access Token Manipulation | Account Manipulation | Account Discovery | AppleScript | Audio Capture | Commonly Used Port | Automated Exfiltration | Data Destruction |
| Exploit Public-Facing Application | CMSTP | Accessibility Features | Accessibility Features | Binary Padding | Bash History | Application Window Discovery | Application Deployment Software | Automated Collection | Communication Through Removable Media | Data Compressed | Data Encrypted for Impact |
| External Remote Services | Command-Line Interface | Account Manipulation | AppCert DLLs | BITS Jobs | Brute Force | Browser Bookmark Discovery | Distributed Component Object Model | Clipboard Data | Connection Proxy | Data Encrypted | Defacement |
| Hardware Additions | Compiled HTML File | AppCert DLLs | AppInit DLLs | Bypass User Account Control | Credential Dumping | Domain Trust Discovery | Exploitation of Remote Services | Data from Information Repositories | Custom Command and Control Protocol | Data Transfer Size Limits | Disk Content Wipe |
| Replication Through Removable Media | Control Panel Items | AppInit DLLs | Application Shimming | Clear Command History | Credentials in Files | File and Directory Discovery | Logon Scripts | Data from Local System | Custom Cryptographic Protocol | Exfiltration Over Alternative Protocol | Disk Structure Wipe |

Figure 3. A Snippet of MITRE matrix for enterprise networks

### 2.3 Related Work

Techniques, including remote data acquisition, management plane acquisition, live forensics and snapshot analysis, have been proposed to collect evidence from cloud environments [12]. Dykstra and Sherman retrieved volatile and non-volatile data from the Amazon EC2 cloud active user instance platform using traditional forensic tools such as EnCase and FTK [13]. In order to validate the integrity of the collected data, they subsequently developed the FROST toolkit that can be integrated within OpenStack to collect logs from an operating system that runs the virtual machines [14]. While this technique assumes that the cloud provider is trustworthy, Zawoad et al. resolved this issue by designing a forensics-enabled cloud [15]. Hay and Nance [16] have conducted live digital forensic

analyses on clouds with virtual introspection, a process that enables the hypervisor or any other virtual machine to observe the state of a chosen virtual machine. Dolan-Gavitt et al. bridged the semantic gap between the high-level state information and low-level sources such as physical memory and CPU registers and developed a suite of virtual introspection tools for Xen and KVM [17]. Several hypervisors, including Xen, VMWare, ESX and Hyper-V, support snapshot features that can be used to obtain information about the running state of the virtual machine.

Meanwhile, in order to reduce the time and effort involved in forensic investigations, researchers have proposed automating evidence correlation and attack reconstruction by leveraging rule-based tools and business process diagrams [2]. However, this work depends on forensic experts when the evidence is missing, disjointed or compromised. To execute security investigators in a methodical manner and help to detect real-time APT attacks, intrusion kill-chain has been modified to facilitate data aggregation within a relational database [1, 3].

## 3. Experimental Attacks in a Cloud Environment

In this section, we show an experimental network attacked by both conventional and cloud cyber-attacks to show how the two frameworks of ATT&CK and cyber-kill chain are used to assist cloud forensic analysis.

### 3.1 Our Experimental Cloud and Sample Attacks

Based on the location of an vulnerability and the source of an attacker, the attacks in the cloud can be categorized to two groups [5, 7]: (1) an attacker from the Internet uses conventional network vulnerabilities to attack a virtual machine connected to the Internet; (2) an attacker from a virtual machine uses the vulnerabilities from the shared cloud management resources to launch attacks to other virtual machines on the same hypervisor. The attacks from both kinds of attacks range from Denial of Service, information leakage to privilege escalation, arbitrary code execution etc.

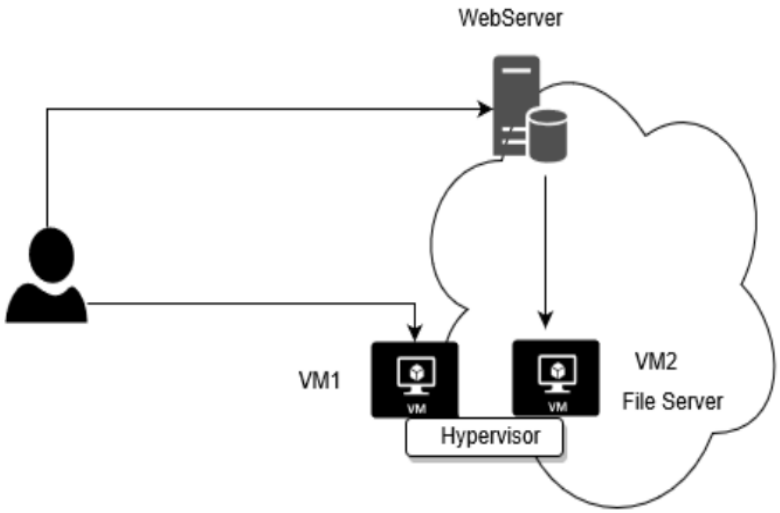



Figure 4. Our experimental cloud and sample attacks toward

Figure 4 illustrates our experimental set up in a lab environment. In this small cloud environment, VM1 and VM2 are two Linux (Ubuntu 14.04) virtual machines that are configured on the same hypervisor (Xen 4.6). We also configured a Windows machine as a webserver, in which a web application can use SQL queries to retrieve the database information stored on VM2--the file server that hosts databases and other files. Toward VM2, we launched two attacks. One is a conventional SQL injection attack exploited by using the vulnerability from the web application that does not sanitize the users' input. The other one is a VM escape attack that could be a kind of APT attacks. The VM escape attack takes advantages of the vulnerability CVE-2017-7228 from VM1, which allows VM1 to control Xen privileged domain, domain 0, and then VM2 so that it can perform local operations, such as deleting a file, in VM2.

### 3.2 Forensic Data Obtained by Using Forensic Tools

To obtain data for forensic analysis, during the attack processes, we logged the accesses toward the webserver, deployed Snort as an IDS to monitor network traffics to the webserver and file server, and installed a VM introspection tool, LibVMI, on Xen Dom0 to capture events and running

processes in the guest VMs(VM1 and VM2). LibVMI is a C library that can be used to monitor the low-level details of a running VM of Xen by viewing its memory, trapping on hardware events, and accessing the vCPU registers. Below are the machine IP addresses and forensic data captured by using the methods/tools mentioned here.

Table 1. The IP address for each machine or VM in Figure 4

| Machine/VM | IP Address |
|---|---|
| Attacker | 129.174.124.122 |
| Web Server | 129.174.125.35 |
| VM1 | 129.174.124.184 |
| VM2(File server) | 129.174.124.137 |

```
[**] SQL Injection Attempt --1=1 [**]
08/08-14:37:27.818279 c:1715 -> 129.174.124.35:8080
TCP TTL:128 TOS:0x0 ID:380 IpLen:20 DgmLen:48 DF
******S* Seq: 0xDEDBEABF  Ack: 0x0  Win: 0xFFFF  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

… …
```

Figure 5. A sample Snort Alert

```
129.174.124.122 - - [08/Aug/2019:14:35:34 -0400] "GET /lab/Test HTTP/1.1" 200 368
129.174.124.122 - - [08/Aug/2019:14:35:39 -0400] "POST /lab/Test HTTP/1.1" 200 981
..
```

Figure 6. Sample access logging from the Web Server

```
130808 14:37:29          …
                40 Query     SET NAMES latin1
                40 Query     SET character_set_results = NULL
                40 Query     SET autocommit=1
                40 Query     SET GLOBAL general_log = 'ON'
                40 Query     select * from profiles where name='Alice' AND password='alice' or
'1'='1'
                40 Quit
```

Figure 7. Sample SQL database logging

…
[ 630] agetty (struct addr:ffff880003c8e200)
[ 669] systemd (struct addr:ffff880076060000)
[ 674] (sd-pam) (struct addr:ffff880076104600)
[ 677] bash (struct addr:ffff880003c8aa00)
[ 703] sudo (struct addr:ffff880004341c00)
[ 704] attack (struct addr:ffff880004343800)

(a) Running processes

test
intel_rapl
x86_pkg_temp_thermal
coretemp
….

(b) Injected Linux modules

Waiting for events...
PID 0 with CR3=77130000 executing on vcpu 1. Previous CR3=788d1000
Waiting for events...
PID 1246 with CR3=788d1000 executing on vcpu 1. Previous CR3=77130000

(c) CPU register values

Figure 8. The Processes, injected Linux modules, and CPU register values of VM2

Table 1 shows the corresponding IP addresses of each machine/VM. According to Table 1, we can see the Snort alert in Figure 5 shows the attacker from 129.174.124.122 attempted to launch a SQL injection attack by using the web application deployed in the webserver with IP address 129.174.125.35 at port number 8080. Figure 6 is the web access history on the webserver, which shows that the attacker machine accessed the web application right before the time when the Snort sent out the alert listed in Figure 5. Figure 7 is the SQL access logging that includes the SQL injection query (the line of "40 Queryselect * from profiles where name='Alice' AND password='alice' or '1'='1'") that resulted in information leakage. Figure 8 are the results obtained by running LibVMI on the attacker VM. These results are forensic data without timestamps. They are composed of the running processes, injected Linux modules and CR3 register values representing the running

processes (the process identifier (PID) is used to find the process name).

## 4. Leveraging the ATT&CK Matrix and Cyber Kill Chain for Forensic Investigation

In this section, we show how we used ATT&CK and the cyber kill chain to assist the investigation process.

### 4.1 Use ATT&CK to Identify the Forensic Data

MITRE ATT&CK includes a knowledge base of 11 tactics and hundreds of techniques an attacker would leverage when compromising an enterprise network. In this framework, while a tactic is a high-level description of a certain type of attack's behaviors, a technique provides a more detailed description of every specific type of behavior within a certain tactic class. The tactics from ATT&CK aren't followed in any linear order as Lockheed's cyber kill chain as shown in Figure 1, and an attacker could bounce between tactics in order to achieve his final goal.

We map forensic data to ATT&CK matrix in order to assist identifying evidence for cloud forensic investigation. As shown in Figure 3, the matrix model's various phases of an attack's lifecycle include "initial access", "execution", "persistence", "privilege escalation", "defense evasion", "credential access", "discovery", "lateral movement", "collection", "command and control", "exfiltration", and "impact". Each of these phases is consisted of different techniques as listed in the matrix table, and the general description is provided below.

- Initial Access consists of techniques that use various entry vectors to gain their initial foothold within a network, which may allow for the attacker's continued access to external remote services.
- Execution consists of techniques resulting in adversary-controlled code running on a local or remote system, which can achieve broader goals such as exploring a network or stealing data by paring with other technologies. Notice that this step does not leave any evidence.

- Persistence consists of techniques that the attacker uses to maintain their foothold on systems even if any interruptions from the system cut off their access.
- Privilege Escalation consists of techniques that an attacker uses to gain higher-level permissions on a system or network. The common approaches include taking advantage of system weaknesses, misconfigurations, and vulnerabilities.
- Defense Evasion consists of techniques such as uninstalling/disabling security software or obfuscating/encrypting data and scripts that an attacker uses to avoid detection throughout their compromise.
- Credential Access consists of techniques that an attack uses to steal credentials to gain access to systems, which provides the opportunity for the attacker to achieve his goals by creating more accounts.
- Discovery consists of techniques an attacker may use to gain knowledge about the system and internal network, which allows the attacker to explore what he can control and discover how the knowledge could benefit his post-compromise information-gathering objective.
- Lateral Movement consists of techniques that an attacker uses to enter and control remote systems on a network. An attacker might install his own remote access tools to accomplish lateral movement or use legitimate credentials with native network and operating system tools.
- Collection consists of techniques an attacker uses to gather information. Frequently, the next goal after collecting data is to steal the data.
- Command and Control consists of techniques that an attacker may use to communicate with systems under his control within a victim network.
- Exfiltration consists of techniques that an attacker may use to steal data from the victim network. The attacker often compresses or encrypt the data to avoid detection, and the channels used to get

data out of a victim network typically include the attacker's command and control channel or an alternate channel with limited size.

- Impact consists of techniques that an attacker uses to disrupt availability or compromise integrity by manipulating business and operational processes, which include destroying or tampering with data.

In our experimental network, we can identify the evidence of the SQL injection attack, because the Snort alert (Figure 5) and SQL query (Figure 7) of this attack clearly show it was a SQL injection attack. For the VM escape attack of using the vulnerability CVE-2017-7228, thought the attack could be observed (the file in VM2 has been deleted), it is hard to construct the attack from the data obtained by using LibVMI (Figure 8), because there was no obvious logging that can identify this attack. Clearly, this is a step that does not leave any evidence. Under this situation, using ATT&CK could narrow down the search scope, helping to find the evidence. According to ATT&CK, the initial access techniques include "drive-by compromise, exploit public facing application, external remote services, hardware additions, replication through removable media, spear phishing attachment, …, trusted relationship". In our experimental cloud, except for the facts that the database in VM2 can be queried by using the web application in the webserver and VM2 shares the same hypervisor (therefore, the hardware) with VM1, it does not have any other connected media, remote services or running applications. Thus, the initial accesses could be narrowed down to the tactics including "exploit public-facing application" from the webserver, and "hardware additions" from the hypervisor. Because the observed attack activities on VM2 include SQL injection alerts and a file deletion, according to ATT&CK, the execution techniques of the attacks could fall into two categories, "exploitation for client execution"(corresponding to application from the webserver) and "command-line interface"(corresponding to the hardware addition). And, the techniques for

"privilege escalation" would be narrowed down to "exploitation for privilege escalation", since the attackers obviously escalated their privileges from the Internet or other VM remotely and other techniques including "access token manipulation", "accessibility features", … etc. are not suitable for the system configuration.

The SQL injection attack has obvious evidence as listed in Figure 5, 6 and 7. However, for the VM escape attack that resulted in the file deletion, the data provided in Figure 8 could only show all running process (including the normal Linux processes and the suspicious user process named as "attack") and injected modules (including normal Linux modules and the suspicious injected user module named as "test"), which does not show the process of exploiting the vulnerability that uses the shared hardware. Therefore, using the potential attack tactics obtained from ATT&CK, the forensic investigators should continue to investigate more forensic data related to exploitations that takes advantage of tactics including "hardware additions", "command-line interface" to launch attacks that allowed the attacker to escalate privileges to the hypervisor level and then delete the file in VM2. According to our previous work [5, 7], system calls constitute good forensic evidence, so a snapshot of VM2 captured during the attacking time was used to retrieve the system calls and kernel messages of the process "attack" and module "test". The system calls we obtained from the process of "attack" are shown in Figure 9, where the arguments following "execve" command in Line 1 clearly shows the attacker from VM1 used command line to execute an attack program named "attack", trying to delete the "samplefile.txt" located at the home folder of VM2(VM2 in named as "victim" in our experiment), and Line 25 clearly shows that the Linux module "test.ko" was injected to the Linux kernel of VM1 to do some work. The kernel activities of VM2 are illustrated in Figure 10, where the kernel messages between Line 1 and Line 6 show that the attacker from VM1 wrote some bytes to the memory after the address "ffff88007c723008", and the messages between Line 8

and Line 19 show that the attacker controlled the page table in Xen to execute the attacker's shellcode by linking the physical memory address where the shellcode is held to the virtual memory address in the page-table. This clearly shows the attacker used the shared memory to launch the attack.

1. execve("./attack", ["./attack", "rm victim ~/samplefile.txt"], [/* 30 vars */]) = 0
2. brk(NULL) = 0x8cd000
3. mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fa3a3022000
4. access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
5. open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

…

25. open("test.ko", O_RDONLY) = 3
26. finit_module(3, "user_shellcmd_addr=1407334317317"..., 0) = 0
27. fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 0), ...}) = 0
28. mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fa3a3021000
29. mmap(0x600000000000, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS|MAP_LOCKED, -1, 0) = 0x600000000000
30. delete_module("test", O_NONBLOCK) = 0
31. exit_group(0) = ?

Figure 9. The system calls obtained by tracing "attack" program

…

1. [ 127.408066] write_byte_hyper(ffff88007c723008, 0x7)
2. [ 127.436071] write_byte_hyper(ffff88007c723009, 0x90)
3. [ 127.460074] write_byte_hyper(ffff88007c72300a, 0xba)
4. [ 127.484055] write_byte_hyper(ffff88007c72300b, 0x26)
5. [ 127.512054] write_byte_hyper(ffff88007c72300c, 0x1)
6. [ 127.548083] write_byte_hyper(ffff88007c72300d, 0x0)
   …
7. [ 127.628071] write_byte_hyper(ffff88007c723010, 0x0)
   …
8. [ 127.660074] going to link PMD into target PUD
9. [ 127.668058] linked PMD into target PUD
10. [ 127.676046] going to unlink mapping via userspace PUD
11. [ 127.684077] mapping unlink done
12. [ 127.692076] copying HV and user shellcode...
13. [ 127.700077] copied HV and user shellcode
14. [ 127.708066] int 0x85 returned 0x7331
15. [ 127.716077] remapping paddr 0x21e8dd000 to vaddr 0xffff880079846800
16. [ 127.724076] IDT entry for 0x80 should be at 0xffff83021e8dd800
17. [ 127.732080] remapped IDT entry for 0x80 to 0xffff804000100800
18. [ 127.740077] IDT entry for 0x80: addr=0xffff82d080229ef0, selector=0xe008, ist=0x0, p=1, dpl=3, s=0, type=15
19. [ 127.748085] int 0x85 returned 0x1337

Figure 10. The kernel message from the injected module

Identifying an attack component is not a trivial task due to the nature of APTs. It requires a lot of detailed analysis such as looking at all processes and process threads that would have altered the state of an object, even under enhanced super-user privileges. As shown in our example, some of these missing steps may need to use system call logs. ATT&CK has techniques and tactics based on real-world observations, which makes it very helpful in identifying the corresponding processes or system calls that are related to a sub-attack phase of an APT attack.

### 4.2 Mapping Log Entries to Attack Steps

Once we identify the evidence by leveraging ATT&CK, we use cyber kill chain to map the corresponding evidence to different attack phases in order to construct the attack steps.

The evidence in Figure 5, Figure 6 and Figure 7 belong to the SQL injection attack, since the timestamps and alerts of these date are consistent. As our analysis in Section 4.1, the data in Figure 5 and 6 show that the attacker from 129.174.124.122 accessed the webserver (129.174.124.35) with SQL injection attempt("—1=1"), which is considered as "initial access" in ATT&CK, that can be easily mapped to the "weaponization" in pre-attack phase. The data in Figure 7 shows, at the same time the database had been queried by using the statement "select * from profiles where name='Alice' AND password='alice' or '1'='1'", which is clearly a SQL injection attack on the database. Since the database itself did not have any security mechanism, this implies the attack succeeded. Therefore, the attack had been delivered, and we can map the data in Figure 7 to "attack" phase in the cyber kill chain.

Likewise, we group the forensic data in Figure 8, Figure 9 and Figure 10 to the same attack by matching the process name "attack" and injected module name "test.ko". Because the data in Figure 8 and 9 only shows the attacker from VM1 ran the process "attack" and module "test" to do

some work that did not show details, we map it to the "weaponization" phase that belongs to pre-attack stage. In addition, as we described in Section 4.1, the data in Figure 9 and 10 shows that the attacker manipulated the shared memory in the same hypervisor to execute shellcode on the victim VM, which can be mapped to "exploitation" phase that belongs to attack stage. Because we observed that the "samplefile.txt" file in the victim VM had been deleted, we knew the attack succeeded, which can be mapped to "action on objectives" in the cyber kill chain that belongs to the post-attack stage.

### 4.3 Co-relate Attack Steps to APTs (Cyber Kill Chains)

Our previous work [18] used a prolog-based tool to generate attack steps by using evidence in the form of Prolog predicates to instantiate rules composed of these predicates representing pre-conditions and post-conditions of an attack. These rules simulate generic attack techniques, which are written in the form of $p :- p_1 , p_2 , p_n,$ where predicate p represents the post-conditions of an attack, and predicates $p_1, p_2 \cdots, p_n$ represent the pre-conditions of the attack. The post-conditions refer to the privileges the attacker obtained after an attack, and the pre-conditions include the attacker's initial privilege, location, system configuration and the vulnerability used to launch the attack.

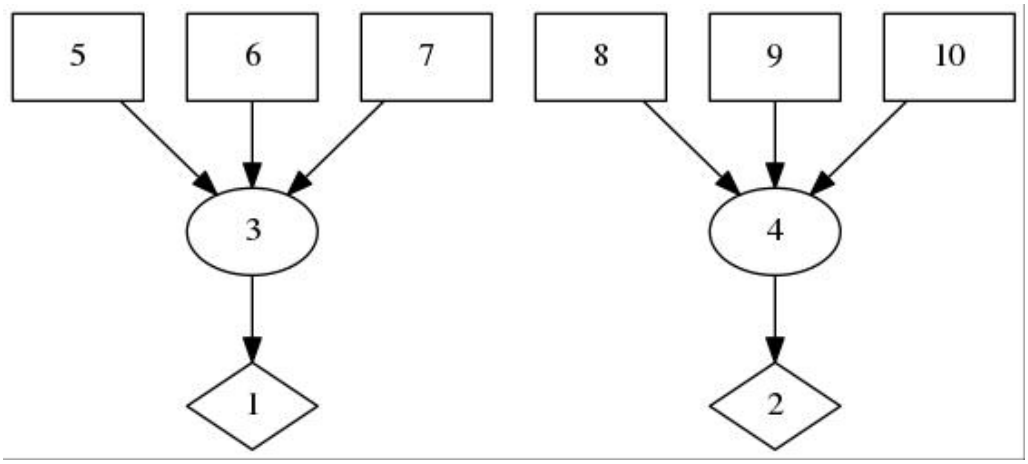
While this prolog-based tool can be used to generate attack steps, it requires users to categorize evidence to post-attack conditions and pre-attack conditions. The tool does not have predicates mapping to the seven cyber kill chain phases. Neither does it have corresponding rules that correlate the evidence corresponding to the seven phases of the cyber kill chain to pre-attack conditions and post-conditions. In this work, we propose to add the missing link by making changes as follows.

(1) We use predicate $Pr_r$, $Pr_w$, $Pr_d$, $A_e$, $A_i$, $Po_c$, $Po_a$ to represent the pre-attack "reconnaissance", "weaponization", "delivery" phases, the attack "exploitation", "installation" phases, and the post-attack "command and control(C2)", "actions on objectives" phases respectively.

(2) We convert techniques in the ATT&CK matrix to corresponding predicates, and map them to the cyber kill chain phases as follows: (a) Predicates of "initial access" are mapped to $Pr_r$, (b) Predicates of "execution", "persistence", "privilege escalation", "defense evasion", "credential access" are mapped to $Pr_w$, (c) Predicates of "discovery" are mapped to $A_i$, (d) Predicates of "lateral movement" are mapped to $A_i$, (e) Predicates of "command and control" are mapped to $Po_c$, and (f) Predicates of "collection", "exfiltration", "impact" are mapped to $Po_a$. Notice that these steps can be modeled as pre-conditions and post-conditions as given in Figure 11.

All predicates are composed of names and variables that depict the system configuration, attacker's privilege, the network topology, software vulnerability and etc. For example, technique "exploit public-facing application" in "initial access" of the ATT&CK matrix is written to Predicate "$Pr_r$ (attackerAccess(_host, _program))", and technique "account manipulation" in "credential access" of the ATT&CK matrix is written to Predicate "$Pr_w$(hasAccount(_principal, _host, _account))", where the variables (such as _host, _program …) following the predicate names(publicApp and hasAccount) in the predicates will be instantiated by concrete information during the run time of using this Prolog tool.

(3) We add rules to use cyber kill chain to correlate the corresponding predicates in different phases to an attack step. The rules are in the in the form of $Po_c$ *:-* $(Pr_r ; Pr_w ; Pr_d)$ , $(A_e ; A_i)$. *and* $Po_a$ *:-* $(Pr_r ; Pr_w ; Pr_d)$ , $(A_e ; A_i)$, where ";" represents logical OR, and "," presents logical AND. These rules mean that, if there is evidence in pre-attack, attack and attack phases, an attack step is constructed.




| | | | |
|---|---|---|---|
| 1 | ExecCode(VM2,read) | 6 | networkServiceInfo(dbServer, httpd, tcp, 3660, user) |
| 2 | ExecCode(VM2,modify) | 7 | vulExists(webServer, 'CWE89', httpd) |
| 3 | THROUGH 3 (remote exploit of a server program) | 8 | hasAccount(attacker,VM1,root) |
| 4 | Through 8 (Access a host through executing code on the machine) | 9 | vulExists(sharedMemory,'CVE-2017-7228', hardware) |
| 5 | attackerAccess(publicWebApp) | 10 | vulProperty('CVE-2017-7228', remoteExploit, privEscalation) |

Figure 11. The constructed attack steps of the experimental network

With these changes in our Prolog based tool, we constructed our attack steps as shown in Figure 11, which shows intuitive graphical attack steps for the two attacks constructed by using cyber kill chain to correlate evidence found in the attacked system. The left path shows that the attacker used a publicly available web application to launch SQL injection to the database in VM2, and the right path shows the attacker used the vulnerability in the shared hardware to attack VM2 and deleted a file in VM2.

We notice that our miniature example provides an initial example of the ATT&CK rule model. Although some rules (for example lateral movement by the attacker or pass the hash attacks) are

missing. We postulate that these missing rules can be learned by using machine learning algorithms and incorporated into the steps of the ATT&CK process.

## 5. Conclusion

Justifying pre-attack, attack, and attack phases, requires having evidence of activities related to these phases. In APT attack analysis, there are many difficulties in constructing attack steps because (1) they do not lend themselves to traditional precondition, post-condition analysis (2) recognizing the pre-attack post-attack phases may require statistical correlation techniques that uses data from multiple sources. Thus, creating valid arguments for such attacks becomes more challenging. In particular assigning a "timestamp" for such an attack in a cloud environment becomes a challenge. We showed that we can leverage ATT&CK to identify the evidence and build the attack steps by mapping the evidence to different phases in the cyber kill chain. We used a case study to validate our framework. We plan to extend the relationship between the kill chain, evidence gathering and attack-attribution in the future work.

## Disclaimer

This paper is not subject to copyright in the United States. Commercial products are identified in order to adequately specify certain procedures. In no case does such an identification imply a recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the identified products are necessarily the best available for the purpose.

## References:


1. S. M Milajerdi, R. Gjomemo, B. Eshete, R. Sekar and V.N. Venkatakrishnan, "HOLMES: real-time APT detection through correlation of suspicious information flows," *arXiv preprint arXiv:1810.01594,* 2018.
2. C. Liu, A. Singhal, D. Wijesekera, "A Layered Graphical Model for Cloud Forensic Mission Attack Impact Analysis", IFIP International Conference Digital Forensics, New Delhi, India, Jan 2018.
3. B.D. Bryant and H. Saiedian, "A novel kill-chain framework for remote security log analysis with SIEM software", *computers & security, 67,* pp.198-210, 2017.

4. K. Ruan, J. Carthy, T. Kechadi,and M. Crosbie, "Cloud forensics", In IFIP International Conference on Digital Forensics, pp. 35-46. Springer Berlin Heidelberg, 2011.
5. C. Liu, A. Singhal and D. Wijesekera, "Identifying evidence for cloud forensic analysis", In IFIP International Conference on Digital Forensics (pp. 111-130). Springer, Cham, 2017.
6. http://libvmi.com/.
7. C. Liu, A. Singhal, R. Chandramouli and D. Wijesekera, "Determining forensic data requirements for detecting hypervisor attacks", International Conference on Digital Forensics, Orlando, U.S.A, Jan 2019.
8. Lockheed Martin, "Gaining the advantage--Applying Cyber Kill Chain® Methodology to Network Defense", retrieved from https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/Gaining_the_Advantage_Cyber_Kill_Chain.pdf.
9. C. Liu, A. Singhal, D. Wijesekera, "Relating admissibility standards for digital evidence to attack scenario reconstruction", Journal of Digital Forensics, Security and Law, 9(2) 15, 2014.
10. P. Gary (2001), 'A Road Map for Digital Forensic Research', Technical Report DTR-T001-01, DFRWS, Report from the First Digital Forensic Research Workshop (DFRWS).
11. MITRE ATT&CK™, retrieved from https://attack.mitre.org.
12. A. Pichan, M. Lazarescu and S. Soh, Cloud forensics: Technical challenges, solutions and comparative analysis, Digital Investigation, vol. 13, pp. 38–57, 2015.
13. J. Dykstra and A. Sherman, Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust and techniques, Digital Investigation, vol. 9(S), pp. S90–S98, 2012.
14. J. Dykstra and A. Sherman, Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform, Digital Investigation, vol. 10(S), pp. S87–S95, 2013.
15. S. Zawoad and R. Hasan, A trustworthy cloud forensics environment, in Advances in Digital Forensics XI, G. Peterson and S. Shenoi (Eds.), Springer, Heidelberg, Germany, pp. 271–285, 2015.
16. B. Hay and K. Nance, Forensic examination of volatile system data using virtual introspection, ACM SIGOPS Operating Systems Review, vol. 42(3), pp. 74–82, 2008.
17. B. Dolan-Gavitt, B. Payne, W. Lee, "Leveraging forensic tools for virtual machine introspection", (Georgia Institute of Technology, Atlanta, GA), Technical Report, 2011.
18. C. Liu, A. Singhal and D. Wijesekera, "A probabilistic network forensic model for evidence analysis", In IFIP International Conference on Digital Forensics (pp. 189-210), Jan, 2016.
19. B. Strom, J. Battaglia, M. Kemmerer, W. Kupersanin, D. Miller, C. Wampler, S. Whitley and R. Wolf, "Finding cyber threats with ATT&CK-based analytics", Technical Report MTR170202, MITRE, 2017.
20. A. D'Amico and K. Whitley, "The real work of computer network defense analysts", In VizSEC 2007 (pp. 19-37).
21. K. Kent, S. Chevalier, T. Grance, "NIST SP 800-86 on Digital Forensics".