Threat Modeling

Thomas Stiehm
Coveros, Inc
@thomasstiehm

Thomas Stiehm

Thomas has been developing and securing applications and managing software development teams for over twenty-eight years. As CTO of Coveros, he is responsible for the oversight of all technical projects and integrating new technologies and testing practices into software development projects. Recently, Thomas has been focusing on how to incorporate DevSecOps and agile best practices into projects and how to achieve a balance between team productivity and cost while mitigating project risks. One of the best risk mitigation techniques Thomas has found is leveraging DevSecOps and agile testing practices in all aspects of projects.

About Coveros

- Founded in 2008, Coveros helps organizations accelerate software delivery

- **Consulting Services**
  - Agile/DevOps Transformations
  - Agile Software Development
  - Agile Testing
  - DevOps Engineering
  - DevSecOps Integrations

- **Agile, DevOps, DevSecOps, Testing Training**

- **Open Source Products**
  - SecureCI—Secure DevOps toolchain
  - Selenified—Agile test framework

Awards

Additional Resources

TechWell Conferences

Agile + DevOps CONFERENCES
TECHWELL EVENTS

STAR CONFERENCES
TECHWELL EVENTS

EPIC EXPERIENCE
A TECHWELL EVENT

Online Communities and Newsletters

AGILE CONNECTION™
A TECHWELL COMMUNITY

STICKYMINDS™
A TECHWELL COMMUNITY

TECHWELL™
INSIGHTS

Agenda

- Introduction to Information Security
- What is Threat Modeling?
- Threat Modeling Approaches
- Threat Modeling Walkthrough
- Wrap-Up

Introduction to Information Security

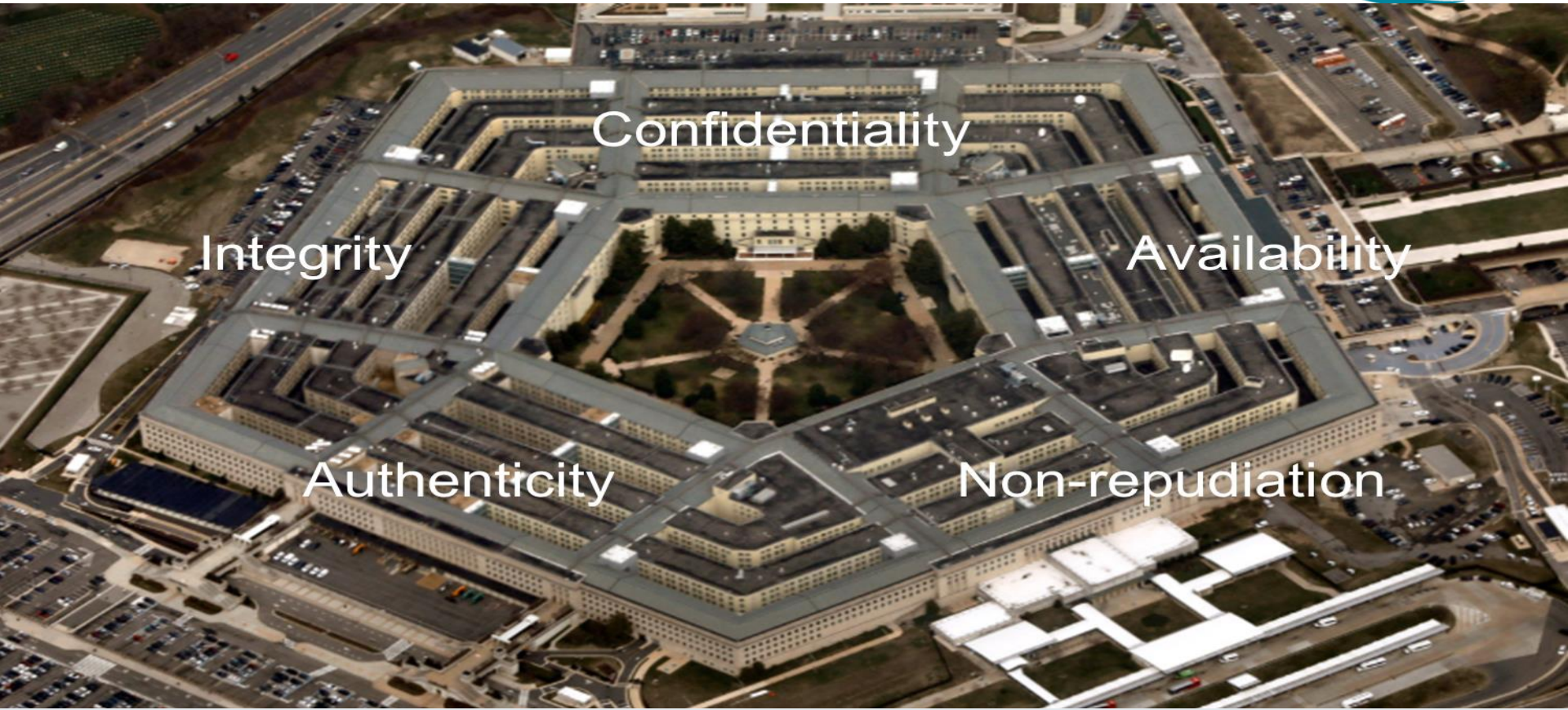
Introduction to Information Security

Information Security is defined as protecting information and information systems from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

Information Security encompasses all aspects of the systems development lifecycle by protecting information and systems across multiple disciplines:

- Physical
- Social
- Networks
- Applications
- Devices

Security Principles



Security Terminology

Assets – a resource of value. May be tangible or intangible.

- For many of our systems our resource of value is information our applications collect, store, and use to perform operations for users

Security Terminology (cont.)

Threat/Risk – Undesired act that potentially occurs causing compromise or damage of an asset.

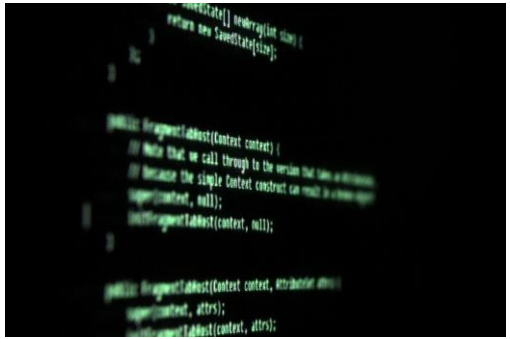
Threat Agent – Something/someone that purposefully makes a threat materialize.

Attack – Malicious act of a threat agent. Also known as Exploit.

Security Terminology (cont.)

Vulnerability – Weaknesses that makes an attack possible (human, process, software, physical, etc.)
- Inside software applications, design flaws and code defects are weaknesses

Security Controls – mechanisms to mitigate vulnerabilities



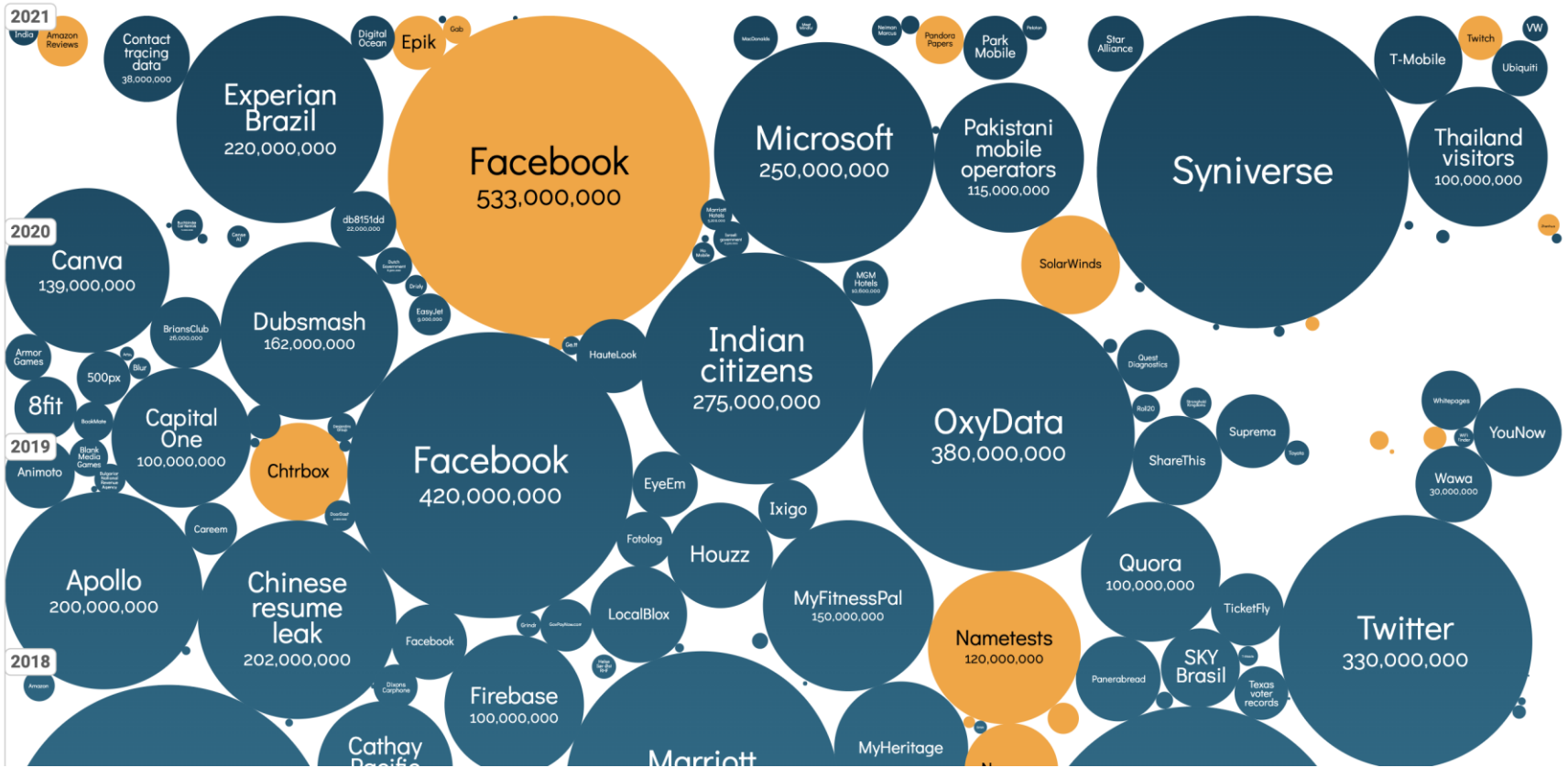
Security Terminology (cont.)

Likelihood – the probability of a threat being realized by an attack on an asset

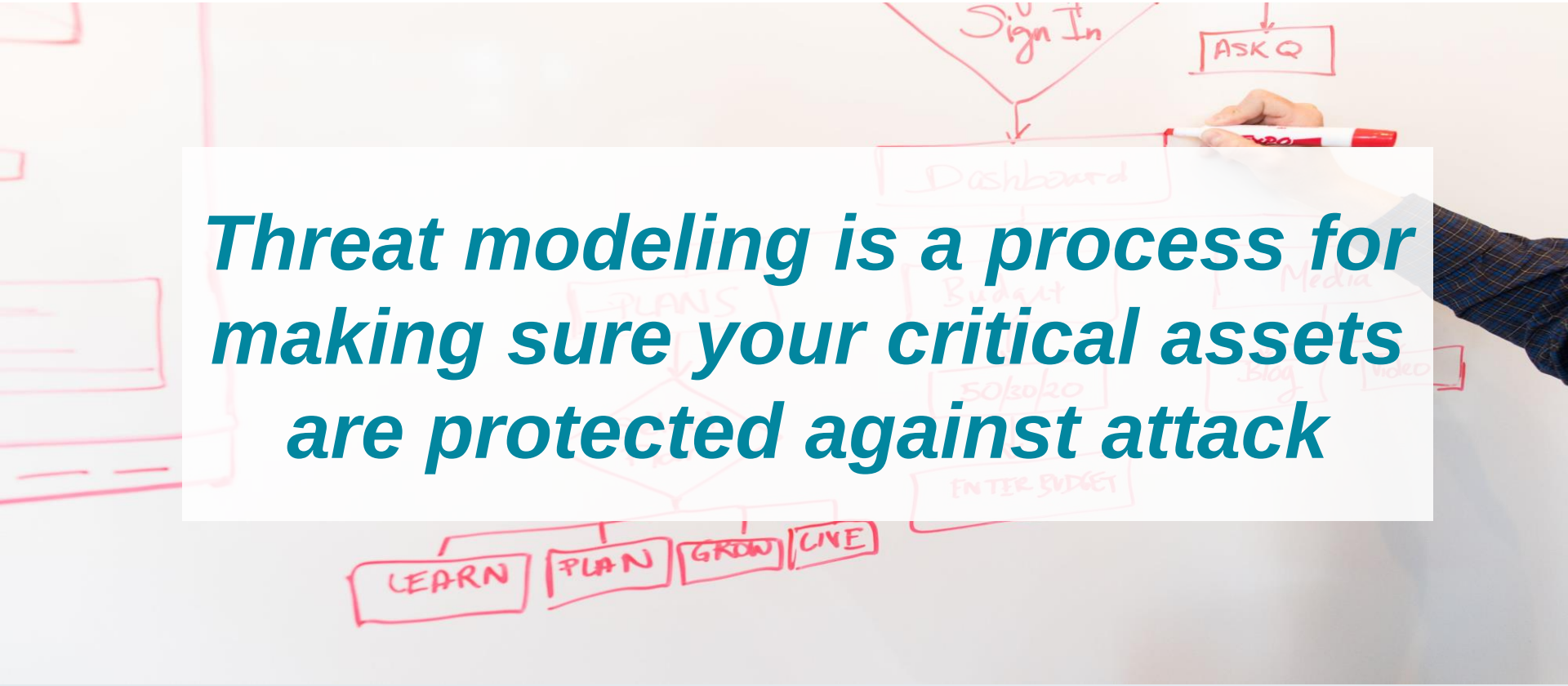
Impact – Business consequence if an asset is compromised by a threat actor

For large systems, estimating Impact and Likelihood is part of the risk analysis we do to determine where to concentrate scarce security resources to protect ourselves as much as possible.

Security Compromises are Common



What is Threat Modeling?



Threat modeling is a process for making sure your critical assets are protected against attack

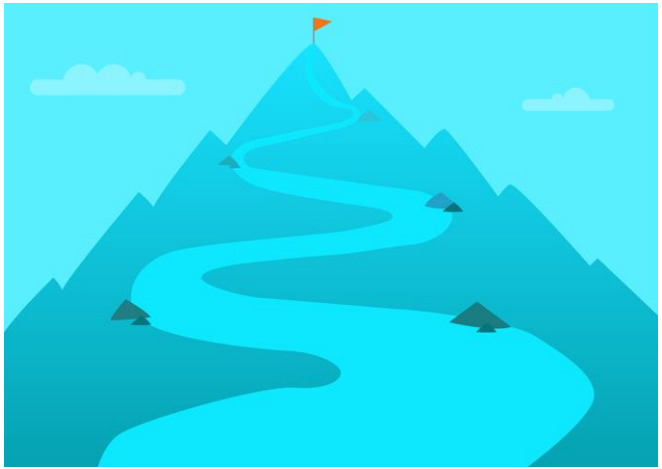
Threat Modeling Questions

Questions to answer during threat modeling

No matter what threat modeling approach you use, success depends on whether it can be used to answer four simple questions:

1. What software are we building?
2. What security risks are present?
3. How do we mitigate identified risks?
4. Did we do a good job threat modeling?

Each of these questions aligns with a step in the threat modeling process



Threat Modeling – What are we building?

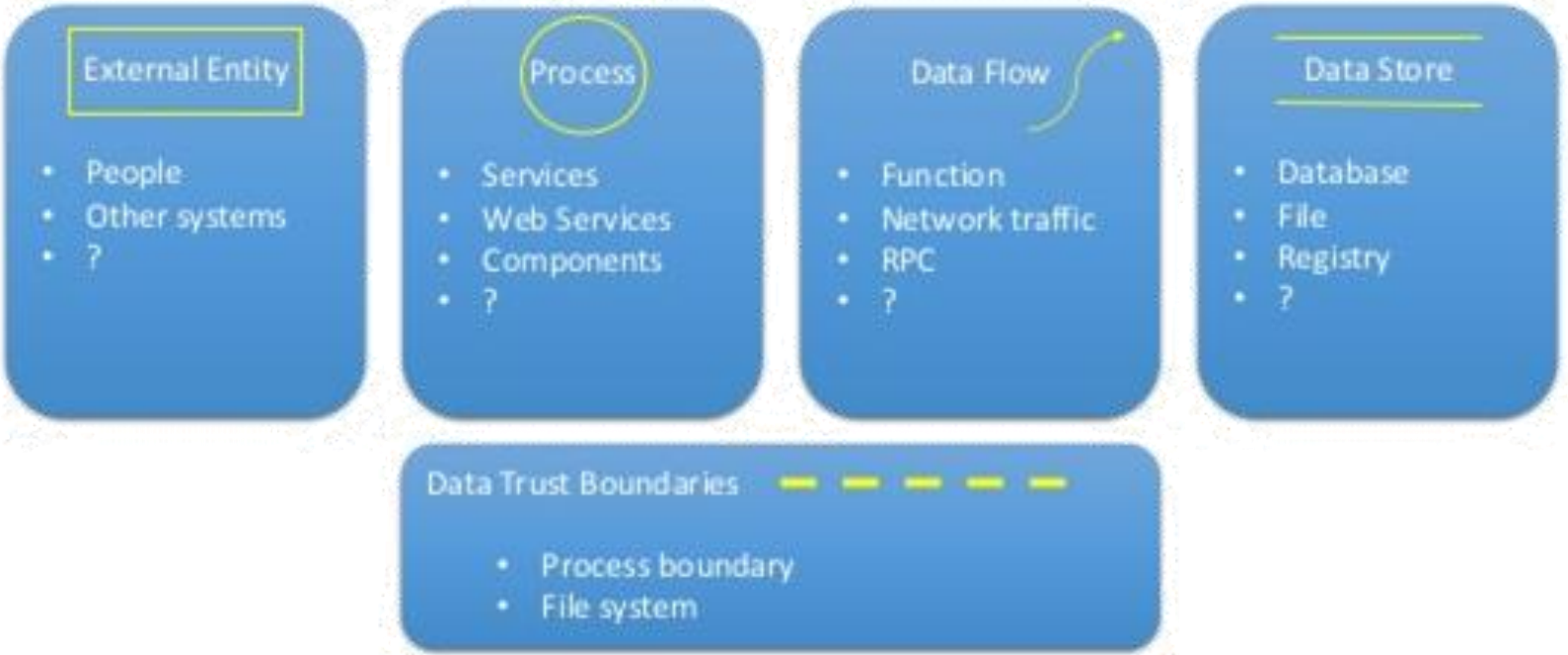
Diagram Your System

As a starting point you need to define the scope of the Threat Model. To do that you need to understand the application you are building. Examples of helpful techniques are:

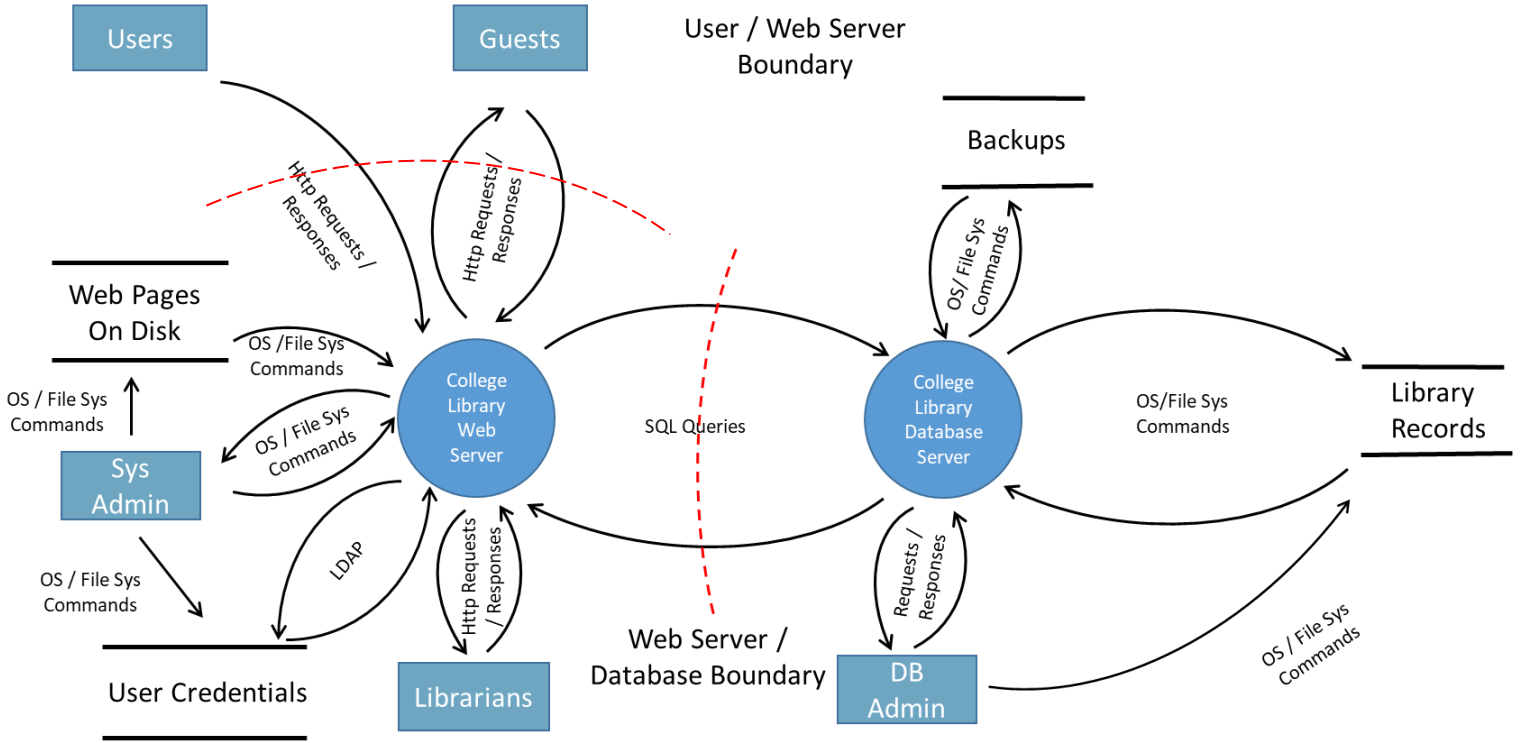
- Architecture diagrams
- Dataflow transitions
- Data classifications

The process of thinking through the security of your applications is more important than the diagramming or threat modeling approach you use!

Threat Modeling – Dataflow Diagrams



Dataflow Diagram Example



Threat Modeling – What are we building?

Classify your information assets

Data classification segments information assets by the security principles that must be upheld, and the impact of a compromise to the organization.

Determine which security principle(s) must be upheld for each information asset

Determine the business consequence of a breach for each information asset

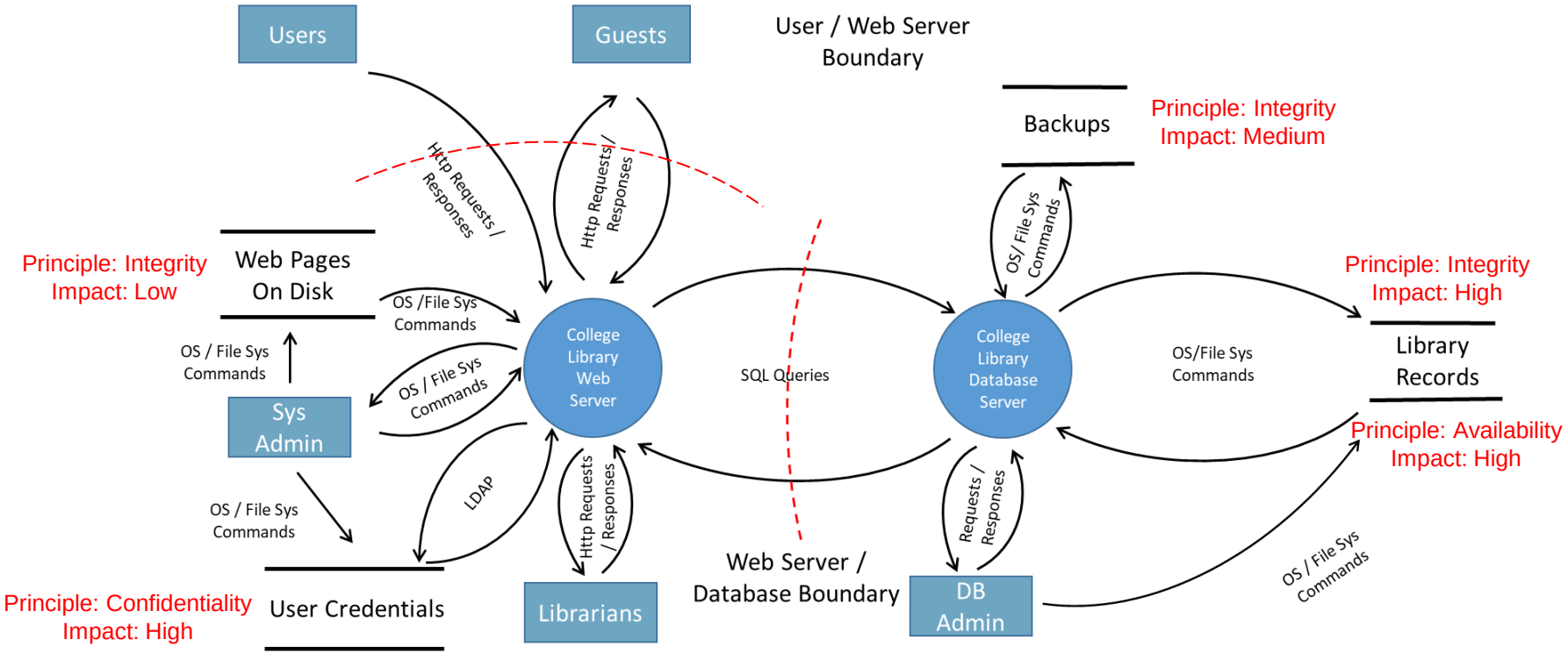
- High – There is significant business loss if a breach occurs
- Medium – There is some business loss if a breach occurs
- Low – There is little business loss if a breach occurs

Threat Modeling – What are we building?

Classify your information assets

Asset / Information	Description	Security Principles & Business Consequence
Library records	Books in the library and their current status	
User credentials	ID & Passwords for various types of users	
Web site pages	Web site pages for library	
Backups	Backups of library records	

Data Classification Example



What security risks are present?

2. Examine system to identify areas of security concern and potential attacks

First technique focuses on system components and their interfaces

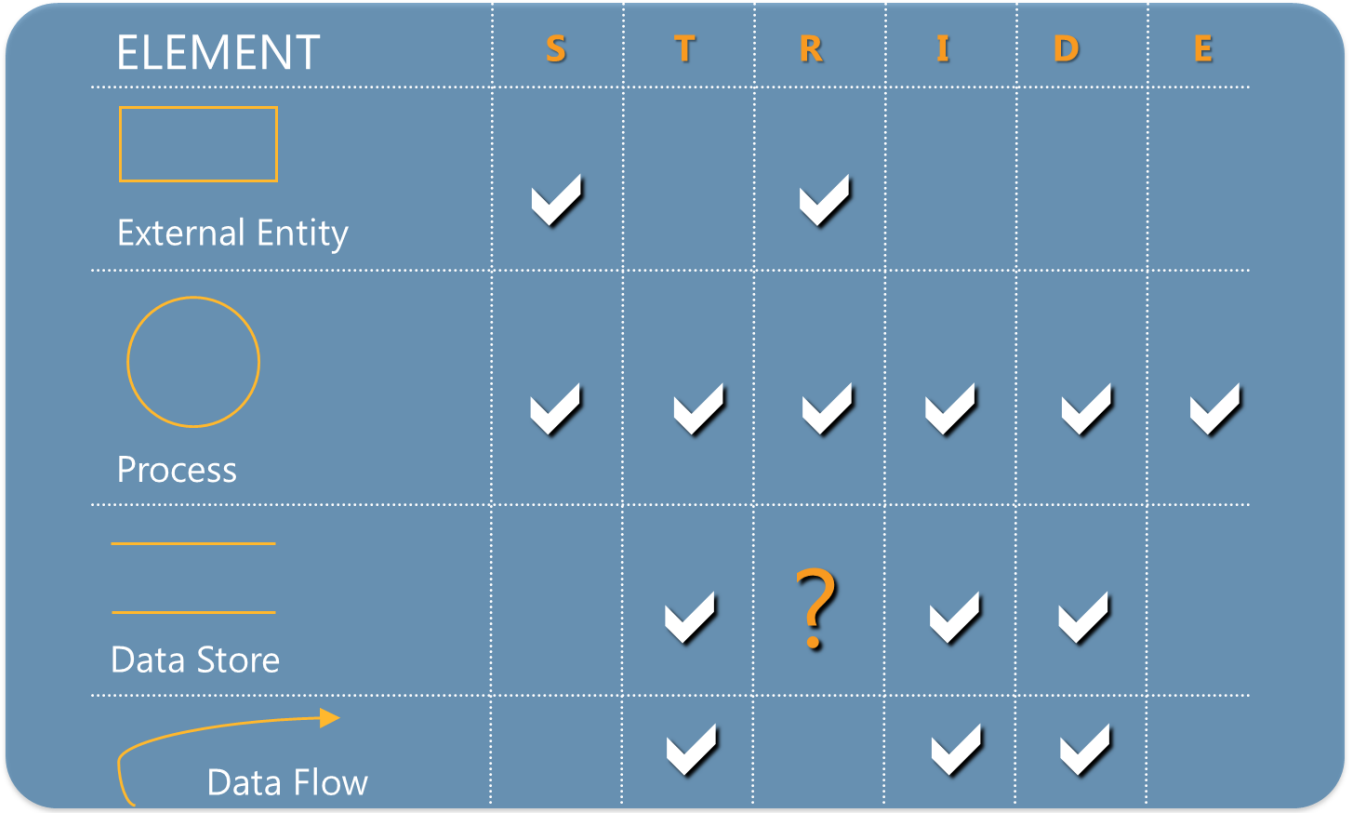
- Examine your external entities, processes/services, dataflows, and datastores to determine which threats are possible between components
- Focus on interactions that cross trust boundaries
- Use results of Data Classification to focus on critical areas
- Software development teams often use this approach to identify where security controls should be integrated into an application

Threat Modeling – Identify Threats

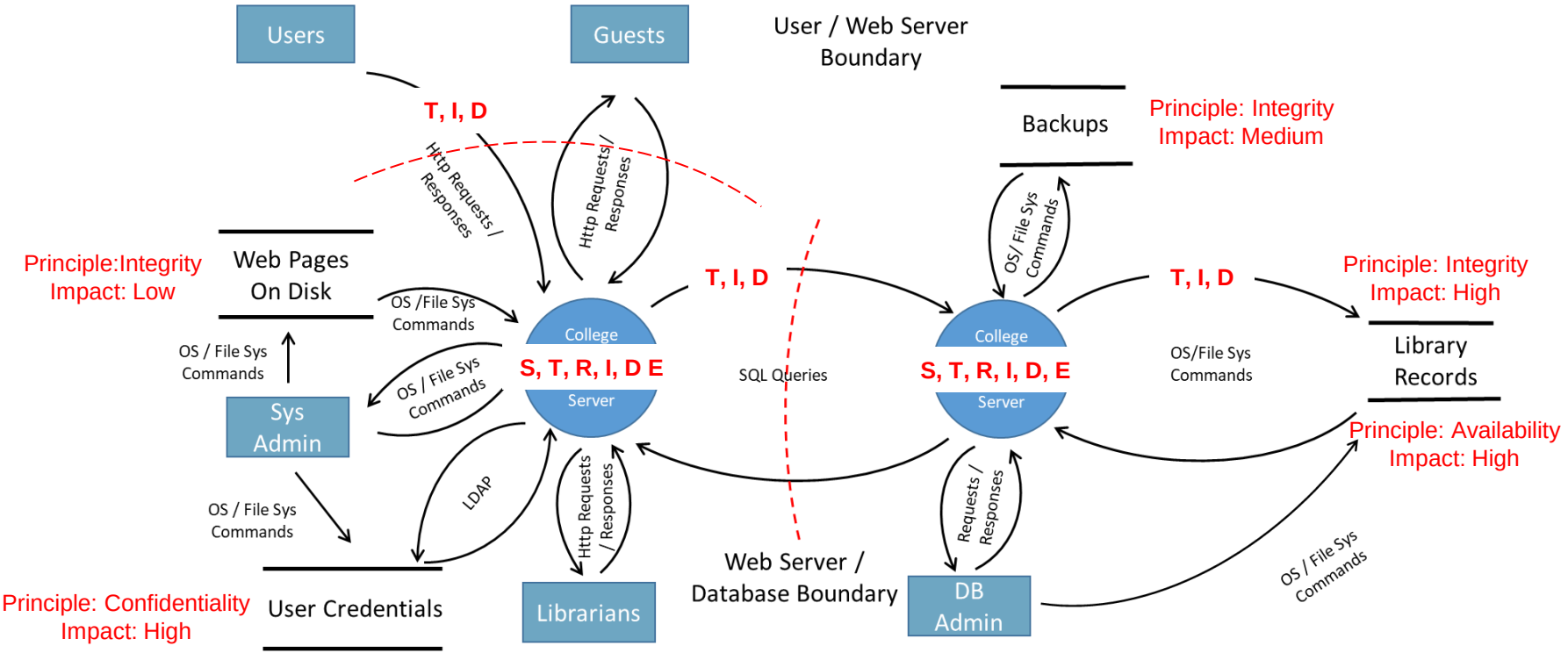
Common Threat Categories (STRIDE)

- Spoofing – pretending you are someone you aren't
- Tampering – modifying information
- Repudiation – being able to deny something happened
- Information disclosure – disclosing private information to others
- Denial of service – disallowing access to critical information
- Escalation of privilege – increasing level of access

Threat Modeling – Identify Threats



Annotate your model with STRIDE



What security risks are present?

2. Examine system to identify areas of security concern and potential attacks

Second approach is to focus on threat agents and attack patterns

Seek to identify what attacks are possible and what threat agents are of biggest concern

- Determine those threat agents most likely to be a risk
- Examine attack patterns and previous attacks
- Pay special attention to insiders!
- Penetration testers often use this approach to security test your systems

Threat Modeling – Attackers

Common Threat Actors

- Identity thief – seeking to steal the identity of customers
- Malicious administrator – seeking to harm the organization
- Professional hacker – looking to steal IP, hold information for ransom
- Organized crime – money laundering, stealing identities
- Cyber terrorist – shut down critical infrastructure, steal IP
- Script kiddie – bored kids/students running automated scripts
- Disgruntled employee – seeking to harm the organization
- Thief/Spy – seeking to steal secrets, IP, identities

Threat Modeling – Attackers

Identify Potential Threat Actors

Threat Actor	Asset(s) of interest	Likely to Attack
Professional hacker		
Disgruntled sys admin		
Disgruntled user		
Identity thief		
Disgruntled DB admin		

Threat Modeling – Attack Patterns

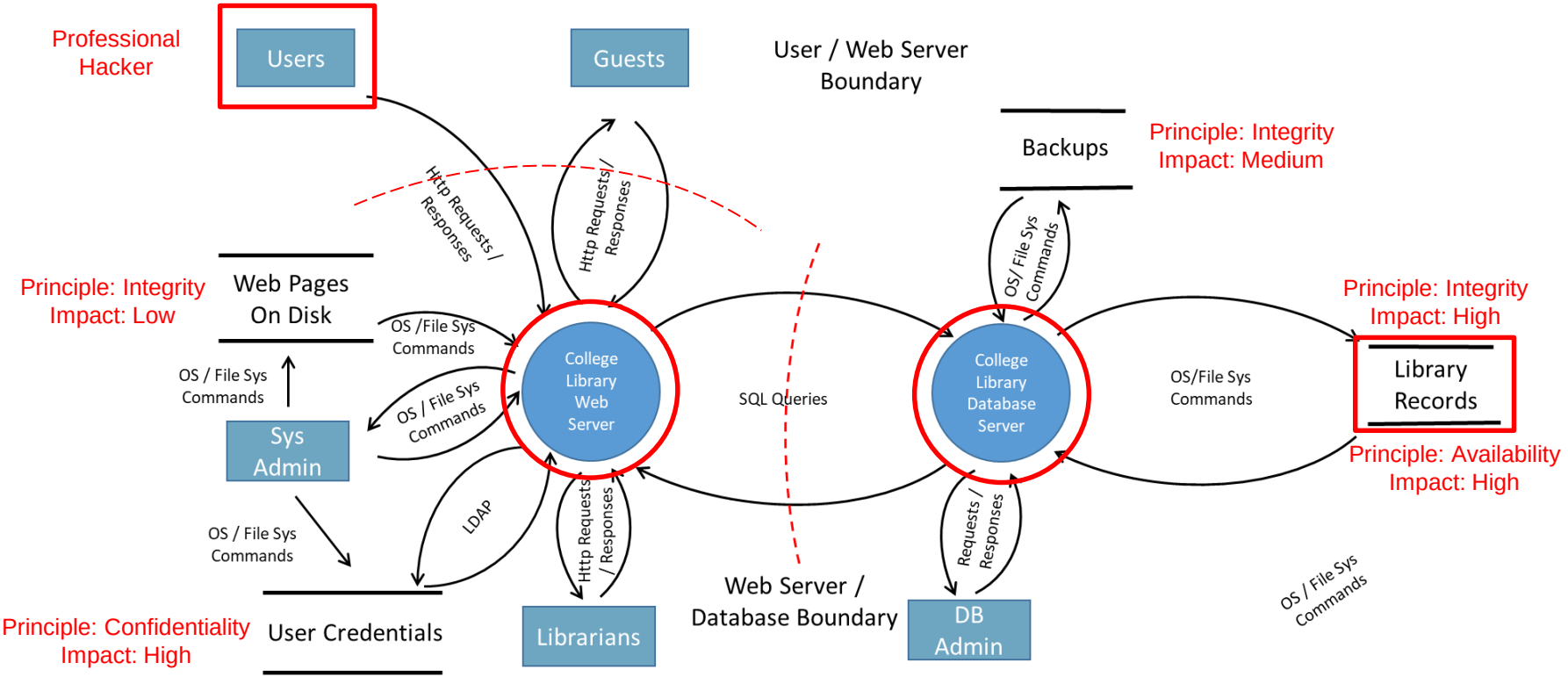
Common Domains of Attack

- Exploiting trust in client
- Man in the middle attack
- API manipulation
- Privilege abuse
- Flooding
- Excessive allocation
- Resource leak exposure
- Parameter injection
- Identity spoofing
- File manipulation

Common Domains of Attack (cont.)

- Software integrity attack
- Reverse engineering
- Malicious logic insertion
- Functionality bypass
- Use of knowledge domain credentials
- Obstruction
- Functionality misuse
- Resource location spoofing
- Traffic injection
- Protocol manipulation

Threat Modeling – Attack Patterns



Threat Modeling – Attack Patterns

Attack Categories Against Availability

- Flooding – Sending large amounts of data into the UI or API
- Fuzzing – Sending improperly formed data into the system
- Obstruction – Thwarting operations from happening
- Lockouts – Attempt to lock users out from the system

Scenario to mitigate:

Professional Hacker comes in as a User and uses availability attacks against the Database Server through the Web Server to try and thwart availability for others

Threat Modeling – Ranking Scenarios

- Based on the number of critical assets, size of system, number of realistic threat actors, and identified threats, you may need to prioritize your mitigation activities by scenario

- As discussed, Impact is based on the consequence of an asset associated with a threat being breached

- Likelihood is more difficult to determine and should be based on a combination of:
  - Attack surface for threat actors
  - Frequency of threats / attacks (Top 10s)
  - Historical attack data
  - System/company profile or line of business
  - Security posture

Impact \ Likelihood	High	Medium	Low
High	High	Medium	Medium
Medium	High	Medium	Low
Low	Low	Low	Low

How do we mitigate identified risks?

3. Identify ways to protect our systems against attacks

Apply mechanisms to reduce the impact or likelihood of identified threats

- Integrate security controls
- Apply defensive design principles
- Use secure coding practices
- Secure libraries and software component analysis

Use defense in depth for the most critical risks!

Threat Modeling – Security Controls

- Mechanisms put into software to uphold the security principles that must be met for critical assets

- Example security controls
  - Confidentiality—Encryption, App firewalls
  - Integrity—Hash functions, Encryption
  - Availability—Load balancers, Redundant networks
  - Authenticity—Login/Password, Two factor authentication
  - Non-repudiation—Email receipts, delivery devices, package tracking
  - Cross-cutting controls – multi-admin logins, logging, application monitoring, security capabilities in frameworks/platforms/cloud

- Don’t forget defense in depth

Threat Modeling – Defensive Design

- Best practices, patterns, principles for how to design software to reduce the risk of attacks
- Defensive design examples:
  - Principle of least privilege—don't give a user or service any more permissions than they need
  - Fail securely—anticipate errors and exceptions and handle them securely
  - Provide multiple layers of security protection for critical assets
- Tools can point to SOME violations of defensive design, but many are based on your business logic

Threat Modeling – Secure Coding

- Best practices for writing code that does not include known vulnerabilities or weaknesses in the libraries and languages you are using.

- Secure code examples:
  - Use functions that manage string and buffer sizes
  - Validate inputs so they do not contain executable code
  - Use strongly typed languages to thwart type confusion
  - Avoid calling the operating system directly

- Leverage IDE security grammar checkers and code scanners to validate posture

Threat Modeling – Secure Libraries & OS

- Libraries of secure components exist and can be leverages to make sure critical functionality is properly implemented
- Don’t forget to use secure security controls too!
- Use open-source frameworks, operating systems, services, and libraries that have no known vulnerabilities in them (National Vulnerability Database)

Did we do a good job?

4. Get feedback and review our results

Carry out a retrospective activity over the work you have done to check quality, feasibility, progress, and/or planning.

Walk through threat model with someone who wasn't involved in the process

Ultimately our threat modeling will be evaluated during security testing, penetration testing, and what happens in the field!

Threat Modeling Approaches

STRIDE

STRIDE, Microsoft's threat modeling methodology, is the oldest, most well-documented, and most mature methodology. It was developed to help ensure that developers of Microsoft software think about security during the design phase. As such, STRIDE is ***highly development-focused***.

STRIDE stands for **Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege**, and it tries to map security principles of the CIA triad to architecture and data-flow diagrams.

Once a team constructs a data-flow diagram, engineers check the application against the STRIDE classification scheme. The outputs show threats and risks and are derived right from design diagrams as part of the development process.

PASTA

The Process for Attack Simulation and Threat Analysis, or PASTA, is a seven-step process for risk analysis that is ***attacker-focused***. The goal of this methodology is to align business objectives with technical requirements while taking into account business impact analysis and compliance requirements.

The process focuses on assets to evaluate risk with regard to impact to the business. PASTA threat modeling works best for organizations that wish to align threat modeling with strategic objectives because it incorporates business impact analysis.

TRIKE

TRIKE is a compliance-focused threat modeling process focused on satisfying security auditing requirements. TRIKE focuses on a requirements model that assigns acceptable levels of risks to each asset. It is **compliance-focused**.

Once in place, the team creates data flow diagrams and threats are enumerated with appropriate risk values. Users then build mitigating controls and prioritize threats. Because it requires the team to understand the entire system, it can be a challenge to apply this process to large-scale systems.

VAST

The Visual, Agile, and Simple Threat modeling methodology scales the threat modeling process across the infrastructure for the entire software development lifecycle, integrating with agile and DevOps practices. VAST is **enterprise-focused** and provides actionable outputs for the different needs of every stakeholder.

Because developers' security concerns will be different than those of the infrastructure team, VAST allows teams to create either process flow diagrams mapping out the application, or operational threat models showing data flow.

INFORMAL ANALYSIS

While formal models can help structure your overall threat modeling approach, many use a more informal, white boarding approach as ultimately there is only one key step in a threat modeling process:

THINK

about security

Common Mistakes

Common mistakes

Not thinking like an attacker.

It's hard to think like an attacker if you probably aren't, in fact, an attacker. At best, you'll be guessing what an attacker is thinking about – or how they're planning to behave.

Check Out: Cybersecurity basics from the NIST or SANS guidelines before you get into the more esoteric. Continue your education through training, webinars, conferences, and Meetups!

Check Out: MITRE CAPAC and ATT&CK frameworks to learn more about attack patterns

TRY: Bringing your cybersecurity team into the discussion and rely on their subject matter expertise.

TRY: Starting with STRIDE as it was built for non-security personnel

Common mistakes

Trying to do it all.

No, extraterrestrials are not going to corrupt your data with their advanced system interface technology. If they could, what are you actually going to do about it? Focus on the risks that are real and manageable.

TRY: Prioritize by Impact and Likelihood to focus on those risks and scenarios that are most important to protect your valuable assets

Common mistakes

Threat modeling is a one and done.

This is a double-edged sword. Never assume that the threat modeling team can imagine every potential threat that will ever exist, and don't postpone deployment of a new system because there is a miniscule amount of risk either. So when the boss asks if the threat model is complete, give them a realistic risk assessment and tell them that when the risk profile changes, there's a plan to update the threat models.

TRY: Treat your threat model like a living document or a wiki.

TRY: If using Agile, review threat model at Sprint / Iteration kickoff to make sure proposed features don't change our threat profile

Common mistakes

No CISSP, no dice.

Gather a diverse team and include the stakeholders as well as customer voices, if possible – no certifications required. If necessary, have someone stand in for the customer – support techs are good at that role. Bring cybersecurity expertise to the team, but don't exclude anyone based on that criterion alone.

TRY: Build a team with someone from all areas of the business: Developer, Operations, Product Owner, Security, Quality, etc.

Common mistakes

Don’t worry about that old system, we don’t need to develop a threat model for that.

This is a huge mistake – just look at some of the recent data breaches making headlines. Go back through the service catalog and build threat models for any systems that don’t have one. It could be a colossal task, but the cost of not protecting your data and systems is only getting higher.

Wrap Up

Tools and Resources

Microsoft Threat Modeling Tool
https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling

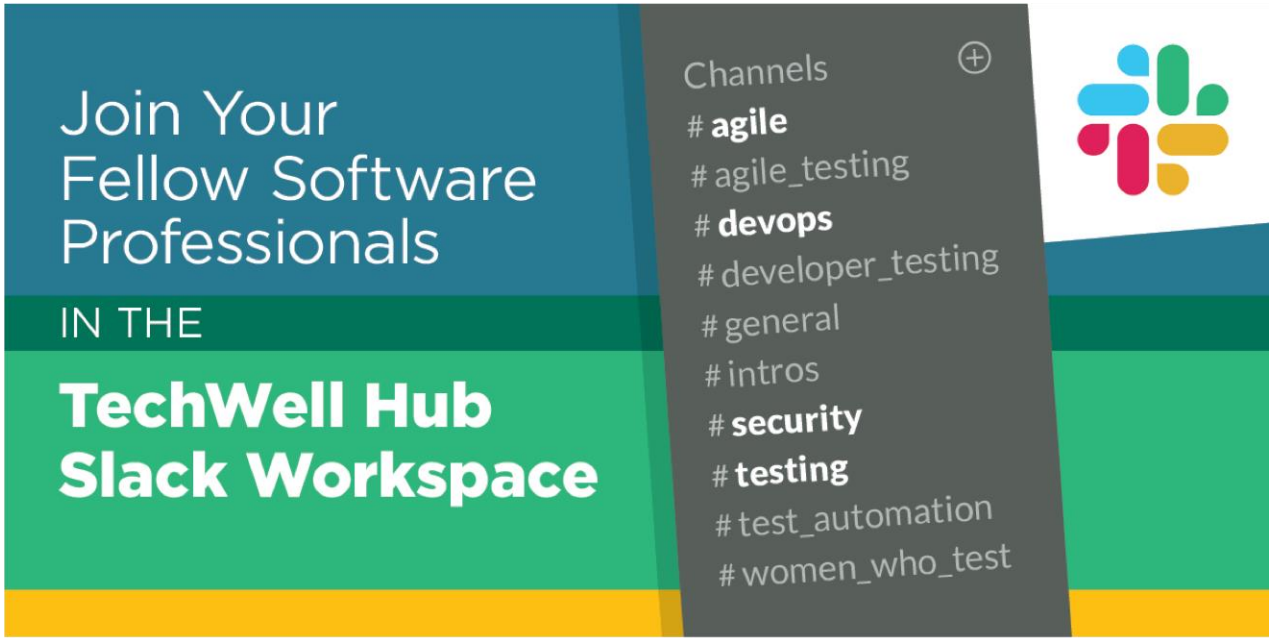
OWASP Threat Dragon Threat Modeling Tool
https://threatdragon.org/login

OWASP Threat Modeling Cheat Sheet
https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html

“Threat Modeling: Designing for Security” by Adam Shostack
https://www.amazon.com/dp/B00IG71FAS

Threat Modeling Card Game
https://www.microsoft.com/en-us/download/details.aspx?id=20303

Additional Resources



Join today at Hub.TechWell.com

Questions

tom.stiehm@coveros.com

@thomasstiehm

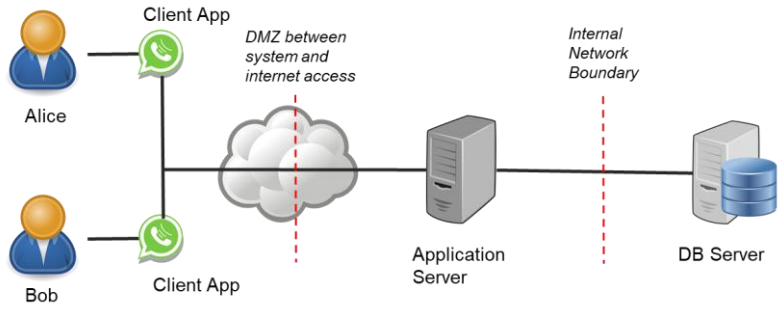
Threat Modeling Walkthrough (Homework)

Exercise #1 - Creating a DFD

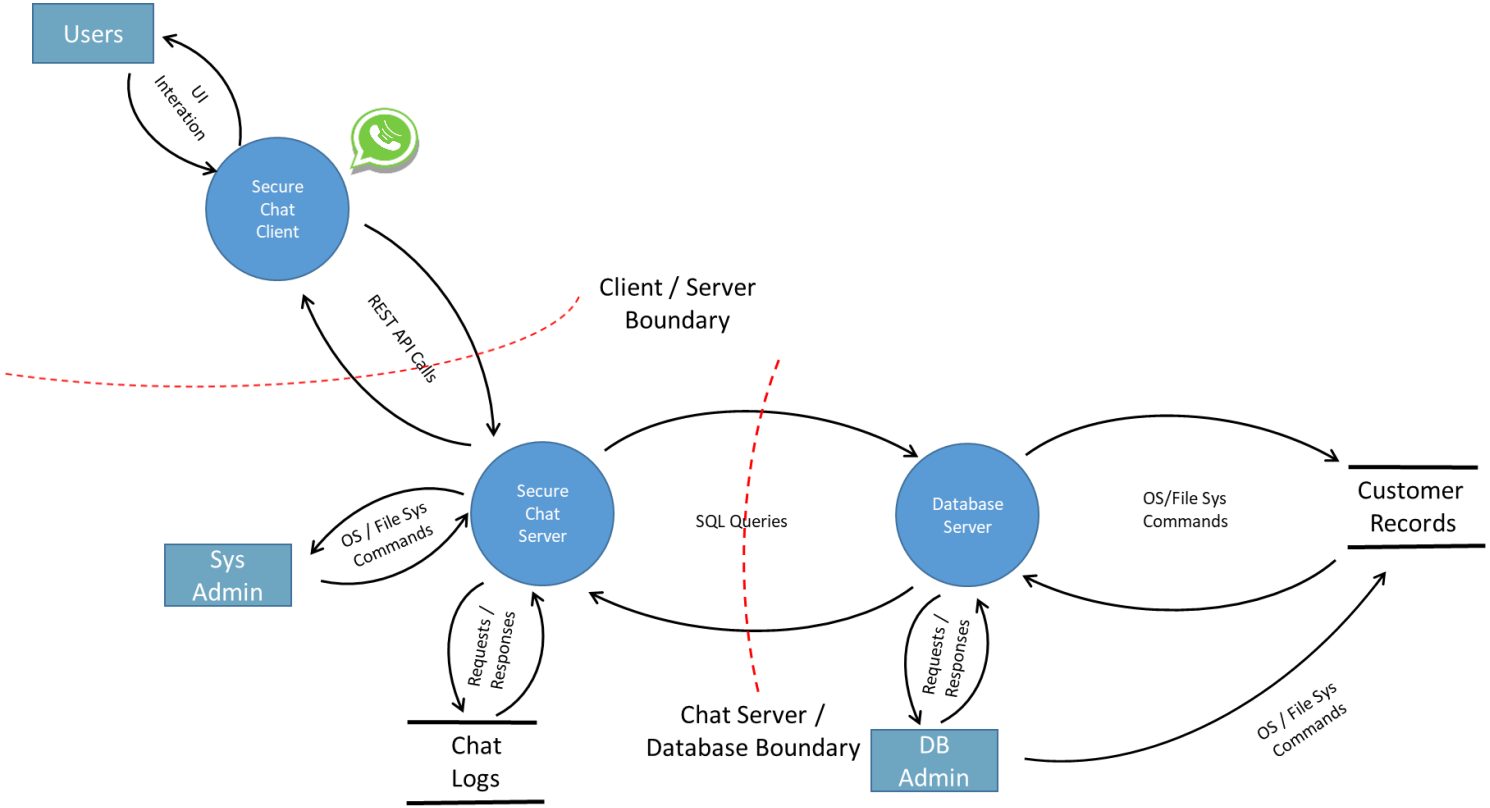
Create a DFD from the following requirements.

WhatsApp is developing a nextgen instant messaging program to be used for private use by its customers to allow them to chat privately with other users.

- “WhatsApp 2.0” (WA2) requires users to sign up with an account prior to using the system. After authenticating with a username and password, each user can message other users and expect their conversations to be private. Users have the ability to add/remove friends from their contact list, search for friends based on their email, block users from messaging them, and become “invisible” to all users on demand. Users can change their password or update their user profile within the application.
- Message archives and activity logs document user behavior and can be retrieved by the user or a WhatsApp Administrator through the application or by the administrative console, respectively.



Exercise #1- Sample Result



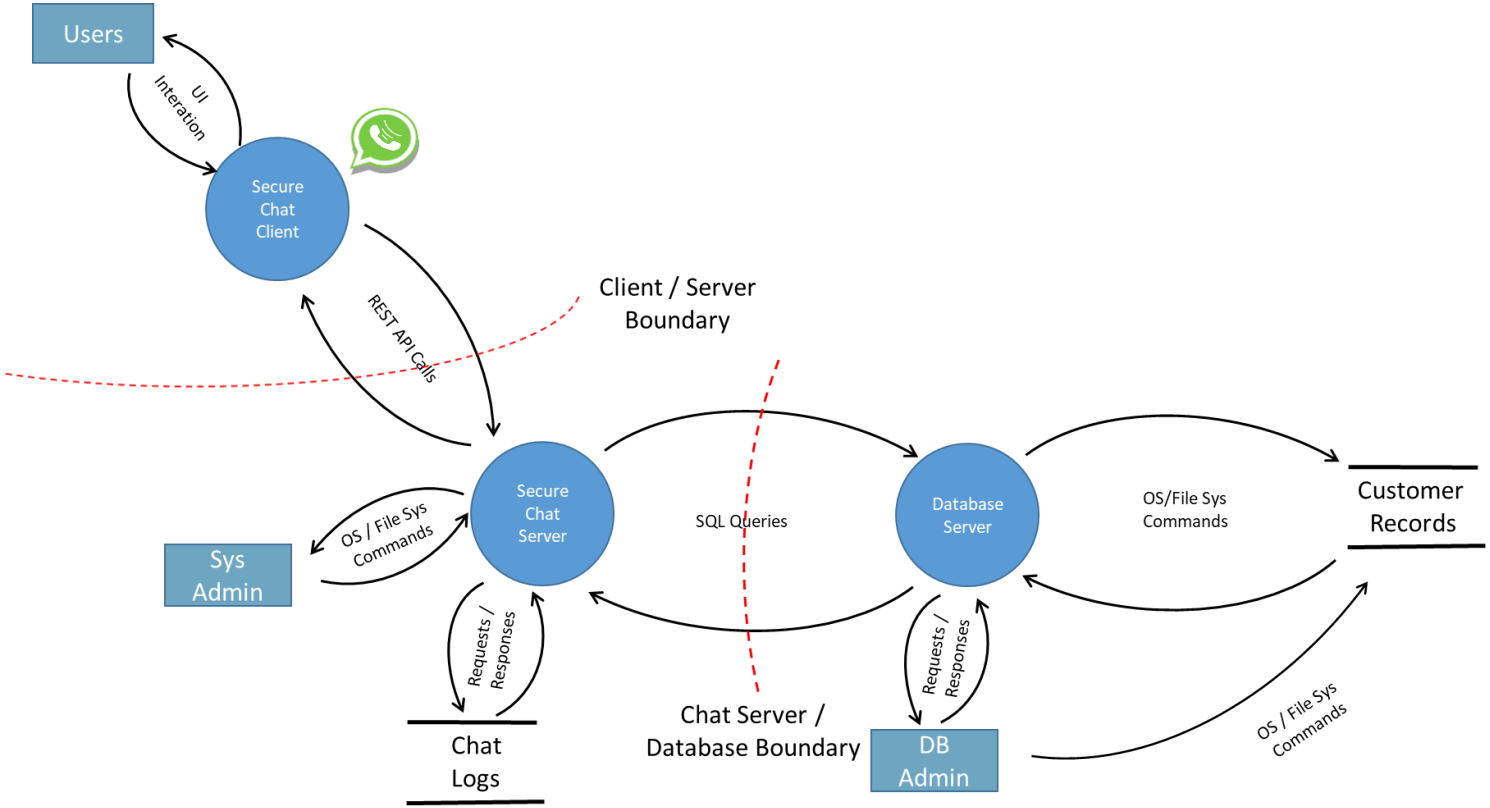
Exercise #2 - Data Classification

Determine which security principle is most important to protect (Confidentiality, Integrity, Availability, Authenticity, Non-reputiation)

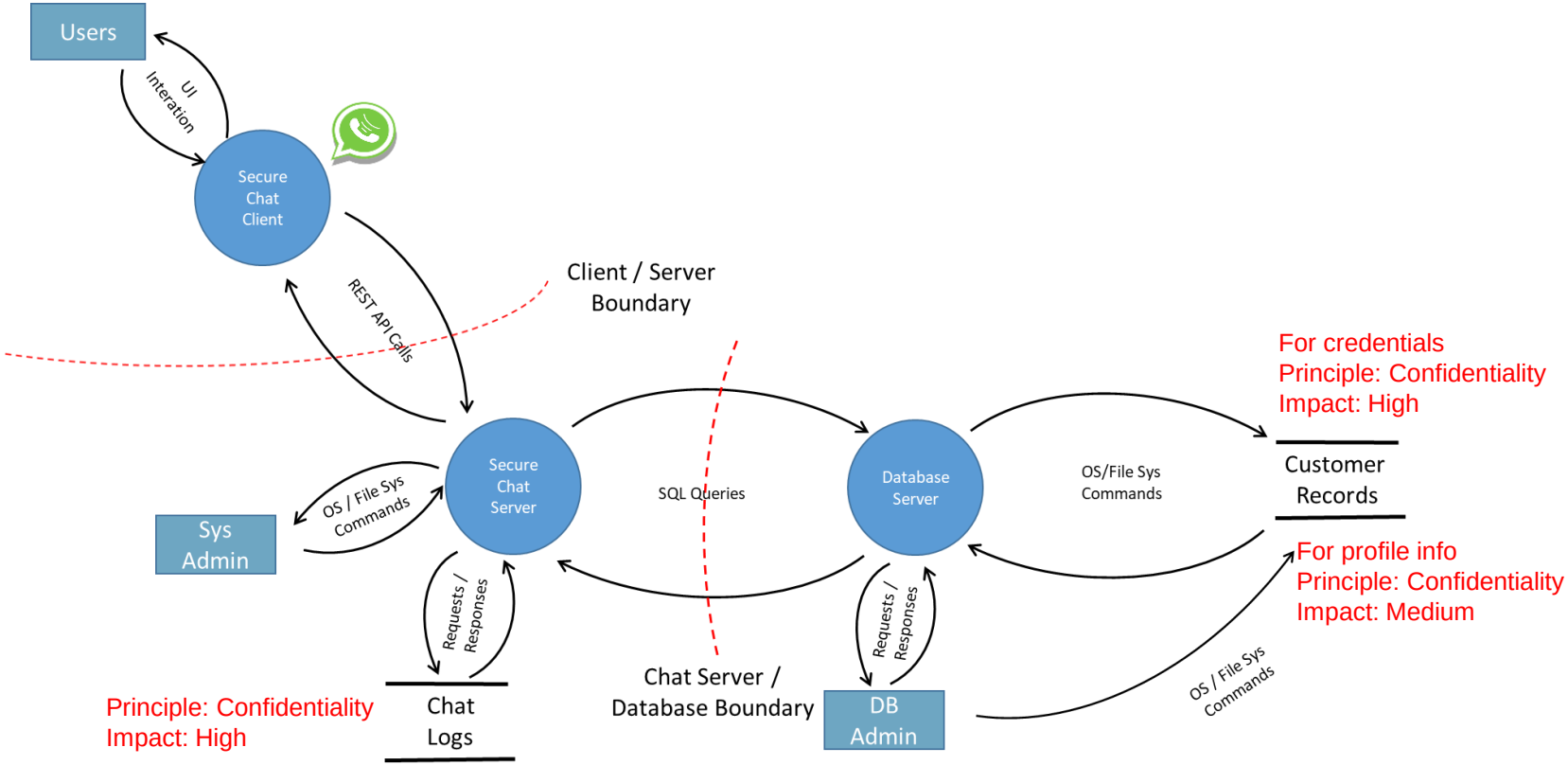
Classify the Impact of data breaches (High, Medium, Low)

Data	Security Principle(s)	Impact of Breach
User credentials		
Personal chat data		
Friends list		
Chat logs		
User profile info		

Exercise #2- Classify Data



Exercise #2- Sample Result



Threat Modeling Walkthrough

STRIDE

STRIDE is a threat modeling approach developed at Microsoft for identifying computer security threats. It provides a mnemonic for security threats in six categories.

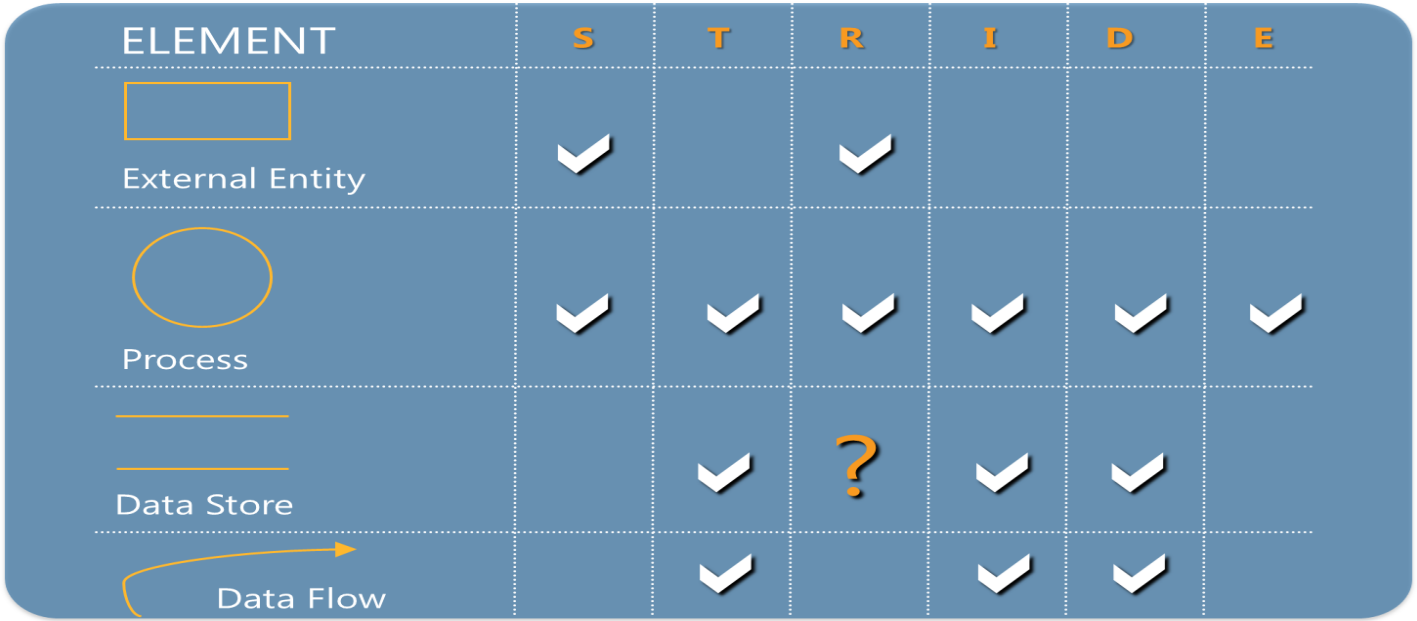
- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege

Threat Modeling Walkthrough

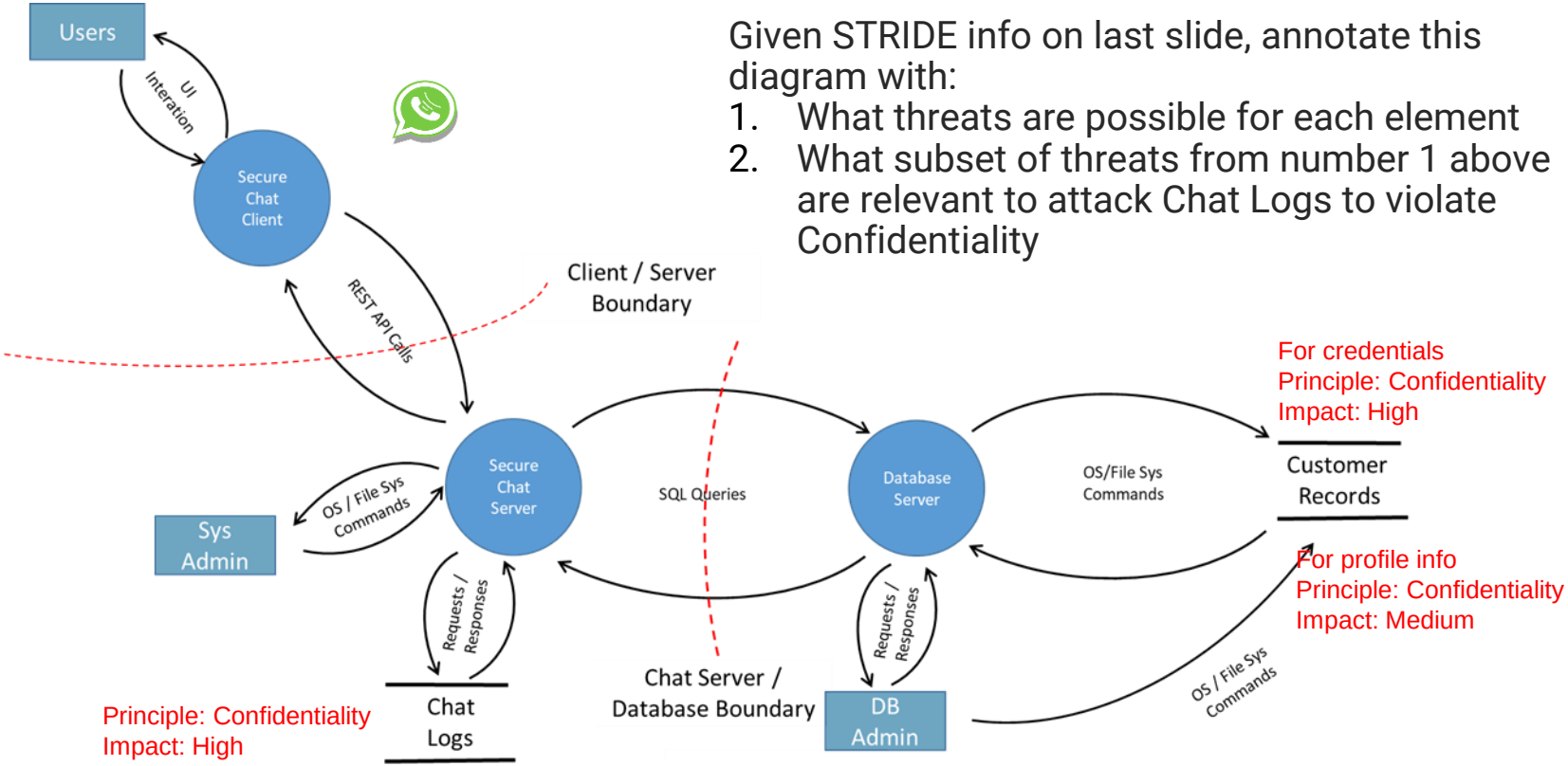
Threat	Property Violated	Definition	Example
Spoofing	Authentication	Impersonating something or someone else	Pretending to be any of Bill Gates, Paypal.com, or ntdll.dll
Tampering	Integrity	Modifying data or code	Modifying a DLL on disk or a packet as it traverses the network
Repudiation	Non-Repudiation	Claiming to have not performed an action	“I didn’t send that email,” “I didn’t modify that file,” “I certainly didn’t visit that website, dear!”
Information Disclosure	Confidentiality	Exposing information to someone not authorized to see it	Allowing someone to read the Windows source code; publishing a list of customers to a website
Denial of Service	Availability	Deny or degrade service to users	Crashing Windows or a website, sending a packet and absorbing seconds of CPU time, or routing packets into a black hole
Elevation of Privilege	Authorization	Gain capabilities without proper authorization	Allowing a remote internet user to run commands is the classic example, but going from a limited user to admin is also EoP

Threat Modeling Walkthrough

STRIDE Cheatsheet



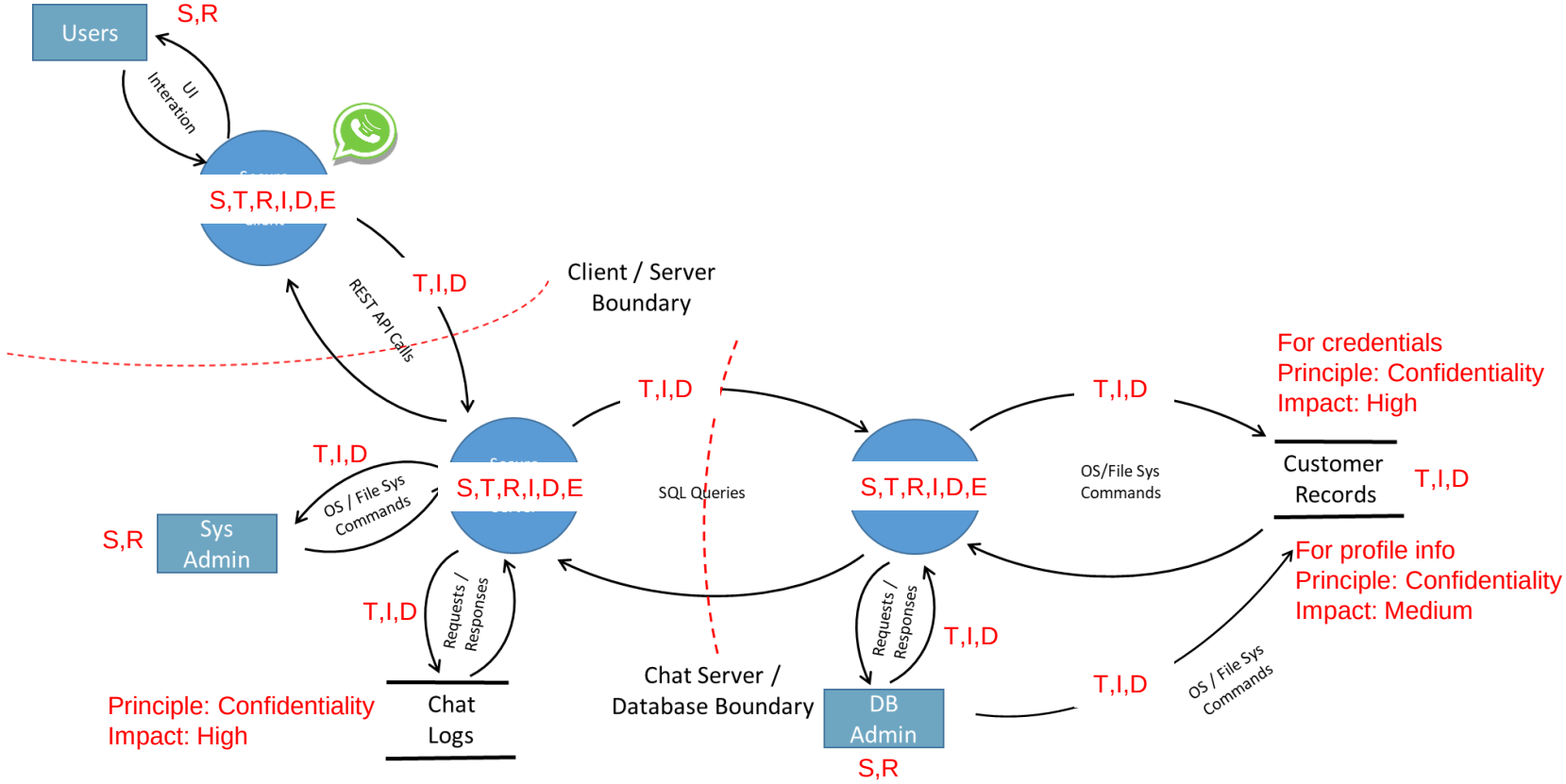
Exercise #3 – Annotate Diagram with Threats



Given STRIDE info on last slide, annotate this diagram with:

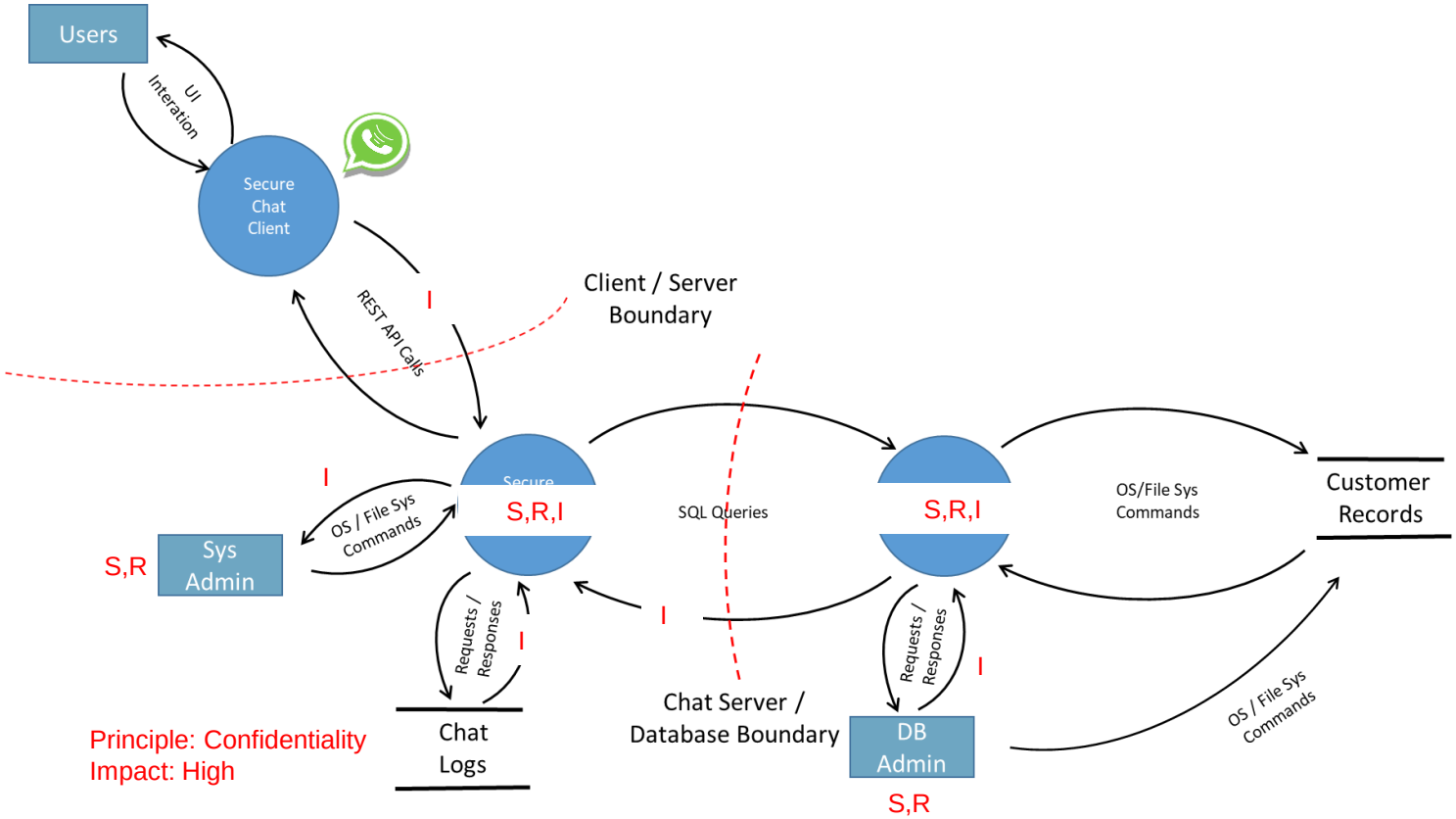
1. What threats are possible for each element
2. What subset of threats from number 1 above are relevant to attack Chat Logs to violate Confidentiality

Exercise #3- Result 1: Generic*



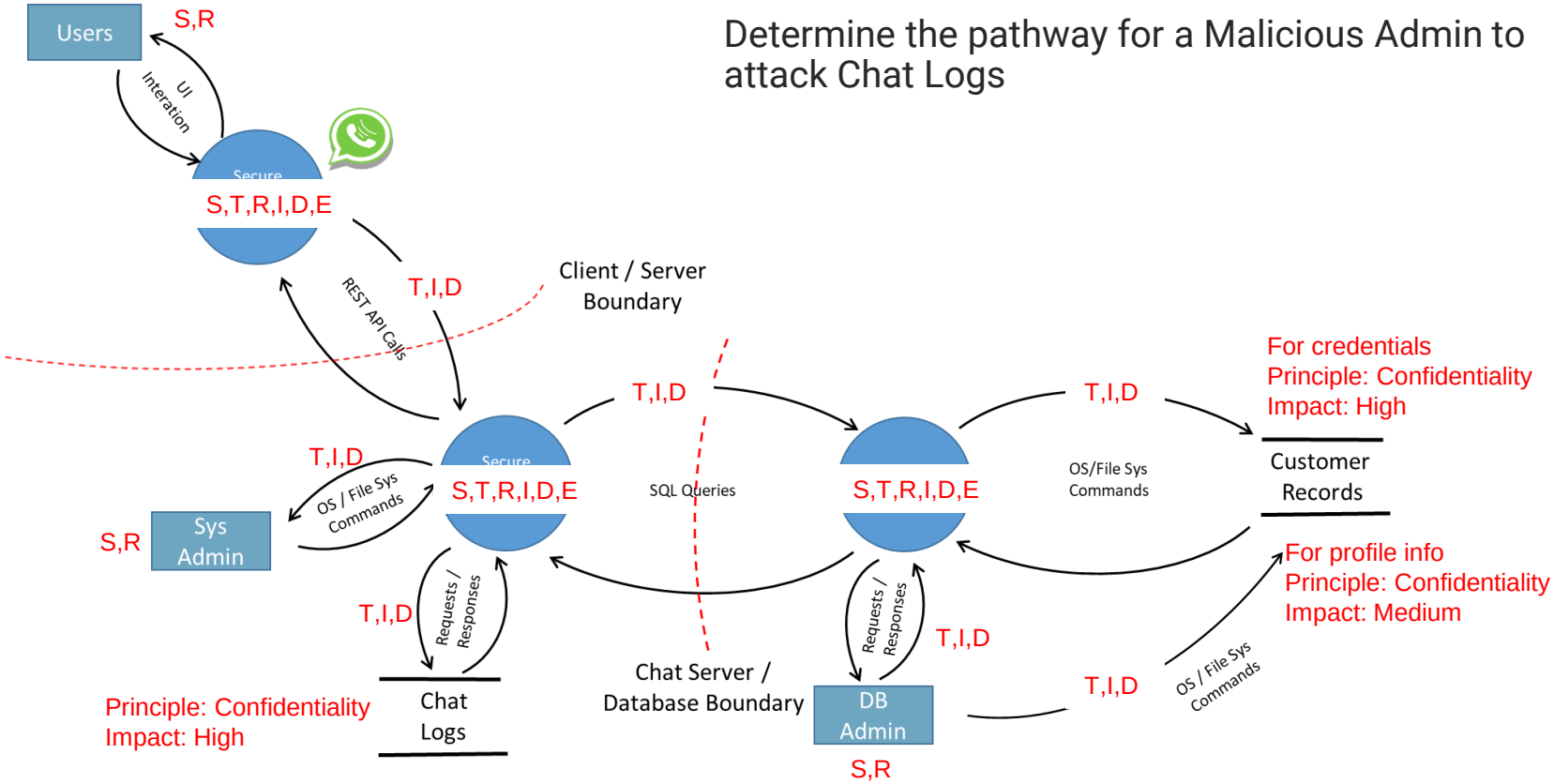
* Generic means STRIDE applied to all entities using STRIDE Cheat Sheet

Exercise #3- Result 2: Specific to chat logs

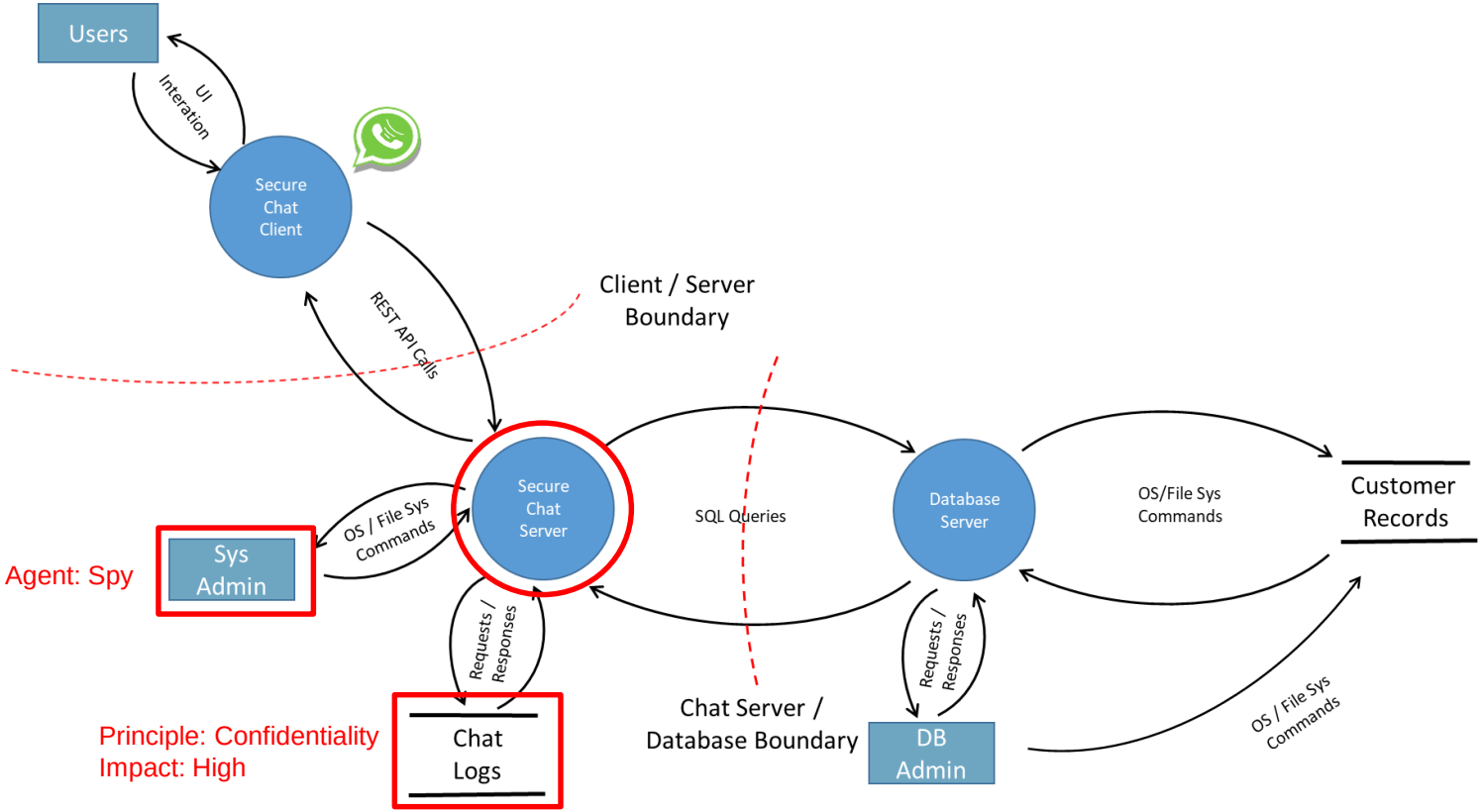


* Specific means STRIDE applied based on asset and security principle to protect

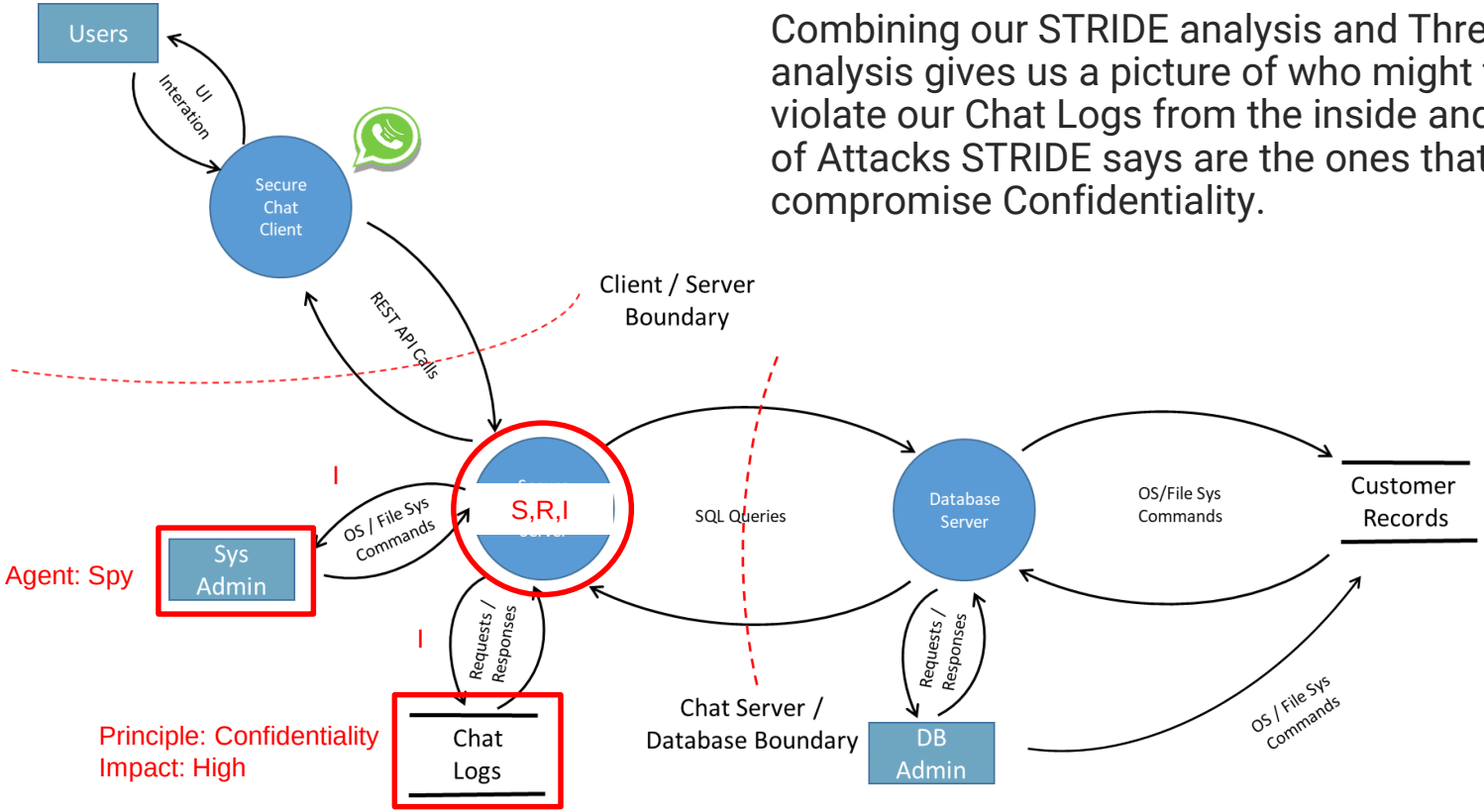
Exercise #4- Use Actors and Impact



Exercise #4- Sample Result



Exercise #4- Combo Result



Combining our STRIDE analysis and Threat Actor analysis gives us a picture of who might try to violate our Chat Logs from the inside and the types of Attacks STRIDE says are the ones that could compromise Confidentiality.

Threat Mitigations

Protect Confidentiality for Chat Logs

- Two-factor authentication to decrease risk of cracking credentials
- Encryption of chat data at rest older than 1 day
- Secure logging of system access and administrative commands
- Multi-login for admin access to chat logs