Microsoft Azure

Secure DevOps: Application Security Principles and Practices

Threat Modeling

Your name
Your Title
Microsoft

Module Overview

- Threat Modeling rationale
- Threat Modeling process
- Diagramming
- Threat enumeration and priority
- Mitigation and validation
- Lab – Model a simple Azure solution

Microsoft Azure

Threat Modeling rationale

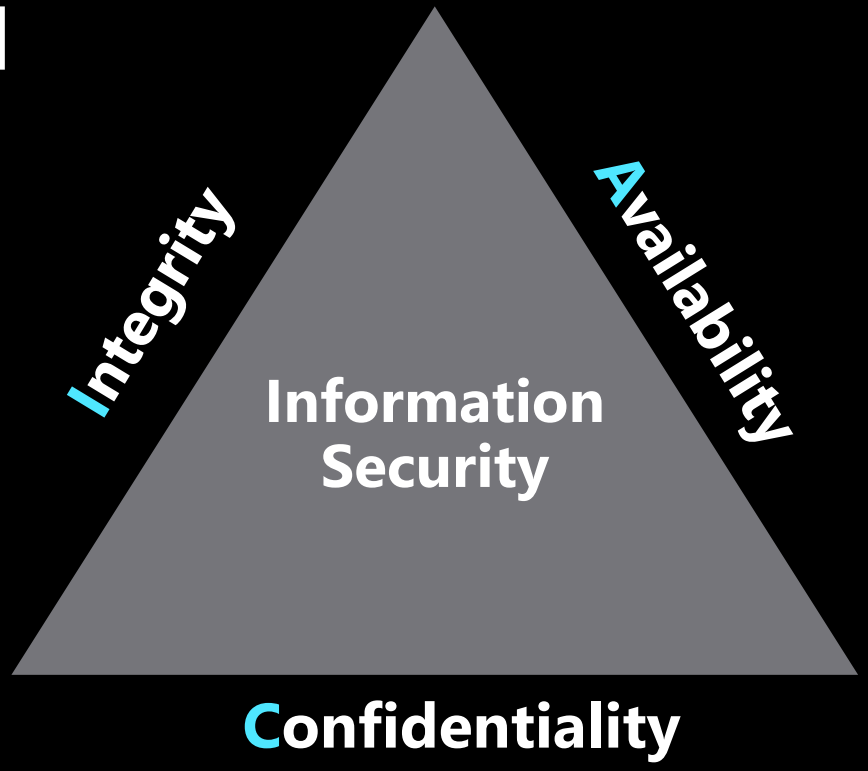
High value assets & information

"If you protect your paper clips and diamonds with equal vigour, you'll soon have more paper clips and fewer diamonds."

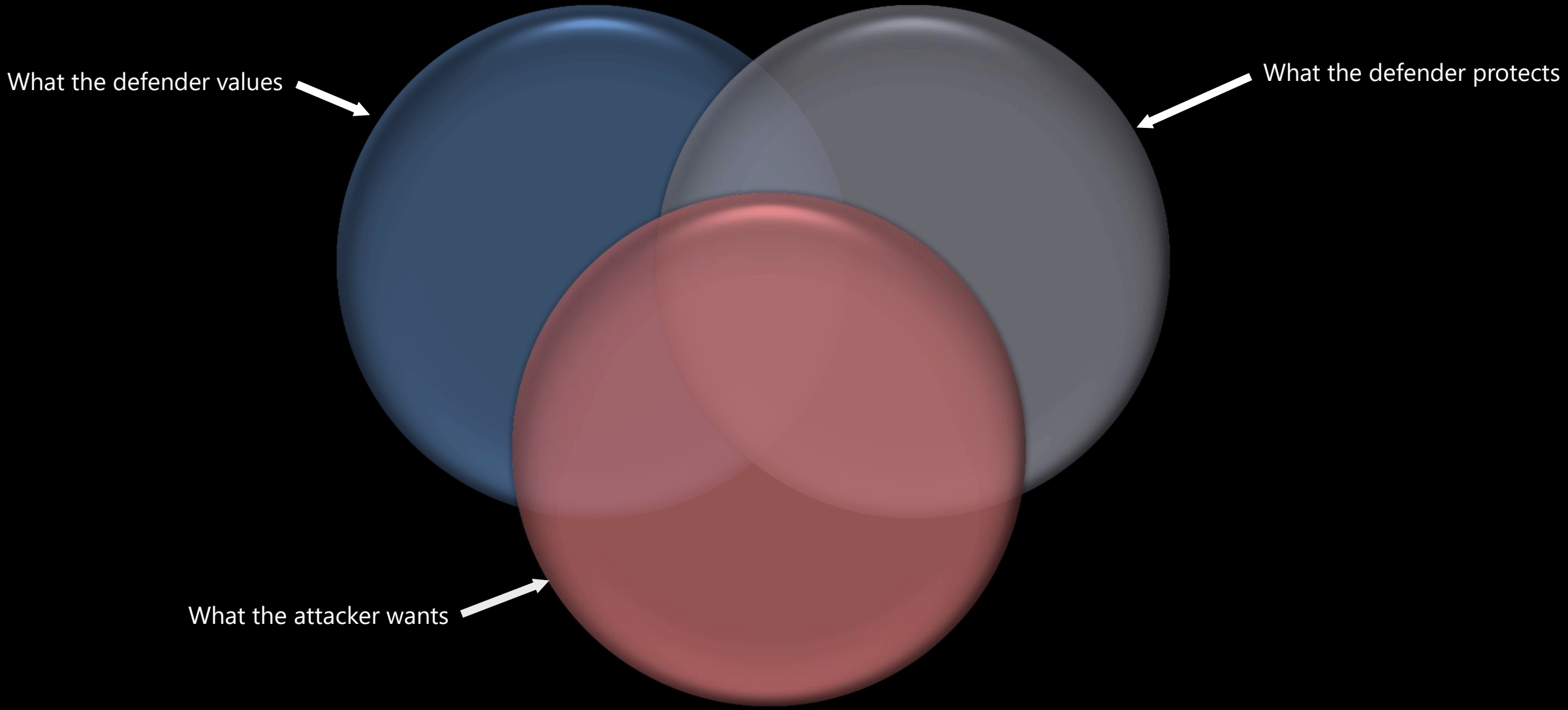
-Former US Secretary of State Dean Rusk

Core pillars of information security

CIA Triad



Security effectiveness



Why isn’t our security effective?

Asset misclassification

- Classifications not easily understood
- Implications of misclassification immense

Policies and controls not applied

- Dependent on users
- Tools not aligned to policy and a lack of automation
- Tooling reduces productivity, users bypass the processes

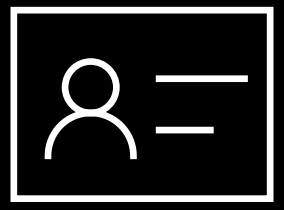
Inadequate reporting

- No consistent reporting across heterogeneous solutions
- Is it working or not?

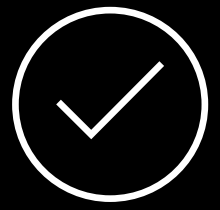
Impact cost rarely understood

- Applying classifications and protections costs time, money, and resources.
- Understanding lacking with regard to loss of one record vs aggregated records.

The properties of good systems

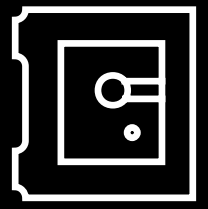


Authentication

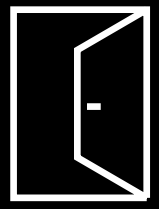


Integrity

Nonrepudiation



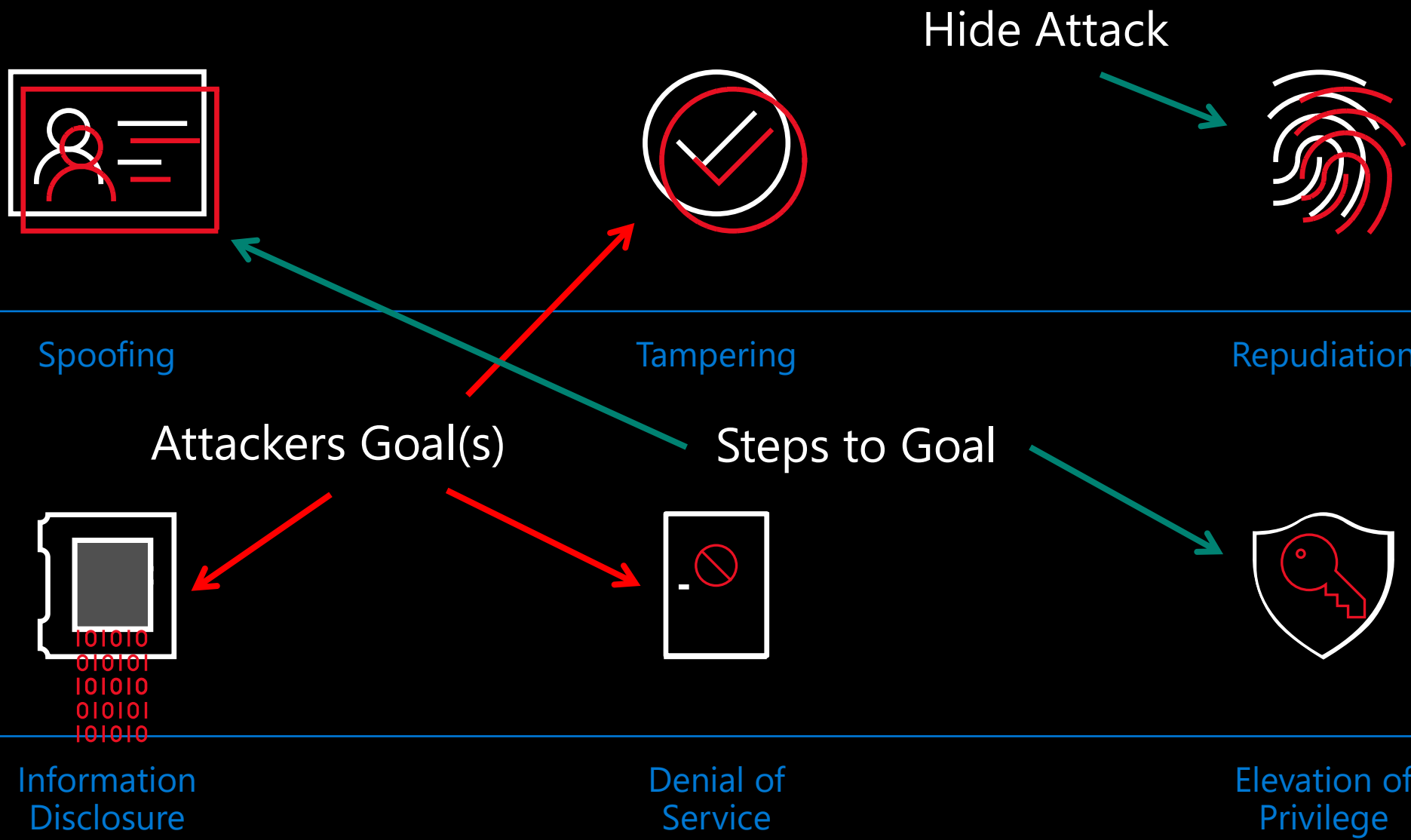
Confidentiality



Availability

Authorization

The properties of bad systems: STRIDE



Threat modeling

"If we had our hands tied behind our backs (we don't) and could do only one thing to improve software security… we would do threat modeling every day of the week"

Michael Howard
Senior Principal Cybersecurity Architect
Microsoft Corporation

Threat modeling

Threat modeling is a process to understand security threats to a system, determine risks from those threats, and establish appropriate mitigations.

- ✓ Let your Security Goals Surface
- ✓ Improves Awareness
- ✓ Reduces the Residual Risk
- ✓ Let you Align with the Organizational Risk Policy

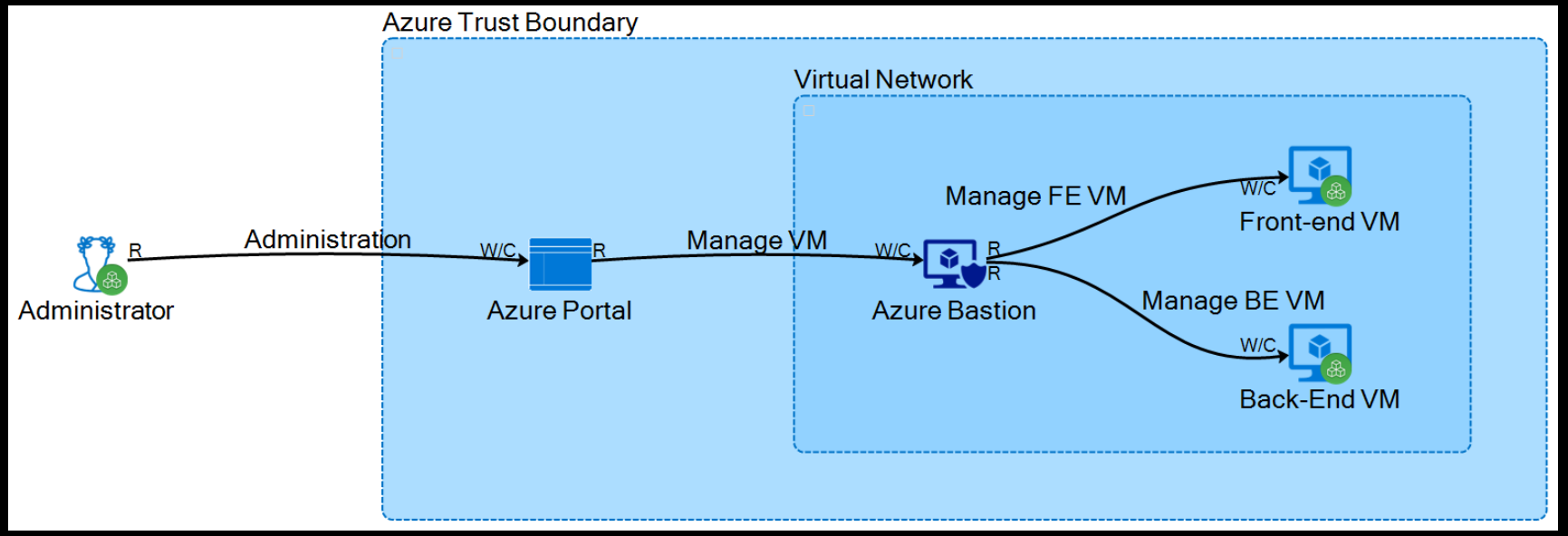
Microsoft Azure

Threat modeling process

Understand the solution

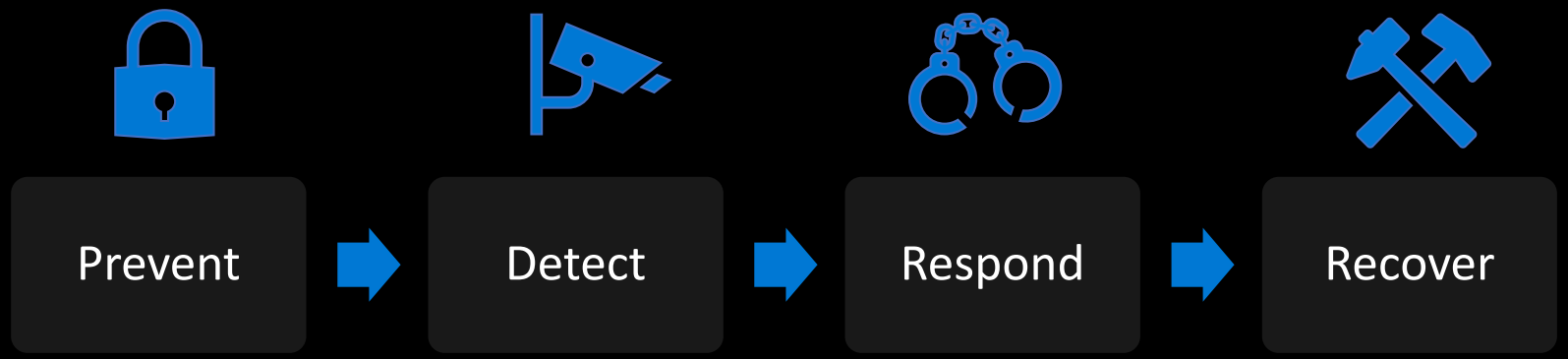
Understand the solution | Identify risks | Identify mitigations | Test mitigations | Implement mitigations | Select mitigations | Manage residual risk

- ✓ Review documents
- ✓ Interview architects and teams
- ✓ Deepen the knowledge on key topics



Identify mitigations

- ✓ Identify standard mitigations
- ✓ Search for alternatives, control variety
- ✓ Define a roadmap



Effective Threat Modeling meetings

Develop draft threat model before the meeting
- Use the meeting to discuss

Start with a DFD walkthrough

Identify most interesting elements
- Assets (if you identify any)
- Entry points/trust boundaries

Walk through STRIDE against those elements

Threats that cross elements/recur
- Consider library, redesigns

Microsoft Azure

Diagramming

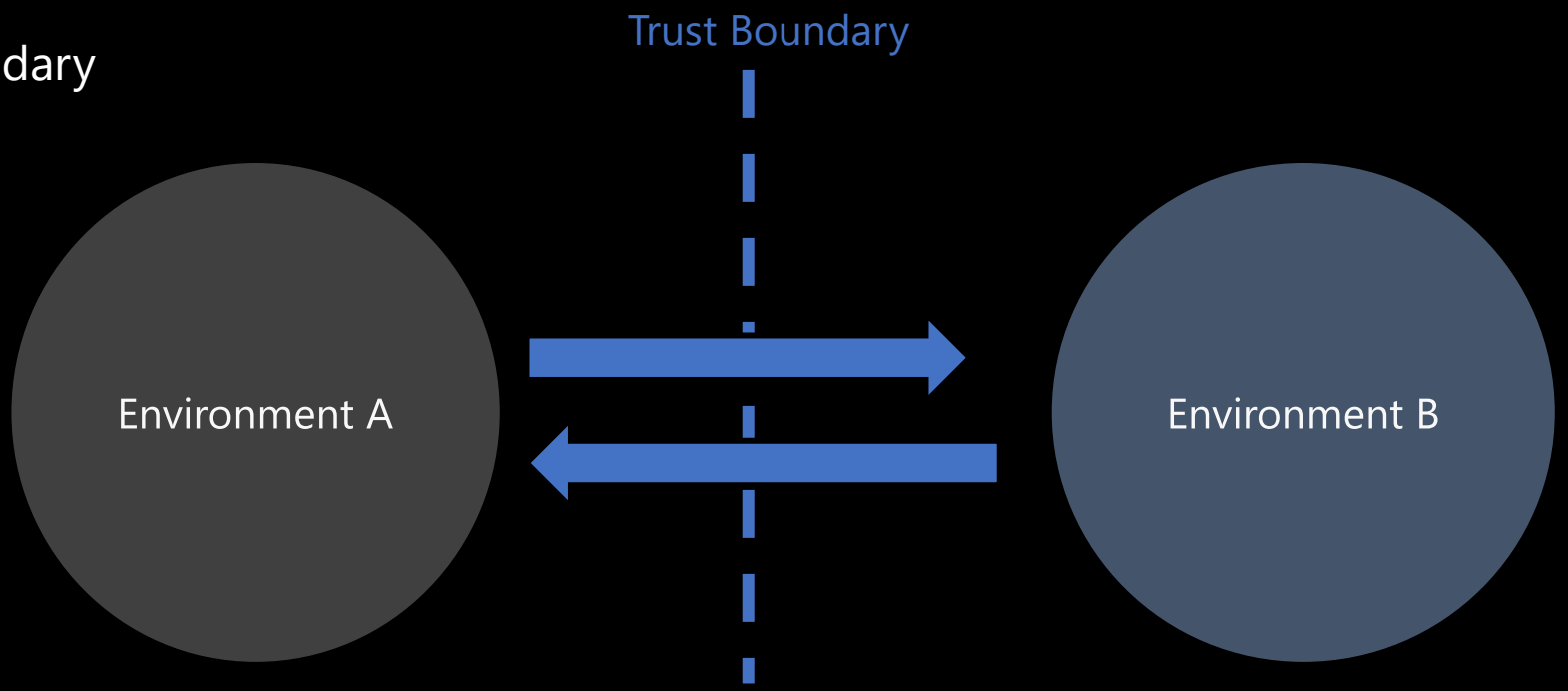
Isolation and trust boundaries

ALL communications MUST cross the Trust Boundary
- No secret communication path or data transit

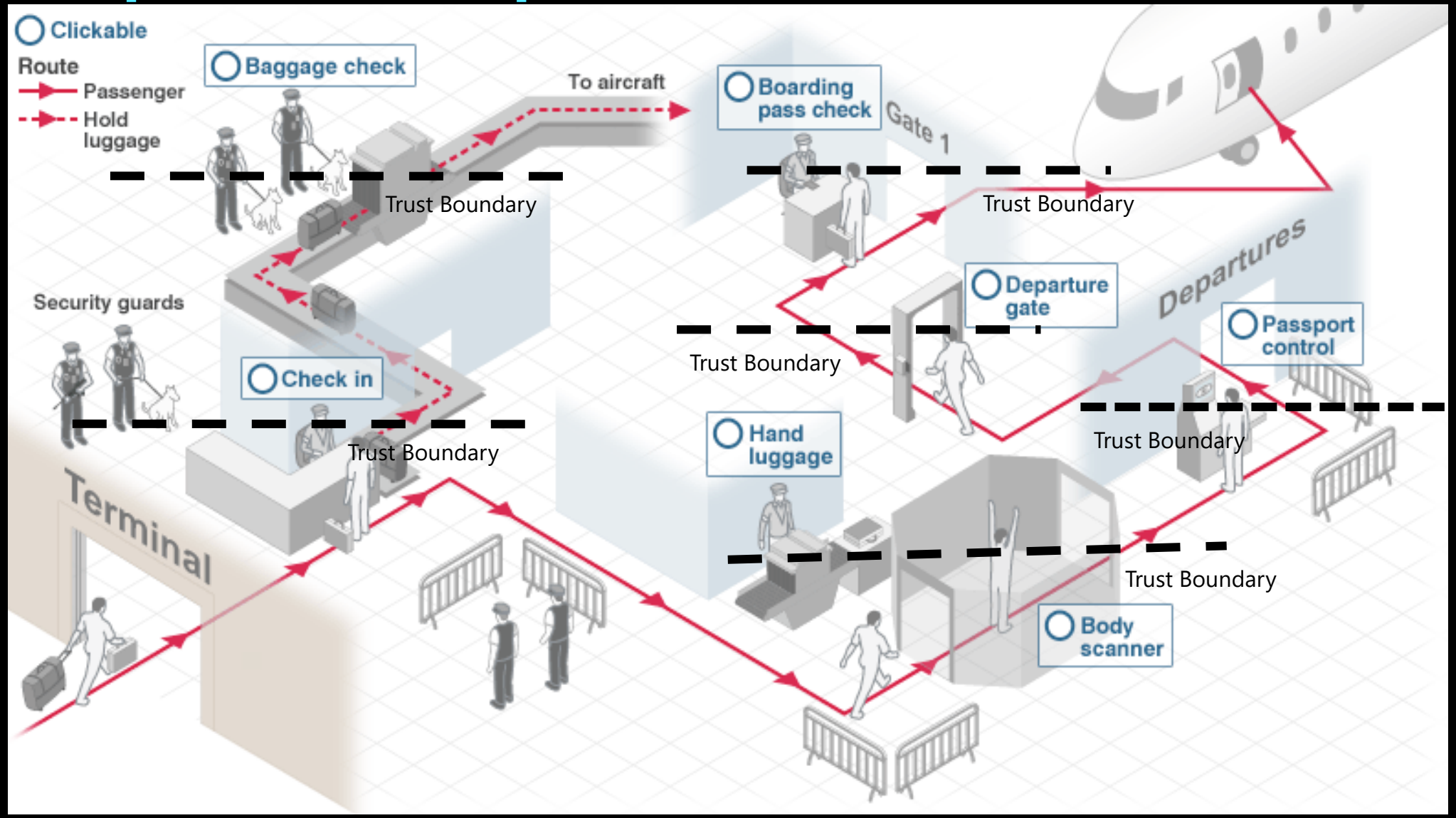
The Trust Boundary defines the policies of what is trustable
- For instance, airport security check

Ask STRIDE questions at the boundary

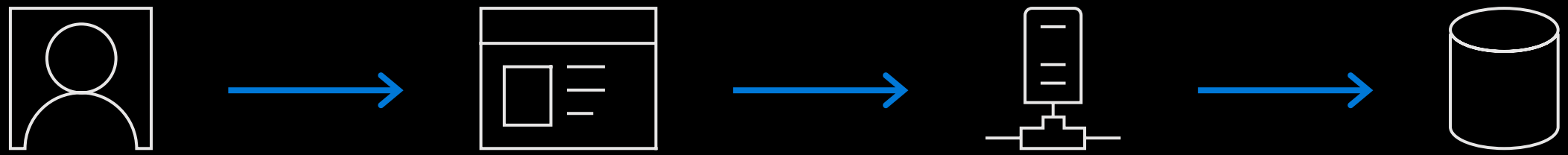
Spoofing
Tampering
Repudiation
Information Disclosure
Denial of Service
Elevation of Privilege



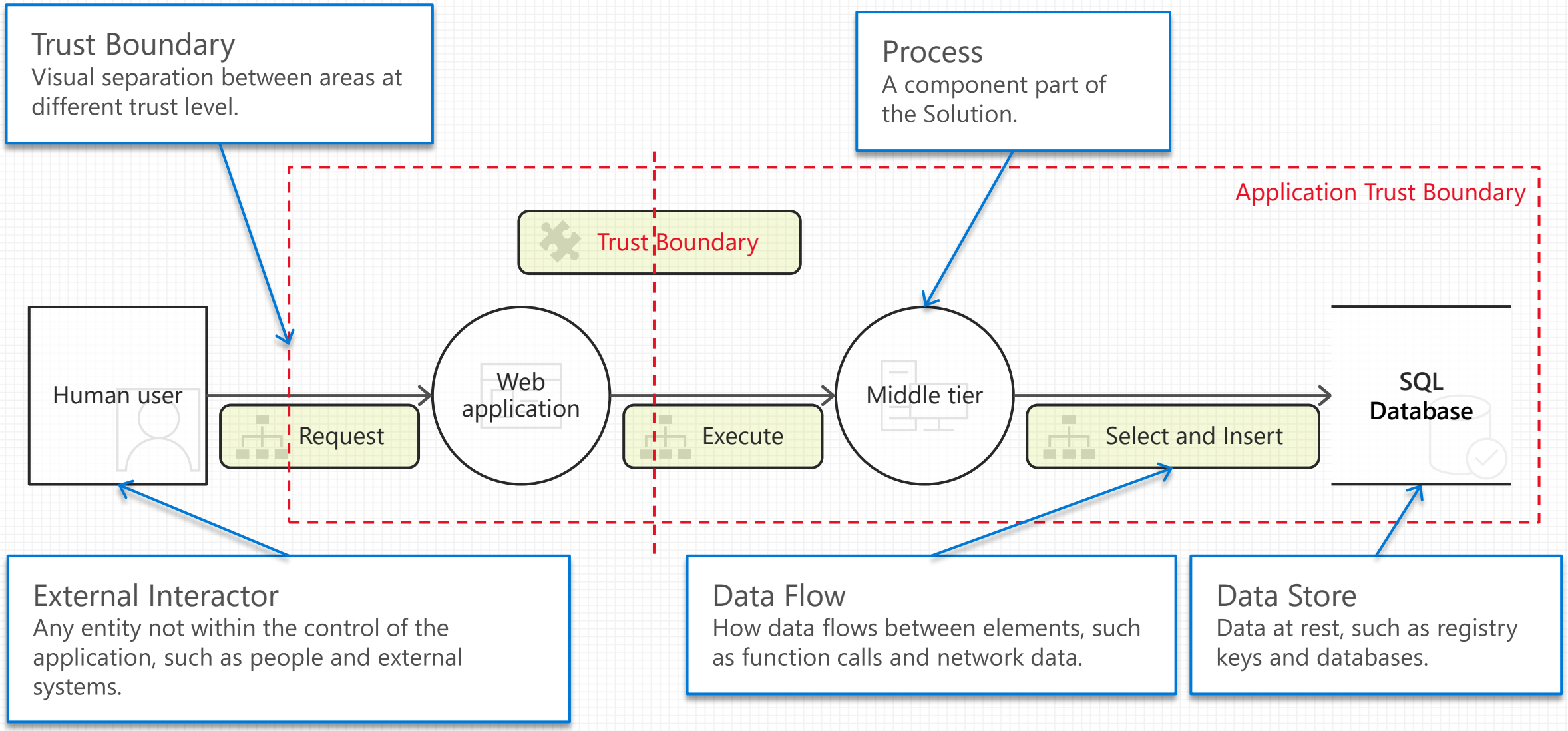
Airport example



Threat Modeling sample



Threat Modeling sample illustrated



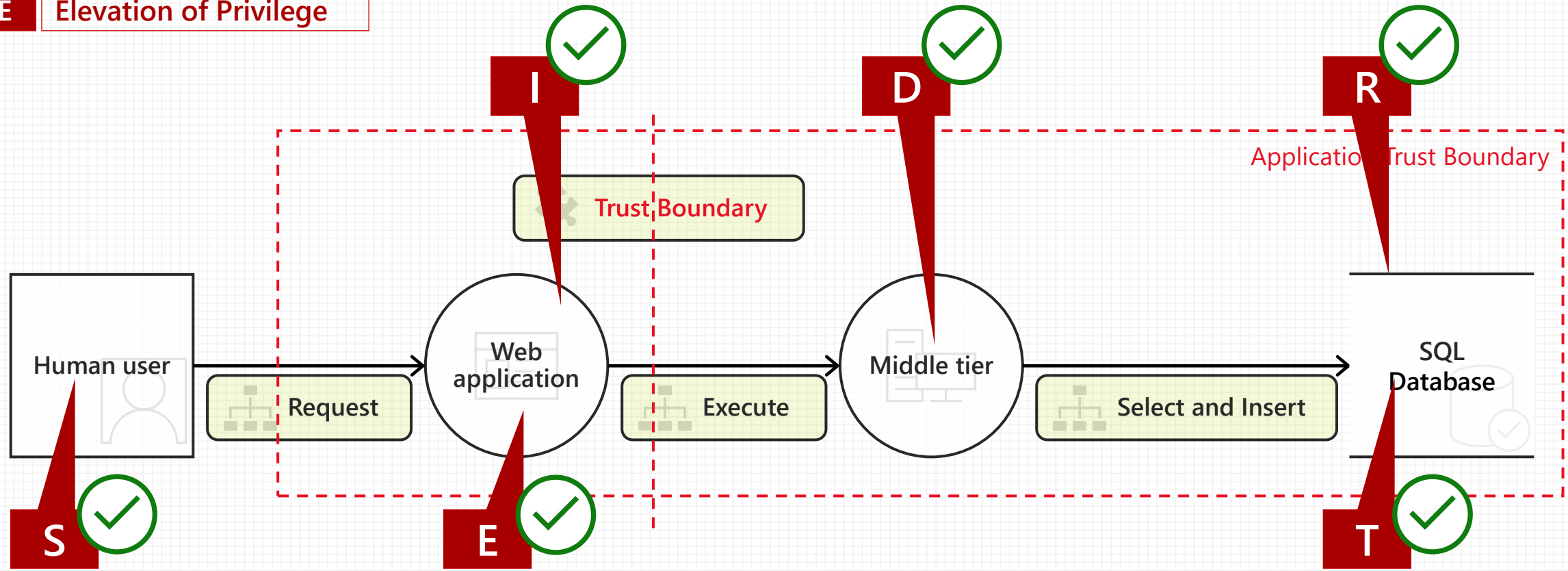
Threat identification

Experts:
Brainstorming and other informal methods.

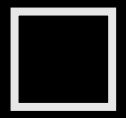
Experts and Non-Experts
Use STRIDE threat types.
Based on Microsoft Security Response Center (MSRC) issues and Common Vulnerability and Exposures (CVE) (see http://cve.mitre.org for more information).

Threat Modeling sample illustrated

S	Spoofing
T	Tampering
R	Repudiation
I	Information disclosure
D	Denial of service
E	Elevation of Privilege



Identifying STRIDE threats by element

Element	S	T	R	I	D	E
 External entity	✓		✓			
 Process	✓	✓	✓	✓	✓	✓
 Data Store	!	✓	!	✓	✓	✓
 Data Flow		✓		✓	✓	

✓ Threat

! Depends on store

Common mistakes to avoid

Missing interactors
They represent the highest risk and the Threat Model must include them all.
For example: *"Who reads the Audit DB?"*

Missing data sources
If the application consumes data, it should come from somewhere.
For example: *"What is the source used for pre-populated DBs?"*

Missing data processors
Data cannot flow between databases without the help of some logic.
For example: *"Are Jobs or an ETL process built with some SSIS package?"*

Practical STRIDE

Per Flow
- Spoofing of any endpoint; source and destination (including replay attacks)
- Tampering and Information Disclosure in transit and/or with malicious input
- Repudiation by any participant (depends on trust level)
- Denial of Service – physical or from the source, rarely from destination
- Elevation of Privilege with malicious input or information from previous actions (graph thinking)

Processes
- Direct Tampering and Information disclosure on end-user device
- Tampering, Information Disclosure due to weak RBAC
- Denial of Service and Information Disclosure through sloppy error handling
- Secret Handling and Cryptography

Data Stores
- Tampering and Information Disclosure
- Denial of Service due to resource exhaustion
- Elevation of privilege due to weak AuthZ (ACLs, RBAC)
- Elevation of privilege due to injection attacks

Distractions

Do not worry about these threats (unless you are modeling them specifically)

- The computer is infected with malicious software (malware).
- Someone removed the hard drive and tampers
- Admin is attacking user
- A user is attacking himself

How to prioritize threats

Do the easy fixes first.
«Bug Bar» method
CVSS – for some industries https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator
DREAD (Risk = Damage + Reproducibility + Exploitability + Affected Users + Discoverability)
«Delphi Oracle» method

A bug bar is a quality gate that applies to the entire software development project and defines the severity thresholds of security vulnerabilities.

The bug bar, once set, should never be relaxed.

Microsoft Azure

Mitigation and validation

Address identified threats

Mitigations (in order of preference)

- Redesign
- Standard mitigations
- Invented mitigations (You are not an expert)

Business Decision

- Transfer risk (within) policy
- Accept residual risk (within) policy

Identify External Responsibilities.

Examples of standard mitigations

Threat	Example Standard Mitigations	
Spoofing	Multi-Factor Authentication (MFA) Digital signatures	Message authentication codes OpenID Connect
Tampering	ACLs Digital signatures	Message Authentication Codes
Repudiation	Strong Authentication	Secure logging and auditing
Information Disclosure	Encryption	ACLs, RBAC
Denial of Service	ACLs Quotas	High availability designs
Elevation of Privilege	ACLs Group or role membership	Input validation

Identify external responsibilities

Examples:

- Azure Services from Microsoft.
- Third-party Software as a Service (SaaS) solutions

What is safe to assume (maybe):

- Implemented and tested securely
- Patched regularly against known vulnerabilities

What is not safe to assume:

- Configured securely

Validating quality of threats and mitigations

Good Threats describe

- The attack
- The context
- The impact

Good Mitigations

- Associate with a threat
- Describe the mitigations
- File a bug or work item.

Filing bugs or work item make the Threats actionable!

Lab – Model a simple Azure solution