GRAND CANYON UNIVERSITY™

Activity 1 Guide: The ABCs of Information Security

Table of Contents

Part 1 – Data Breach Wall of Shame

Real Company Experiences with Data Breach Problems

Read the summary and related news articles about companies that were infamous for being victims of data problems. We will apply these stories to a list of data protection concepts throughout this lesson.

1. Yahoo Data Breach

- **Date**: 2013–2014 (disclosed in 2016)
- **Background**: Yahoo suffered one of the largest data breaches in history, with over 3 billion user accounts compromised. The breach occurred over two years and was discovered and disclosed only in 2016. Hackers used a method called "forged cookies" to bypass Yahoo's security. These forged cookies allowed them to access user accounts without needing a password. Additionally, hackers exploited vulnerabilities in Yahoo's outdated security infrastructure and took advantage of weak security questions to gain access to accounts. This massive breach included the exposure of personal information, such as names, email addresses, dates of birth, hashed passwords, and security questions and answers, often used for account recovery. The delay in discovering and disclosing the breach highlighted significant gaps in Yahoo's security monitoring and incident response processes.
- **Outcome**: The breach exposed names, email addresses, dates of birth, and security questions and answers. This significantly impacted Yahoo's reputation and led to a $350 million reduction in its acquisition price by Verizon. Yahoo faced multiple lawsuits and regulatory investigations, resulting in further financial and reputational damage. The incident underscored the importance of robust security measures, timely breach detection, and transparent communication with affected users.
- **CNN Business – "Every Sing Yahoo Account Was Hacked – 3 Billion in All," by Larson (2017)**
- **Reuters – "Yahoo Says All 3 Billion Accounts Hacked in 2013 Data Theft," by Stempel and Finkle (2017)**
- **Dark Reading – "10 Years After Yahoo Breach, What's Changed? (Not Much)," by Nelson (2024)**

2. Equifax Data Breach

- **Date**: 2017
- **Background**: Equifax, a major credit reporting agency, experienced a data breach that exposed the personal financial information of 147 million people. The breach was caused by a vulnerability in a web application framework (Apache Struts) that Equifax failed to patch, despite being aware of the vulnerability and its available fix for months. Hackers exploited this vulnerability to gain access to Equifax's systems and extract sensitive data, including Social Security numbers, birth dates, addresses, driver's license numbers, and credit card numbers. The breach also exposed dispute documents with personal

identifying information. Equifax's delayed response and inadequate security measures, such as the lack of encryption for sensitive data, were heavily criticized. Internal investigations revealed that the breach could have been prevented with proper patch management and more stringent security protocols.

- **Outcome**: The breach led to the theft of Social Security numbers, birth dates, addresses, and more. Personal, unchangeable personal data that is stolen can result in fraud and identity theft years after the incident. Equifax faced significant legal and financial repercussions, including a $700 million settlement with the Federal Trade Commission (FTC), the Consumer Financial Protection Bureau (CFPB), and 50 U.S. states and territories. The settlement included funds for affected consumers, credit monitoring services, and penalties. Equifax's CEO and other top executives resigned, and the company committed to enhancing its cybersecurity measures and monitoring.
- **CSO – "Equifax Data Breach FAQ: What Happened, Who Was Affected, What Was the Impact?" by Fruhlinger (2020)**
- **FTC – "Equifax Data Breach Settlement" (2024)**
- **CNN – "Giant Equifax Data Breach: 143 Million People Could Be Affected," by O'Brien (2017)**

3. Target Data Breach

- **Date**: 2013
- **Background**: Target experienced a data breach during the holiday shopping season, affecting 40 million credit and debit card accounts. Hackers gained access to Target's network using credentials stolen from a third-party vendor that provided HVAC (heating, ventilation, air conditioning) services to Target. Once inside the network, the attackers installed malware on the point-of-sale (cash register and credit card reader) systems, which allowed them to capture customer payment card data, including card numbers, expiration dates, and CVV codes. The breach also exposed the personal information of 70 million customers, including names, addresses, phone numbers, and email addresses. Stolen credit card information was sent back to the hacker's servers in small quantities to avoid detection. Target's security systems detected the malware, but the alerts were not acted upon in time to prevent the data exfiltration. The breach highlighted vulnerabilities in third-party vendor management and seemingly unrelated computer systems.
- **Outcome**: The breach resulted in the theft of customer payment card data and personal information. Target faced numerous lawsuits from banks, consumers, and shareholders, leading to over $200 million in legal fees and settlements. The company also invested heavily in upgrading its cybersecurity infrastructure and enhancing its monitoring and response capabilities. The incident caused significant reputational damage and led to a temporary decline in customer trust and sales.
- **US Senate – "A 'Kill Chain' Analysis of the 2013 Target Data Breach," by the Committee on Commerce, Science, and Transportation (2014)**
- **Reuters – "Target Missed Many Warning Signs Leading to Breach: U.S. Senate Report," by Chiacu (2014)**
- **Card Connect – "Case Study: What We've Learned from the Target Data Breach of 2013" (2023)**

4. NASA's Mars Climate Orbiter

- **Date**: 1999
- **Background**: The Mars Climate Orbiter, a NASA space probe, was intended to study Mars' atmosphere and climate. The mission failed due to a data corruption issue stemming from a unit conversion error. One engineering team used metric units (newton-seconds) for thrust data, while another used imperial units (pound-seconds). This mismatch led to incorrect navigation commands being sent to the spacecraft. The probe's trajectory was miscalculated, causing it to enter the Martian atmosphere at an improper angle and ultimately disintegrate. The error went unnoticed due to inadequate checks and balances in the project's software development and testing processes. The investigation revealed that there were multiple missed opportunities to catch the error during the mission planning and execution phases.
- **Outcome**: The loss of the mission cost NASA approximately $125 million. The incident prompted NASA to implement more rigorous review and verification processes for unit conversions and data handling. The failure highlighted the importance of clear communication and consistent use of measurement units across engineering teams, as well as the need for thorough testing and validation procedures in complex projects.
- **NASA – System Failure Case Studies – Lost in Translation**
- **Think Reliability – "Root Cause Analysis – The Loss of the Mars Climate Orbiter"**

5. Knight Capital Group Trading Glitch

- **Date**: 2012
- **Background**: Knight Capital Group, a major financial services firm, experienced a significant trading glitch due to corrupted data in its software deployment. A faulty software update caused the firm's automated trading system to send erroneous orders. Specifically, the update inadvertently activated an obsolete software module that had been dormant in the system for years. This module generated a series of unintended trades, causing the system to malfunction. The update was not thoroughly tested before deployment, and there were inadequate safeguards to detect and halt the erroneous trades. The glitch led to massive disruptions in the stock market, affecting the prices of numerous stocks and causing a spike in market volatility.
- **Outcome**: The glitch resulted in a loss of $440 million in just 45 minutes and ultimately forced the company to seek financial rescue, significantly damaging its reputation. Knight Capital had to secure a $400 million financing deal to stay afloat. The incident underscored the critical importance of thorough testing, change management, and the implementation of robust safeguards in automated trading systems.
- **CIO – "Software Testing Lessons Learned From Knight Capital Fiasco," by Heusser (2012)**
- **Henrico Dolfing – "Case Study 4: The $440 Million Software Error at Knight Capital" (2019)**

6. Colonial Pipeline Ransomware Attack

- **Date**: 2021
- **Background**: Colonial Pipeline, a major fuel pipeline operator in the U.S., was hit by a ransomware attack, causing a temporary shutdown. The DarkSide ransomware group gained access to Colonial's systems by exploiting a leaked password for an inactive VPN account. The attackers used the stolen credentials to infiltrate the network and deploy ransomware, encrypting data and demanding a ransom for its release. The shutdown of the pipeline, which supplies nearly half of the East Coast's fuel, led to widespread fuel shortages and panic buying.
- **Outcome**: The shutdown led to fuel shortages and price spikes. Colonial Pipeline paid a ransom of $4.4 million, although part of it was later recovered by the FBI. The incident highlighted the vulnerabilities of critical infrastructure to cyberattacks and prompted increased government and industry focus on improving cybersecurity measures and resilience against such threats.
- **Tech Target – "Colonial Pipeline Hack Explained: Everything You Need to Know," by Kerner (2022)**
- **Georgetown – "Cybersecurity Policy Responses to the Colonial Pipeline Ransomware Attack," by Wood (2023)**

7. SolarWinds Supply Chain Attack

- **Date**: 2020
- **Background**: SolarWinds software monitors networks and server actions. The SolarWinds cyberattack involved the insertion of malware into the company's software update, affecting numerous government and corporate networks. Hackers, believed to be state-sponsored, compromised SolarWinds' Orion software build system, embedding malicious code in the updates distributed to customers. Since SolarWinds software was signed and trusted by internal systems, the attack was not immediately detected. This allowed the hackers to infiltrate networks and extract sensitive data. The attack was particularly sophisticated, as it leveraged the trust organizations placed in SolarWinds' software updates. The malware, dubbed "Sunburst," created backdoors in affected systems, enabling persistent access and data exfiltration without detection for months.
- **Outcome**: The attack compromised sensitive data across various sectors, including U.S. government agencies and Fortune 500 companies. It highlighted vulnerabilities in software supply chains and led to increased scrutiny and security measures. The incident prompted calls for stronger supply chain security protocols and collaboration between the public and private sectors to address emerging cyber threats.
- **Government Accountability Office – "SolarWinds Cyberattack Demands Significant Federal and Private-Sector Response (Infographic)" (2021)**
- **Solar Winds – "FAQ: Security Advisory" (2021)**

8. AT&T Data Breach

- **Date**: March 2024 (discovered)
- **Background**: Hackers breached AT&T's systems, stealing personal data of 7.6 million current and 65.4 million former customers, including sensitive information like social security numbers, account numbers, and passcodes. The breach appeared to involve data from 2019 or earlier and was discovered when it surfaced on the dark web. Hackers used phishing and social engineering techniques to gain initial access and then exploited vulnerabilities in AT&T's systems. The breach followed a previous leak in January 2023 that affected nine million users, raising concerns about AT&T's ability to secure customer data.
- **Outcome**: AT&T launched an investigation and faced multiple class action lawsuits. The breach highlighted the ongoing challenges of securing vast amounts of customer data and the need for continuous improvements in security practices. The company implemented measures to enhance its security infrastructure and prevent future breaches.
- **USA Today – "How to Know If You Were Affected by the AT&T Data Breach and What to Do Next," by Hauari (2024)**
- **Mozilla Foundation – "AT&T Had a Huge Data Breach: Here's What You Need to Know," by Caltrider and MacDonald (2024)**

9. Ticketmaster Data Breach

- **Date**: May 2024
- **Background**: Over 560 million customer records, including order history, payment information, names, addresses, and email data, were leaked online and offered for sale by hackers. The breach involved SQL injection attacks on Ticketmaster's databases, which allowed hackers to extract and sell sensitive customer information. The incident was part of a larger wave of attacks targeting entertainment and ticketing companies. SQL injection is a common technique where attackers insert malicious SQL code into web forms to manipulate and access the underlying database.
- **Outcome**: Ticketmaster advised customers to monitor their accounts for fraudulent activity. The Justice Department prepared a federal antitrust lawsuit against Live Nation, Ticketmaster's parent company, exacerbating the company's legal troubles. The breach prompted Ticketmaster to review and enhance its database security measures to prevent similar attacks in the future.
- **Ticketmaster – "Ticketmaster Data Security Incident" (2024)**
- **Security Magazine – "Security Leaders Respond to Ticketmaster Breach," by Alger (2024)**

10. CrowdStrike Global Technology Outage

- **Date:** July 19, 2024
- **Background:** A global technology outage was caused by a software update from the cybersecurity firm CrowdStrike. Although not an attack, the error caused service disruptions for airlines, health care systems, banks, and numerous other businesses and services around the world. Like the SolarWinds software, CrowdStrike is tightly integrated in the operating system of servers. The software update resulted in crashes of

machines running via the Windows operating system as they were booting up. The software needed to be removed manually, causing expensive downtime for customers.

- **Outcome** Multiple U.S. airlines grounded all flights for several days; Delta airlines did not recover normal operations for more than a week. Airports worldwide saw flight cancelations when ticketing and boarding systems were down. The United Parcel Service and FedEx reported disruptions. TD Bank customers faced issues accessing online accounts, and several state and municipal court systems closed for the day. The outage caused hospitals to cancel noncritical surgeries. Emergency response systems in the United States were also affected, with 911 lines down in multiple states.
- **The Verge – "CrowdStrike and Microsoft: All the Latest News on the Global IT Outage," by Sato (2024)**
- **Tech Crunch – "What We Know About CrowdStrike's Update Fail That's Causing Global Outages and Travel Chaos," by Whittaker and Franceschi-Bicchierai (2024)**

Part 2 – CIA

Confidentiality, Integrity, and Availability: The CIA of Information Security

The CIA triad—confidentiality, integrity, and availability—seen in Figure 1 is a way to remember three critical goals in information security. Decisions related to design, policies, and security controls usually can be categorized in one or more of these areas.

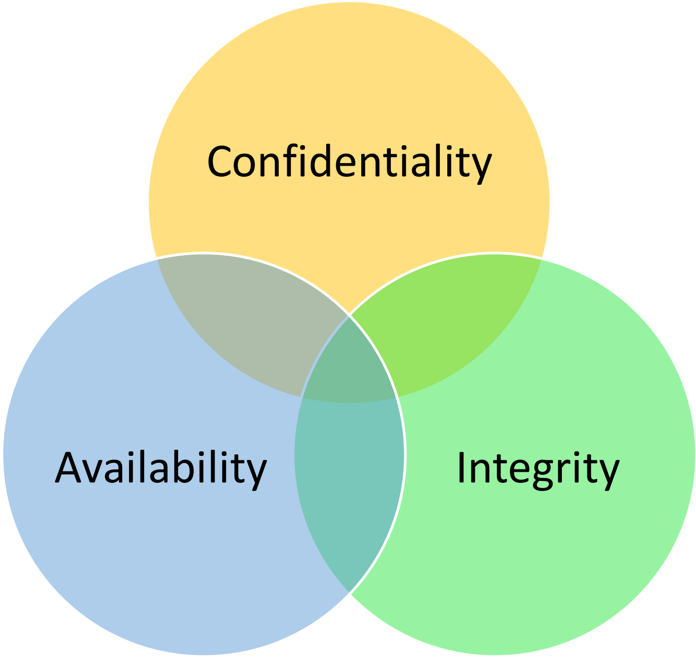

Figure 1 The C.I.A. describes the goals of information security.

Confidentiality

Confidentiality refers to protecting information from unauthorized access. Confidentiality means that only authorized individuals or systems can access sensitive data. For example, when you use online banking, confidentiality measures like encryption and strong passwords protect your financial information from other users or hackers. In a school setting, confidentiality ensures that students' personal records are only accessible to authorized personnel.

Integrity

Integrity means data are trustworthy, complete, and have not been accidentally altered or modified by an unauthorized user. It ensures that information remains accurate and reliable. For instance, if a student submits an assignment online, integrity measures ensure that the submitted file remains unchanged until it is graded by the teacher. Checksums and digital signatures are examples of techniques used to maintain data integrity.

Availability

Availability means data are accessible when you need them. It ensures that information and resources are available to authorized users whenever they require access. For example, if a school's online learning platform is down during exam week, it disrupts students' ability to study and submit assignments. System downtime is frustrating and expensive. Availability measures, such as regular backups and redundant systems, help prevent such disruptions and ensure continuous access to important information.

Checkpoint #1

1. Read the summary of real-world incidents provided at the beginning of this lesson.
2. Research news reports published near the time of the incident.
3. Identify which of the three aspects of C.I.A. is most closely related to the problem described.
4. Explain in at least one sentence the reason for your choice.

Problem	**C.I.A. Issue(s) Involved**	**Reason(s) for Your Decision**	**Source Used to Make Your Decision (APA Citation)**
1. Yahoo			
2. Equifax			
3. Target			
4. NASA			
5. Knight			

6. Colonial Pipeline			
7. SolarWinds			
8. AT&T			
9. Ticketmaster			
10. CrowdStrike			

Part 3 – MOM: Motive, Opportunity, and Method

MOM is an acronym in the context of cybersecurity and criminal investigations that stands for motive, opportunity, and method, seen in Figure 2. This framework helps in understanding and analyzing cyber threats and criminal activities.

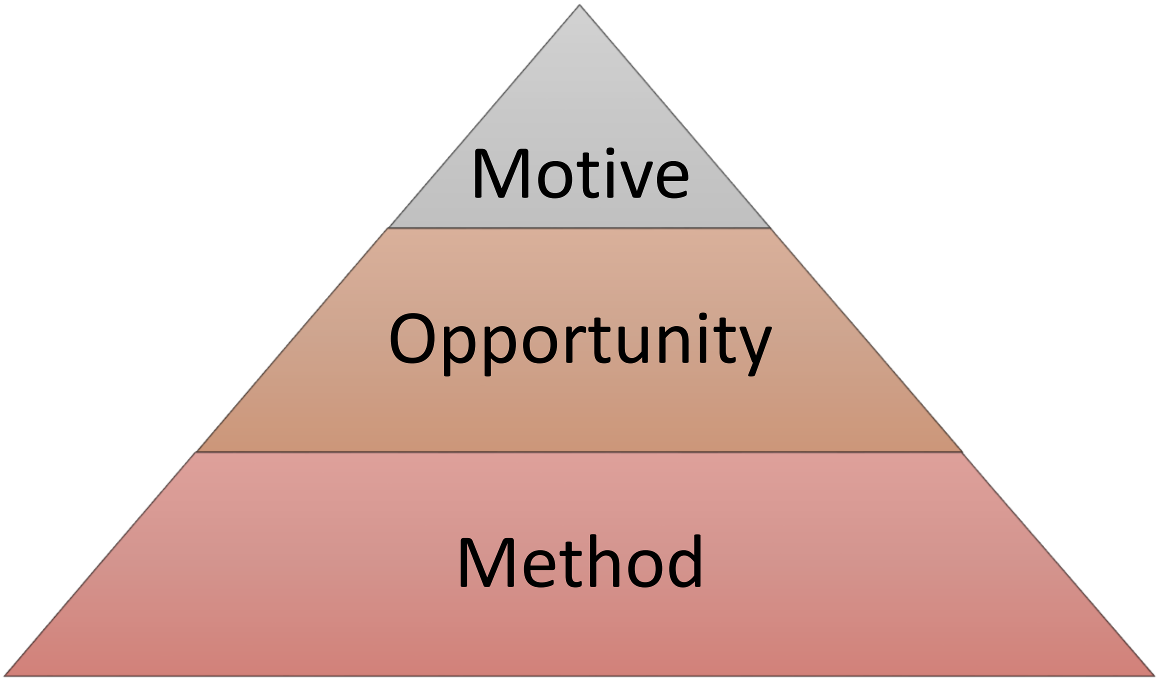

Figure 2 MOM defines the profile of bad actors in information security.

Motive

Motive refers to the reason why an individual or group commits a cybercrime or engages in malicious activity. Understanding the motive behind an attack can provide insights into the potential targets, the level of threat, and the resources the attacker might be willing to invest. Common motives include:

- **Financial Gain**: Cybercriminals often seek monetary benefits, such as stealing credit card information, committing fraud, or demanding ransoms through ransomware attacks.
- **Espionage**: State-sponsored attackers may seek to gather intelligence or steal sensitive information for political, economic, or military advantage.
- **Hacktivism**: Activists or hacktivists may conduct cyberattacks to promote their political or social agendas, often targeting organizations they view as adversaries.
- **Revenge**: Disgruntled employees or individuals with personal grievances may launch attacks to harm their employers or others they hold responsible for their grievances.
- **Challenge or Recognition**: Some attackers, particularly within the hacker community, may be motivated by the challenge of breaking into secure systems or gaining recognition for their skills.

Method

Method refers to the techniques, tools, and procedures used by attackers to carry out their activities. Understanding the method helps in identifying patterns, vulnerabilities, and the likely impact of an attack. Common methods include:

- **Phishing**: Deceptive emails or messages designed to trick recipients into revealing sensitive information or installing malware.
- **Malware**: Malicious software, such as viruses, worms, ransomware, and spyware, used to disrupt, damage, or gain unauthorized access to systems.
- **Exploitation of Vulnerabilities**: Vulnerabilities (security weaknesses) may be present for extended periods of time, but when a hacker discovers and takes advantage of them, the action (exploit) becomes an attack.
- **Social Engineering**: Manipulating individuals into divulging confidential information or performing actions that compromise security.
- **Denial of Service (DoS)**: Overloading a system or network to make it unavailable to users.

Opportunity

Opportunity refers to the circumstances or conditions that make it possible for an attacker to carry out their activities. This involves identifying vulnerabilities, security gaps, or situations that an attacker can exploit. Factors contributing to opportunity include:

- **Weak Security Measures**: Inadequate or outdated security controls, such as weak passwords, lack of encryption, or unpatched systems.
- **Human Error**: Mistakes or negligence by employees, such as falling for phishing attacks, misconfiguring systems, sloppiness, or failing to follow security protocols. Human error can result in self-inflicted outages without attacks from threatening actors.
- **Access to Resources**: Availability of tools, information, or access that an attacker can use to execute their plan. Many resources are unnecessarily public (configuration, version

numbers, error messages, company structure, current activities) and can be used to exploit a system.

- **Insider Threats**: Employees or contractors with legitimate access to systems who may intentionally or unintentionally aid in an attack.

Real Company Experiences with Data Information Problems

Checkpoint #2

1. Refer to the 10 company data problems mentioned earlier.
2. Research news reports published near the time of the incident.
3. Choose one of the 10 incidents where one of the MOM (motive, opportunity, or method) factors is most prominent in the story.
4. Explain in at least one sentence the reason for your choice.

Factor Most Obvious in the Story	Company Involved	Reason(s) for Your Decision	Source Used to Make Your Decision (APA Citation)
Motive			
Opportunity			
Method			

Part 4 – Cybersecurity Response Types

Cybersecurity involves a range of response types to address threats and vulnerabilities. These responses can be categorized into several types: prevention, detection, mitigation, recovery, deterrence, as seen in Figure 3.

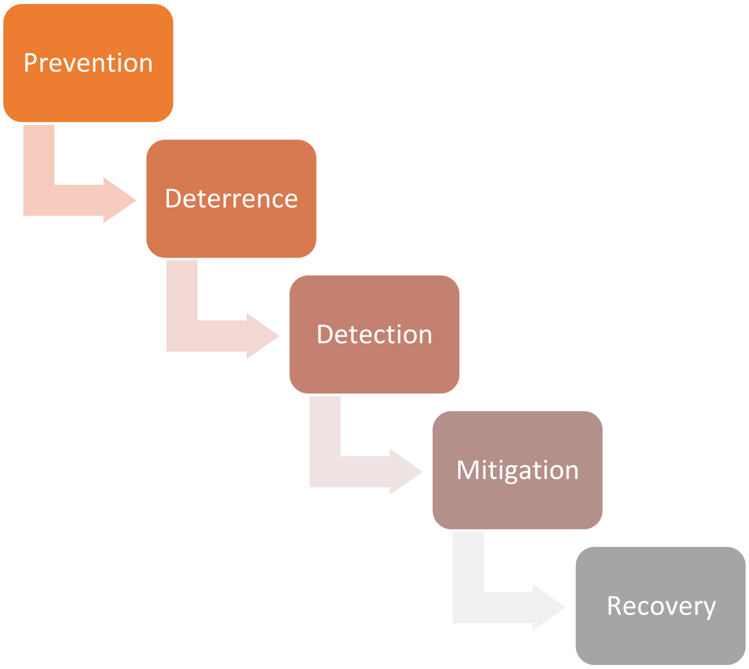

Figure 3 Response types for information security problems.

Here is a summary of each response type, along with additional related defensive measures:

Prevention

Prevention focuses on measures taken to stop cyber threats before they occur. It involves implementing controls and practices designed to minimize vulnerabilities and deter attackers. The value of investments in prevention are difficult to measure until the considerable costs of downtime and recovery are considered. Key prevention strategies include:

- **Access Control**: Restricting access to systems and data to authorized users only, using methods like multi-factor authentication (MFA) and role-based access control (RBAC).
- **Security Training**: Educating employees about cybersecurity best practices and how to recognize potential threats like phishing attacks.
- **Patch Management**: Regularly updating software and systems to fix security vulnerabilities and protect against exploits.
- **Firewalls and Antivirus Software**: Using tools to block unauthorized access and detect and prevent malware infections.
- **Reducing Attack Surface**: Implementing measures to minimize the number of potential entry points for attackers, such as disabling unnecessary services and securing open ports.

Deterrence

Deterrence involves measures aimed at discouraging attackers from attempting to breach security defenses by making the potential cost or difficulty of an attack higher than the perceived benefits. Key deterrence strategies include:

- **Legal and Regulatory Measures**: Enforcing laws and regulations that penalize cybercriminal activities, thereby deterring potential attackers through the threat of legal consequences.
- **Strong Security Postures**: Use of standard and effective security measures discourage attackers who may seek easier targets.
- **Security Policies and Procedures**: Establishing clear policies and procedures that outline severe penalties for insider threats and security breaches.
- **Removal of Motive**: Some solutions allow businesses to remove the temptation for attack. For example, a company may choose to collect and store minimal amounts of data about their clients. Google sign-on, for example, removes passwords from a company's database.

Detection

Detection involves identifying and monitoring suspicious activities and potential security incidents. Effective detection mechanisms enable quick identification of threats, allowing for prompt response. Key detection methods include:

- **Intrusion Detection Systems (IDS)**: Monitoring network traffic for signs of malicious activity and generating alerts when potential threats are detected.
- **Security Information and Event Management (SIEM)**: Collecting and analyzing data from various sources to identify patterns and anomalies indicative of security incidents.
- **Log Monitoring**: Regularly reviewing system and application logs to detect unusual behavior or unauthorized access attempts.
- **Network Monitoring**: Continuously observing network traffic to identify potential intrusions or abnormal patterns.
- **AI-Driven Cybersecurity** Many detection products incorporate artificial intelligence to sort through large sets of data and identify subtle patterns that may indicate a threat. Automated threat detection and response capabilities enable organizations to respond to incidents more quickly, minimizing potential damage. AI can identify potential threats before they become actual attacks, allowing security teams to take preemptive action. AI products often are trained on the normal behavior of every user, device, and network within an organization and respond to unusual changes in behavior.

Mitigation

Mitigation involves taking steps to limit the impact of a detected threat or ongoing attack. This includes actions taken to contain the threat and prevent further damage. Key mitigation strategies include:

- **Incident Response Plans**: Predefined procedures for responding to security incidents, including roles and responsibilities, communication plans, and action steps.
- **Containment Strategies**: Isolating affected systems to prevent the spread of malware or unauthorized access, such as disconnecting compromised machines from the network.
- **Threat Neutralization**: Removing or disabling the threat, such as deleting malware, terminating unauthorized sessions, or blocking malicious IP addresses.

Recovery

Recovery focuses on restoring normal operations and minimizing the long-term impact of a security incident. This includes restoring data, repairing affected systems, and improving security to prevent future incidents. Key recovery actions include:

- **Data Restoration**: Recovering lost or corrupted data from backups to restore normal business operations.
- **System Repair and Rebuilding**: Fixing or reinstalling affected systems to remove any traces of the attack and ensure they are secure.
- **Business Continuity Planning (BCP)**: Ensuring that critical business functions can continue during and after a security incident. This involves creating and testing plans for maintaining operations in the face of disruptions.
- **Disaster Recovery Planning (DRP)**: Focusing on the technical aspects of recovering IT infrastructure and data after a significant disruption or disaster.
- **Post-Incident Analysis**: Investigating the incident to understand how it occurred, what vulnerabilities were exploited, and what can be done to prevent similar incidents in the future.
- **Improving Security Posture**: Updating security policies, procedures, and controls based on lessons learned from the incident.
- **Communication and Reporting**: Effectively communicating with stakeholders, including employees, customers, regulators, and the media, during and after an incident. This includes timely reporting of breaches to comply with legal and regulatory requirements.
- **Recovery Time Objective (RTO)**: Defines the maximum acceptable amount of time that a system or application can be down after an incident before it significantly impacts the business. Some systems cannot afford to lose a second's worth of operations while others may tolerate hours or days of outage without significantly affecting an organization.
- **Recovery Point Objective (RPO)**: Determine the maximum acceptable amount of data loss measured in time. This defines the point in time to which data must be restored after an incident. If an organization can tolerate 24 hours of lost data, then nightly backups are sufficient. If a shorter RPO is needed, then backups need to be more frequent.

Examples of Decisions Involving RTO and RPO

When making decisions about recovery time objective (RTO) and recovery point objective (RPO), organizations must consider the criticality of their systems and data. Here are three examples where RTO and RPO are super critical and five where they might not be worth the expense:

Super Critical RTO and RPO:

1. **Financial Services Transaction Systems**:
   - **Example**: Real-time banking transaction systems.
   - **RTO/RPO Importance**: Extremely critical because downtime or data loss can result in significant financial losses, legal penalties, and loss of customer trust.
   - **Decision**: Invest heavily in redundant systems, real-time data replication, and automated failover to ensure minimal downtime and data loss.

2. **Health Care Electronic Medical Records (EMR) Systems**:
    - **Example**: EMR systems used in hospitals.
    - **RTO/RPO Importance**: Super critical as these systems are essential for patient care, and any downtime or data loss can lead to life-threatening situations.
    - **Decision**: Implement robust backup solutions, high availability systems, and frequent data replication to ensure continuous operation and minimal data loss.
3. **Telecommunications Networks**:
    - **Example**: Core network infrastructure for mobile and internet services.
    - **RTO/RPO Importance**: Super critical as downtime can disrupt communication services for millions of users, leading to severe business and customer impact.
    - **Decision**: Invest in redundant network paths, real-time monitoring, and quick failover mechanisms to maintain continuous service availability.

Not Worth the Expense:

1. **Development and Testing Environments**:
    - **Example**: Non-production systems used by developers for testing.
    - **RTO/RPO Importance**: Not critical as downtime does not impact end users or business operations.
    - **Decision**: Implement basic backup and recovery procedures, accepting longer RTO and RPO to save costs.
2. **Internal Communication Tools**:
    - **Example**: Internal chat systems or non-essential email services.
    - **RTO/RPO Importance**: Moderate importance; while inconvenient, downtime does not halt business operations.
    - **Decision**: Use standard backup solutions with reasonable recovery times, balancing cost and convenience.
3. **Marketing Websites**:
    - **Example**: Company marketing websites that do not handle transactions or sensitive data.
    - **RTO/RPO Importance**: Low to moderate importance as these sites are not mission critical.
    - **Decision**: Implement basic web hosting and backup services, accepting longer recovery times during outages.

Checkpoint #3

1. Refer to the 10 company data problems mentioned earlier.
2. Research news reports published near the time of the incident.
3. Choose one of the 10 incidents where one of the companies response (or lack of) is most clearly shown.
4. Explain in at least one sentence the reason for your choice.

Factor Most Obvious in the Story	Company Involved	Reason(s) for Your Decision	Source Used to Make Your Decision (APA Citation)

Prevention			
Deterrence			
Detection			
Mitigation			
Recovery			

Part 5 – Stages of Data Vulnerability

Similar to the idea of states of matter (solid, liquid, gas), data in modern systems can exist in various states. Each state presents unique security challenges to protect sensitive information from unauthorized access, tampering, and loss.

By breaking down data into its various states—such as data in transit, at rest, or in use—you can more precisely target your security efforts. Additionally, understanding life cycle management, access control, and data classification helps in creating a comprehensive security framework that mitigates risks associated with data handling and storage. Each category requires specific strategies to ensure the integrity, confidentiality, and availability of data, thereby protecting it from various threats.

Category	Description	Security Considerations	Potential Hacking Techniques
Data in Transit (Data in Motion)	Data actively moving from one location to another, such as across the internet or through a private network.	Encryption (e.g., TLS/SSL), ensuring data integrity, protection from interception	Man-in-the-middle attacks, packet sniffing, traffic analysis, replay attacks.
Data at Rest (Data in Storage)	Data stored on a physical or digital medium, such as databases, files, or backups.	Encryption, access controls, data classification, physical security of storage devices.	Disk decryption attacks, physical theft, unauthorized access, malware attacks (e.g., ransomware).
Data in Use (in process)	Data actively being processed by applications or systems.	Securing data in memory, ensuring data is only accessible by authorized processes and users, protecting against data leaks during processing.	Side-channel attacks (e.g., listening to CPU sounds, electromagnetic emissions), memory scraping, keylogging, buffer overflow attacks, malware targeting in-memory data.
Data Life Cycle Management	Management of data throughout its	Data classification, retention policies, secure disposal, compliance with	Insufficient data deletion (e.g., remnants on storage devices), improper disposal

	lifecycle, from creation to deletion.	legal and regulatory requirements.	of physical media, unauthorized retention of data.
Data Integrity	Ensuring data accuracy and protection from tampering.	Checksums, digital signatures, redundancy, version control, regular audits to detect unauthorized changes.	Data tampering, unauthorized data modification, hash collision attacks, injecting false data, alteration of data during transmission.
Data Masking/Obfuscation	Hiding or obfuscating data to protect sensitive information, particularly in non-production environments.	Techniques such as tokenization, encryption, partial masking, and anonymization.	Reverse engineering, unauthorized re-identification of anonymized data.
Data Ownership and Stewardship	Identifying who is responsible for the data and how it is governed.	Defining roles, classification level, responsibilities, and accountability for data management and protection, assigning data stewards and owners.	Lack of clear ownership leading to security lapses, insider threats, unauthorized data manipulation or misuse by individuals with access.
Data Backups and Recovery	Creating copies of data to protect against data loss.	Secure backup storage, regular testing of backup integrity, secure recovery procedures, off-site storage, disaster recovery planning.	Backup data manipulation, unauthorized access to backups, ransomware targeting backups, improper handling of backup data leading to leakage or corruption.

Checkpoint #4

1. Refer to the 10 company data problems mentioned earlier.
2. Research news reports published near the time of the incident.
3. Choose one of the 10 incidents where of the “data states” is most relevant to the disaster and recovery.
4. Explain in at least one sentence the reason for your choice.

State of Data Most Relevant to the Story	Company Involved	Reason(s) for Your Decision	Source Used to Make Your Decision (APA Citation)
Data in Transit (Data in Motion)			
Data at Rest (Data in Storage)			
Data in Use (in Process)			
Data Life Cycle Management			

Data Integrity			
Data Masking/Obfuscation			
Data Ownership and Stewardship			
Data Backups and Recovery			

Part 6 – The OWASP Top 10

The Open-Source Foundation for Application Security (OWASP) publishes a Top 10 list of the most critical security risks to web applications. It is created by measuring the number of security incidents, trends seen by security professionals and academic research.

The latest updates to the top 10 list were done in 2021: Top 10 Web Application Security Risks

Here is the list of the 10 most serious problems encountered with application security.

1. Broken Access Control
2. Cryptographic Failures
3. Injection
4. Insecure Design
5. Security Misconfiguration
6. Vulnerable and Outdated Components
7. Identification and Authentication Failures
8. Software and Data Integrity Failures
9. Security Logging and Monitoring Failures
10. Server-Side Request Forgery (SSRF)

Let's describe each of the items and provide an example from a Spring Boot perspective.

1. Broken Access Control

Broken access control happens when users can access parts of the system they shouldn't be able to. This could mean unauthorized users accessing sensitive data or performing actions they're not allowed to.

Example: Imagine you have an online bookstore where only admins should be able to add new books. If a regular user can also access the add book page and successfully add a book, that's broken access control.

In **Spring Boot**, you might forget to restrict endpoints properly:

```
// A protected endpoint that should be accessible only to admins
@PostMapping("/admin/addBook")
```

```
public String addBook(Book book) {
    // Logic to add the book
}
```

Make sure you secure it with proper annotations:

```
@PreAuthorize("hasRole('ADMIN')")
@PostMapping("/admin/addBook")
public String addBook(Book book) {
    // Logic to add the book
}
```

2. Cryptographic Failures

Cryptography failure occurs when sensitive data is not properly protected using encryption. This can lead to data being exposed to unauthorized parties.

Example: Storing user passwords in plain text in the database is a cryptographic failure because if the database is compromised, all passwords are exposed.

In **Spring Boot**, always hash passwords before storing them:

```
// Bad example: Storing passwords directly
user.setPassword("plain-text-password");

// Good example: Using BCrypt to hash passwords
BCryptPasswordEncoder encoder = new BCryptPasswordEncoder();
user.setPassword(encoder.encode("plain-text-password"));
```

3. Injection

Injection flaws, such as SQL injection, happen when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands.

Example: Allowing user input to be included in a SQL query without proper validation or escaping:

```
// Bad example: Vulnerable to SQL injection
String query = "SELECT * FROM users WHERE username = '" + username + "'";
```

In **Spring Boot**, use parameterized queries to prevent SQL injection:

```
@Query("SELECT u FROM User u WHERE u.username = :username")
User findByUsername(@Param("username") String username);
```

4. Insecure Design

Insecure design refers to flaws in the application architecture that can't be fixed by proper implementation alone. These flaws might be avoided by using secure design patterns and principles from the beginning.

Example: An ecommerce application that doesn't require users to reauthenticate when performing critical actions like changing their email address.

In **Spring Boot**, ensure critical actions are protected:

```
@PostMapping("/updateEmail")
public String updateEmail(@RequestParam String email, HttpSession session) {
    // Ensure user is authenticated and re-verify sensitive actions
    if (session.getAttribute("user") != null) {
        // Logic to update email
    }
    return "error";
}
```

5. Security Misconfiguration

Security misconfiguratoin happens when security settings are not defined, implemented, or maintained properly, which can lead to vulnerabilities. This includes leaving default settings in place, incomplete configurations, misconfigured HTTP headers, and unnecessary features enabled (e.g., verbose error messages).

Example 1: Leaving default passwords on the database server that allow unauthenticated access.

In **Spring Boot**, always change default settings:

```
# Bad example: Default database credentials
spring.datasource.username=root
spring.datasource.password=root

# Good example: Secure database credentials
spring.datasource.username=bookadmin!0asllx
spring.datasource.password=BookGVEpass=word1@$XXXxx2
```

Example 2: Allowing dangerous files to be uploaded to an application.

- Allowing users to upload files without proper validation can result in uploading malicious files, which can be executed on the server, causing security breaches.
- Examples include not validating file types, allowing executable files (like .php, .exe), or not securing file paths, which can lead to directory traversal attacks.

6. Vulnerable and Outdated Components

Using components with known vulnerabilities can expose the application to various attacks.

Example: Using an outdated version of a library that has known security flaws.

In **Spring Boot**, keep dependencies updated. This dependency is typically found in the POM.XML file of an application.

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <version>2.6.0</version> <!-- Ensure this is the latest version -->
</dependency>
```

7. Identification and Authentication Failures

Explanation: Failures related to identifying users or managing user sessions can allow attackers to gain unauthorized access to systems.

Example 1: Not enforcing strong password policies.

In **Spring Boot**, enforce strong password policies:

```
public class CustomPasswordValidator implements
ConstraintValidator<ValidPassword, String> {
    @Override
    public boolean isValid(String password, ConstraintValidatorContext
context) {
        // Implement strong password validation logic
       // users must select 8 or more letters and use a variety of letters
and symbols.
        return password.length() >= 8 && password.matches(".*[A-Z].*") &&
password.matches(".*[!@#$%^&*].*");
    }
}
```

Example 2: CSRF (cross-site request forgery) failure. If a user is authenticated and the application does not implement proper CSRF protections, an attacker can hijack the authenticated session to perform actions on behalf of the user.

Example in Spring Boot: To protect against CSRF attacks in a Spring Boot application, you can enable CSRF protection, which is enabled by default in Spring Security. You should also use anti-CSRF tokens in forms and verify these tokens on the server side.

```
<!-- login.html -->
<form method="POST" action="/login">
      <input type="hidden" name="${_csrf.parameterName}"
      value="${_csrf.token}"/>
      <label for="username">Username:</label>
      <input type="text" id="username" name="username"/>
      <label for="password">Password:</label>
      <input type="password" id="password" name="password"/>
      <button type="submit">Login</button>
</form>
```

8. Software and Data Integrity Failures

This involves making assumptions about software updates and data integrity without verifying their authenticity.

Example: Not verifying the integrity of updates or relying on untrusted sources for software updates.

In **Spring Boot**, verify integrity using digital hashing signatures using a public key.

```
// Verify digital signatures for software components
public boolean verifySignature(byte[] data, byte[] signature, PublicKey
publicKey) {
    Signature sig = Signature.getInstance("SHA256withRSA");
    sig.initVerify(publicKey);
    sig.update(data);
    return sig.verify(signature);
}
```

9. Security Logging and Monitoring Failures

Not logging and monitoring security events can hinder the detection of breaches and forensic analysis.

Example: Not logging failed login attempts or not monitoring these logs.

In **Spring Boot**, implement logging and monitoring:

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class SecurityController {
    private static final Logger logger =
LoggerFactory.getLogger(SecurityController.class);

    @PostMapping("/login")
    public String login(@RequestParam String username, @RequestParam String
password) {
        // Log login attempts
        logger.info("Login attempt for user: " + username);
        // Authentication logic
    }
}
```

10. Server-Side Request Forgery (SSRF)

SSRF occurs when an attacker can trick the server into making requests to unintended locations, potentially accessing internal se rvices.

Example: An application that fetches data from a URL provided by the user without proper validation.

In **Spring Boot**, validate and sanitize user inputs for URLs:

```
public String fetchDataFromUrl(String url) {
    // Validate the URL to prevent SSRF.
    // Only respond to URLs that begin with yourdomain.com
    if (isValidUrl(url)) {
        // Fetch data logic
    } else {
        throw new IllegalArgumentException("Invalid URL");
    }
}
```

Checkpoint #5

1. Refer to the 10 company data problems mentioned earlier.
2. Research news reports published near the time of the incident.
3. Choose one of the 10 incidents where one if the OWASP top 10 is most clearly shown.
4. Explain in at least one sentence the reason for your choice.

Factor Most Obvious in the Story	**Company Involved**	**Reason(s) for Your Decision**	**Source Used to Make Your Decision (APA Citation)**
Broken Access Control			
Cryptography Failure			
Injection			
Insecure Design			
Misconfiguration of Security			

Vulnerable and Outdated Software			
Identification and Authentication Failure			
Data Integrity Failure			
Security Logging and Detection Failure			
Server-Side Request Forgery			

Where You Will See Examples of These Issues in This Course

OWASP Top 10	Lessons Where You Will See This Issue	Relevance to OWASP Issue
1. Broken Access Control	- Spring Security Configuration - REST API Security with JWT	These lessons cover the importance of correctly configuring access controls, such as restricting resource access to authenticated users and ensuring that unauthorized users cannot access protected resources. These are key aspects of mitigating broken access control vulnerabilities.
2. Cryptographic Failures	- Password Hashing and Cracking - GNU PGP - Wireshark and SSL	These lessons teach the correct implementation of cryptography to protect sensitive data. Password hashing and SSL encryption are crucial in ensuring that sensitive information, like passwords and communication data, is not exposed to attackers.
3. Injection	- SQL Injection - Cross-Site Script Injection	Injection flaws like SQL injection and XSS are addressed in these lessons, highlighting how improper handling of user inputs can lead to severe security breaches. They also emphasize the importance of sanitizing and validating inputs to prevent such attacks.
4. Insecure Design	- Spring Security Configuration - REST API Security with JWT - Wireshark and SSL	These lessons focus on secure design principles, such as implementing proper security controls (e.g., SSL for encrypted communication) and designing secure authentication mechanisms, which are essential in preventing vulnerabilities due to insecure application design.
5. Security Misconfiguration	- Spring Security Configuration - REST API Security with JWT - Wireshark and SSL - SQL Injection	These lessons highlight the risks associated with improperly configured security settings, such as inadequate enforcement of security policies, incorrect configurations, and exposure of sensitive information due to misconfigurations.

	- Cross Site Script Injection	
6. Vulnerable and Outdated Components	- Password Hashing and Cracking - GPG Encryption	Using outdated or vulnerable components in your application, such as outdated cryptographic libraries, can lead to security risks. These lessons emphasize keeping components up to date and using secure, modern cryptographic practices.
7. Identification and Authentication Failures	- Spring Security Configuration - REST API Security with JWT	These lessons focus on implementing strong identification and authentication mechanisms, such as enforcing strong passwords and using JWT tokens, to ensure that users are properly authenticated and unauthorized access is prevented.
8. Software and Data Integrity Failures	- Wireshark and SSL	This lesson addresses the integrity of data transmission by ensuring that data is encrypted using SSL, which helps prevent tampering and ensures that data integrity is maintained during communication.
9. Security Logging and Monitoring Failures	-	This OWASP issue is not directly addressed in the lessons listed, but it is critical to implement logging and monitoring mechanisms to detect and respond to security incidents.
10. Server-Side Request Forgery (SSRF)	-	SSRF is not directly covered in these lessons, but it highlights the importance of validating and sanitizing external inputs to prevent unauthorized access to internal resources through server-side requests.

Deliverables

Submit a Microsoft Word document that contains completed tables for all five checkpoints.

Table 1 CIA

Problem	CIA Issue(s) Involved	Reason(s) for Your Decision	Source Used to Make Your Decision (APA Citation)
1. Yahoo			
2. Equifax			
3. Target			
4. NASA			
5. Knight			
6. Colonial Pipeline			

7. SolarWinds			
8. AT&T			
9. Ticketmaster			
10. CrowdStrike			

Table 2 MOM

Factor Most Obvious in the Story	**Company Involved**	**Reason(s) for Your Decision**	**Source Used to Make Your Decision (APA Citation)**
Motive			
Opportunity			
Method			

Table 3 Company Response

Factor Most Obvious in the Story	**Company Involved**	**Reason(s) for Your Decision**	**Source Used to Make Your Decision (APA Citation)**
Prevention			
Deterrence			
Detection			
Mitigation			
Recovery			

Table 4 States of Data

State of Data Most Relevant to the Story	**Company Involved**	**Reason(s) for Your Decision**	**Source Used to Make Your Decision (APA Citation)**

Data in Transit (Data in Motion)			
Data at Rest (Data in Storage)			
Data in Use (in process)			
Data Life Cycle Management			
Data Integrity			
Data Masking/Obfuscation			
Data Ownership and Stewardship			
Data Backups and Recovery			

Table 5 OWASP 10

Factor Most Obvious in the Story	**Company Involved**	**Reason(s) for Your Decision**	**Source Used to Make Your Decision (APA Citation)**
Broken Access Control			
Cryptography Failure			
Injection			
Insecure Design			
Misconfiguration of Security			
Vulnerable and Outdated Software			
Identification and Authentication Failure			
Data Integrity Failure			

Security Logging and Detection Failure			
Server-Side Request Forgery			

Check for Understanding

Although these questions are not graded, they will help you prepare for upcoming assessments.

1. Which component of the CIA triad ensures that information is only accessible to those with proper authorization?

    - A) Integrity
    - B) Availability
    - C) Confidentiality
    - D) Authentication

Correct Answer: C) Confidentiality

2. In the Equifax data breach, which part of the CIA triad was primarily compromised?

    - A) Integrity
    - B) Availability
    - C) Confidentiality
    - D) Accountability

Correct Answer: C) Confidentiality

3. The Mars Climate Orbiter incident is an example of a failure in which aspect of the CIA triad?

    - A) Confidentiality
    - B) Integrity
    - C) Availability
    - D) Authentication

Correct Answer: B) Integrity

4. What was the primary method used by attackers in the Yahoo data breach?

    - A) SQL Injection
    - B) Forged Cookies
    - C) Phishing
    - D) Man-in-the-Middle Attack

Correct Answer: B) Forged Cookies

5. Which of the following was the main vulnerability exploited in the SolarWinds attack?

   - A) Server-Side Request Forgery (SSRF)
   - B) Insecure Design
   - C) Vulnerable and Outdated Components
   - D) Injection

Correct Answer: C) Vulnerable and Outdated Components

6. The Colonial Pipeline attack is best categorized under which of the MOM. factors?

   - A) Motive
   - B) Method
   - C) Opportunity
   - D) All of the above

Correct Answer: C) Opportunity

7. Which cybersecurity response type involves educating employees about recognizing potential threats like phishing attacks?

   - A) Prevention
   - B) Detection
   - C) Mitigation
   - D) Recovery

Correct Answer: A) Prevention

8. The Knight Capital Group trading glitch is primarily an example of failure in which stage of data vulnerability?

   - A) Data at Rest
   - B) Data in Transit
   - C) Data in Use
   - D) Data Life Cycle Management

Correct Answer: D) Data Life Cycle Management

9. Which OWASP Top 10 issue is demonstrated by the Ticketmaster data breach?

   - A) Cryptographic Failures
   - B) Injection
   - C) Security Misconfiguration
   - D) Insecure Design

Correct Answer: B) Injection

10. Which aspect of the CIA triad was most impacted by the CrowdStrike Global Technology Outage?

- A) Confidentiality
- B) Integrity
- C) Availability
- D) Authentication

Correct Answer: C) Availability