Threat Modeling

JAX 2016

Dominik Schadow | bridgingIT

OWASP Top 10 2013

(1) Injection

(2) Broken Authentication and Session Management

(3) Cross-Site Scripting

(4) Insecure Direct Object References

(5) Security Misconfiguration

(6) Sensitive Data Exposure

(7) Missing Function Level Access Control

(8) Cross-Site Request Forgery

(9) Using Components with Known Vulnerabilities

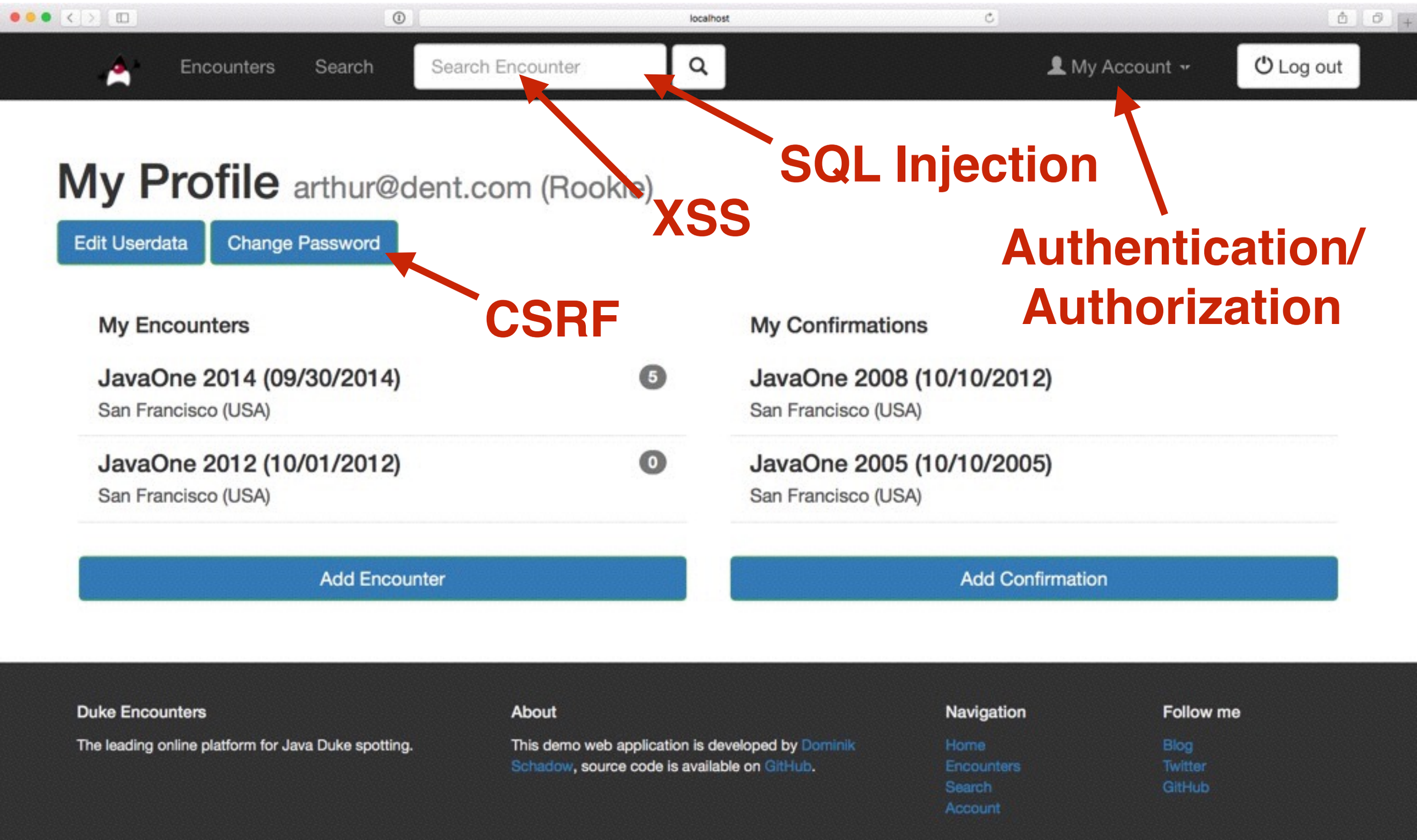
(10) Unvalidated Redirects and Forwards

What a the threats?

Java 8
Spring Boot 1.3 & Spring Security 4
Thymeleaf 2.1, Bootstrap 3.3
Tomcat 8
MySQL 5 (user & application data)

What a the threats?

What a the threats?



We developers tend to focus on typical programming errors (like SQL Injection and Cross-Site Scripting) and ignore the underlying flaws.

Agenda

Threat Modeling **Basics**

Identifying Threats in Applications

Threat Modeling in **Action**

Threat Modeling Basics

Security flaws are introduced early in the development lifecycle, with no code developed yet. And still time to change the application in case of threats.

Different ways to threat model

Identify security flaws early

Focus on attackers: Can you really think like an attacker?

Focus on assets: What is an asset in your application? How do you link them to threats?

Attacks tend to follow the data flow

Follow the data

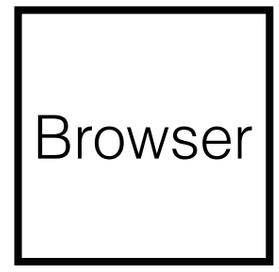
Focus on the system under development

Start with external entities - events which drive activity like a click in the browser.

Data Flow Diagrams

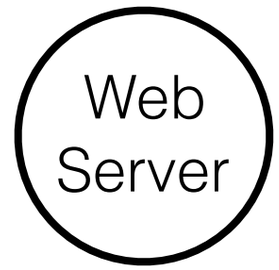
External Entity

Users or code outside the control of the application



Process

Any running code (app or code within one app)

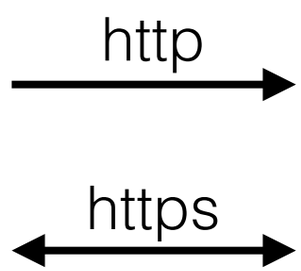


Data Store

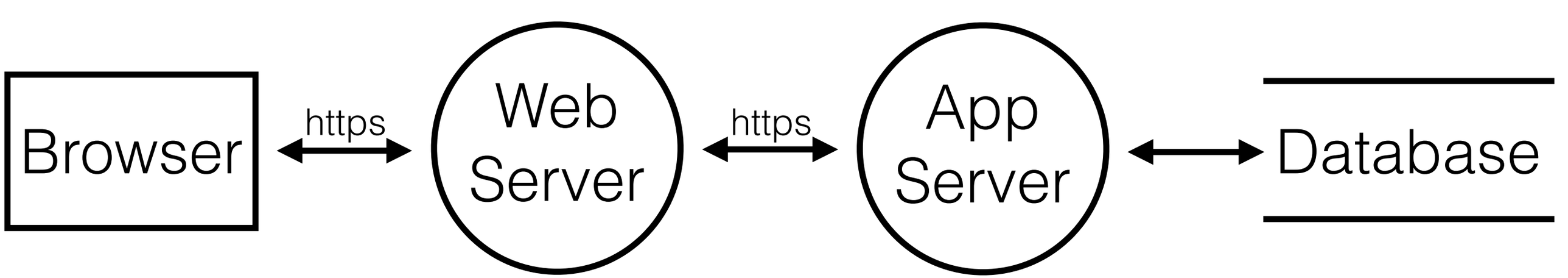
Things that store data

Data Flow

Communication between elements (data and method calls)



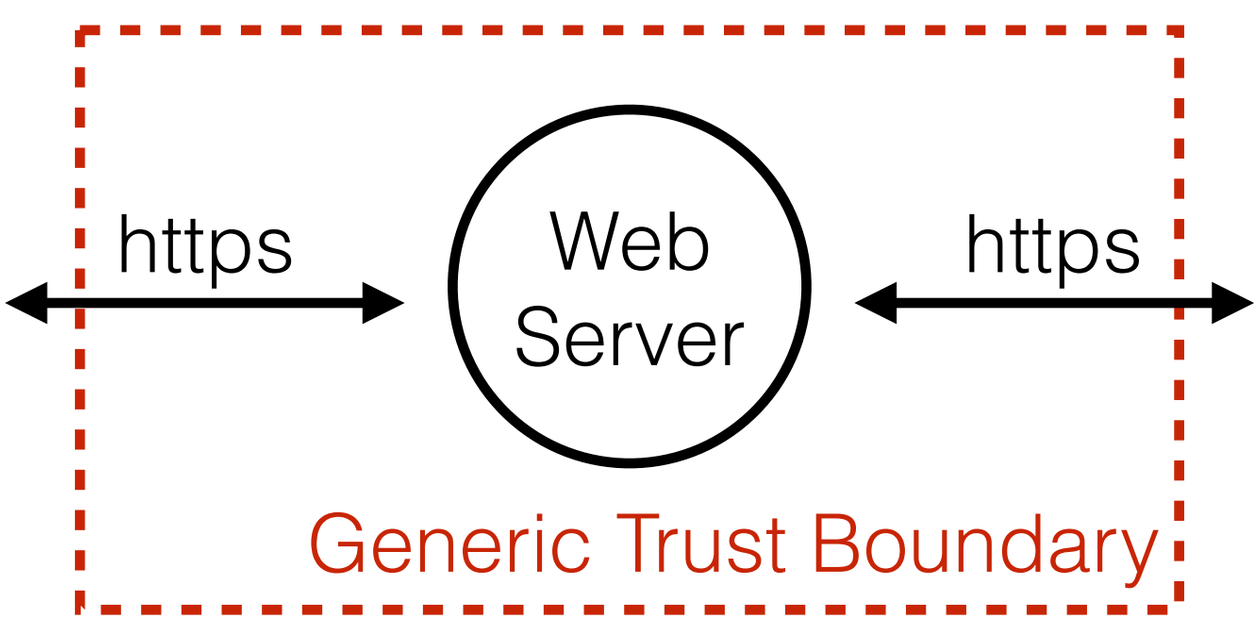
Follow the data

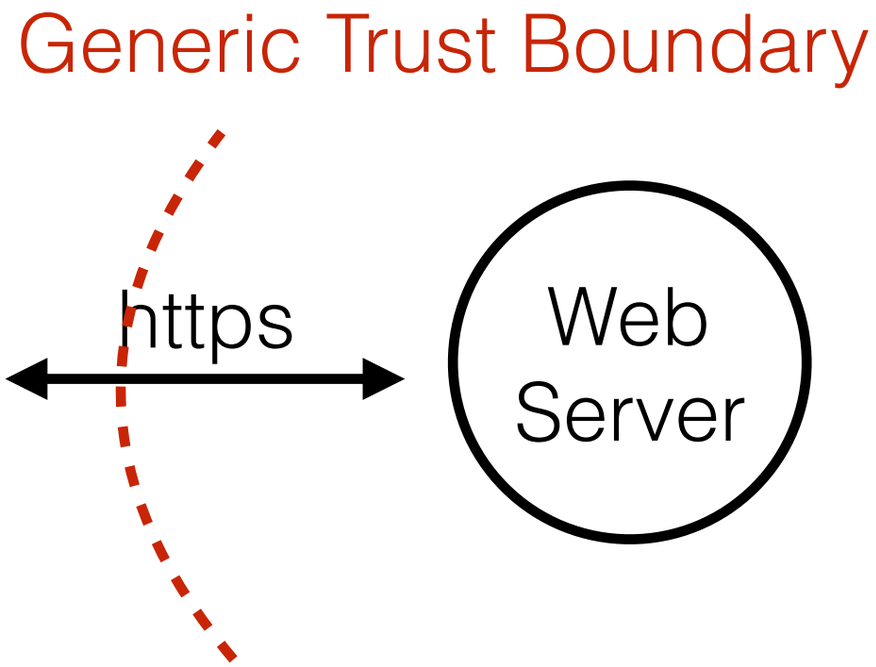


Trust Boundaries

Trust Boundary

Where entities with different privileges interact - trust everything inside



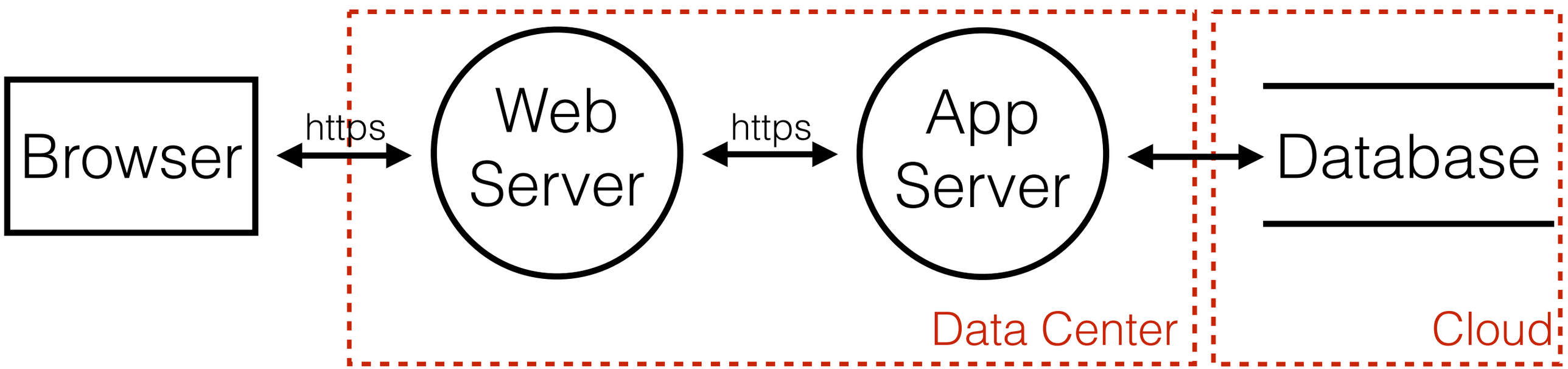


Drawing boundaries

Follow the data flow

Start on one side and add a boundary every time the privilege level changes (web server, database, …).

Add trust boundaries



Typical boundaries

Technical or organizational boundaries

Networks, Servers, VMs, Firewalls,
Departments, Data Centers, Clouds, …

Everything embedded

There is always one boundary

Everything in the system has same level of privilege and has access to everything in the system.

Identifying Threats in Applications

Developers easily model the whole application with all entities, but are having trouble to identify threats. Start with what you know and complete it step by step.

Ask yourself (and others)

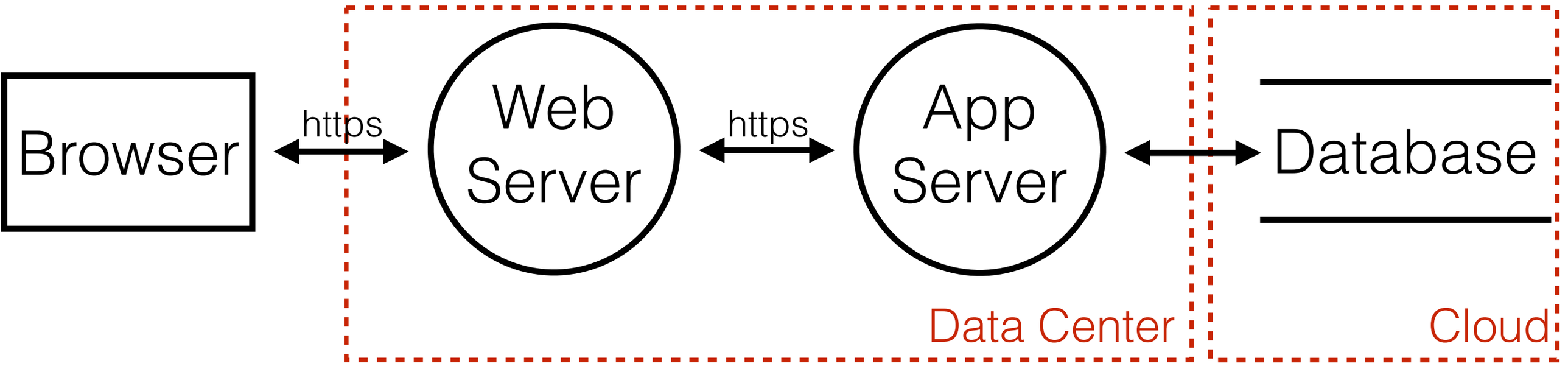
Repeat until you are satisfied

1. What are you building?
2. What can go wrong?
3. What should you do about those things that can go wrong?

What are you building?

Follow the data flow

„Sometimes“ indicates alternatives: model all
No data sinks: show the consumers
Data does not move by itself: draw the process that moves it


Browser
https
Web Server
https
App Server
Database
Data Center
Cloud

What can go wrong?

Start with data crossing boundaries

Brainstorming with technology experts
Elevation of Privilege game
STRIDE

STRIDE

Focus on threat, not on category

Spoofing, **T**ampering, **R**epudiation,
Information Disclosure, **D**enial of Service,
Elevation of Privilege

STRIDE

Spoofing — Pretending to be something or somebody else
Violates: **Authentication**

Tampering — Make unauthorized modifications (disk, memory, network)
Violates: **Integrity**

STRIDE

Repudiation Claiming that someone didn't do something
Violates: **Non-Repudiation**

Information **Disclosure** Exposing information to someone not authorized
Violates: **Confidentiality**

STRIDE

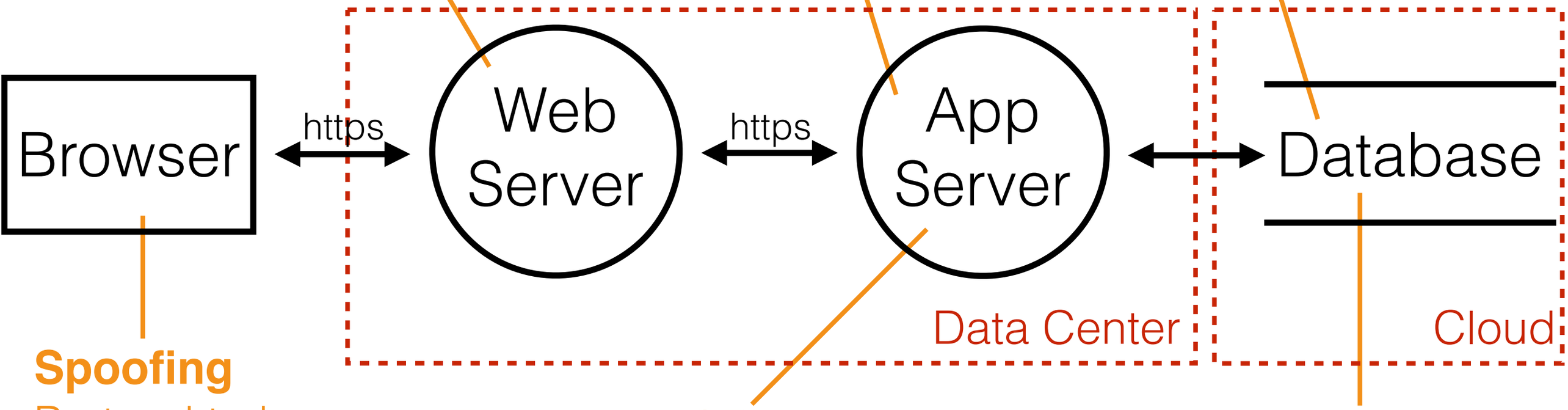
Denial of Service — Absorbing resources needed to provide service
Violates: **Availability**

Elevation of Privilege — Doing something someone is not authorized to do
Violates: **Authorization**

Denial of Service
Flood server

Repudiation
Execute actions in the name of other user

Tampering
Manipulate data



Spoofing
Pretend to be another user

Elevation of Privilege
Access backend logic directly

Information Disclosure
Dump database

Code injections

Should be mitigated by a framework

Cross-Site Scripting, Cross-Site Request Forgery and SQL Injection should be mitigated automatically by the chosen framework.

Identify threats in meetings

Document all identified threats

Add any threat to the bug tracker and tag it as security bug. Document how to deal with it.

Movie Plot Threats

Fun to discuss
Not really helpful
Focus on realistic ones

Address each threat

Decide as early as possible

Either mitigate, eliminate, transfer or accept a threat.

Mitigate it

Preferred solution

Do something to make it harder to take advantage of a threat.
e.g. *introducing a password policy*

Eliminate it

Most secure solution

Usually results in feature elimination.
e.g. *removing admin functionality*

Transfer it

Team solution

Someone/ something else handles the risk (make sure they actually do).
e.g. *operations adding a Web Application Firewall*

Accept it

Last resort solution

Stop worrying about it and live with the risk.

e.g. *a secret service subverting one of your employees*

Threat Target	Mitigation Strategy	Mitigation Technique	Prio	ID
Repudiating actions	Log	Log all relevant actions in audit log	2	101
Spoofing a user	Identification and authentication	Password policy, token, password reset process	1	179
Network flooding	Elastic cloud	Dynamic cloud resources to auto scale	3	16
Tampering network packets	Cryptography	HTTPS/TLS	1	10

Is it complete?

Checks show you are not done, but none shows you are

Easy: STRIDE completely in the diagram
Harder: One threat per diagram element
The truth: You are never done

Breadth before depth

Threat model the whole application

Make sure to threat model all features whose failure have security or privacy implications and all features that cross trust boundaries.

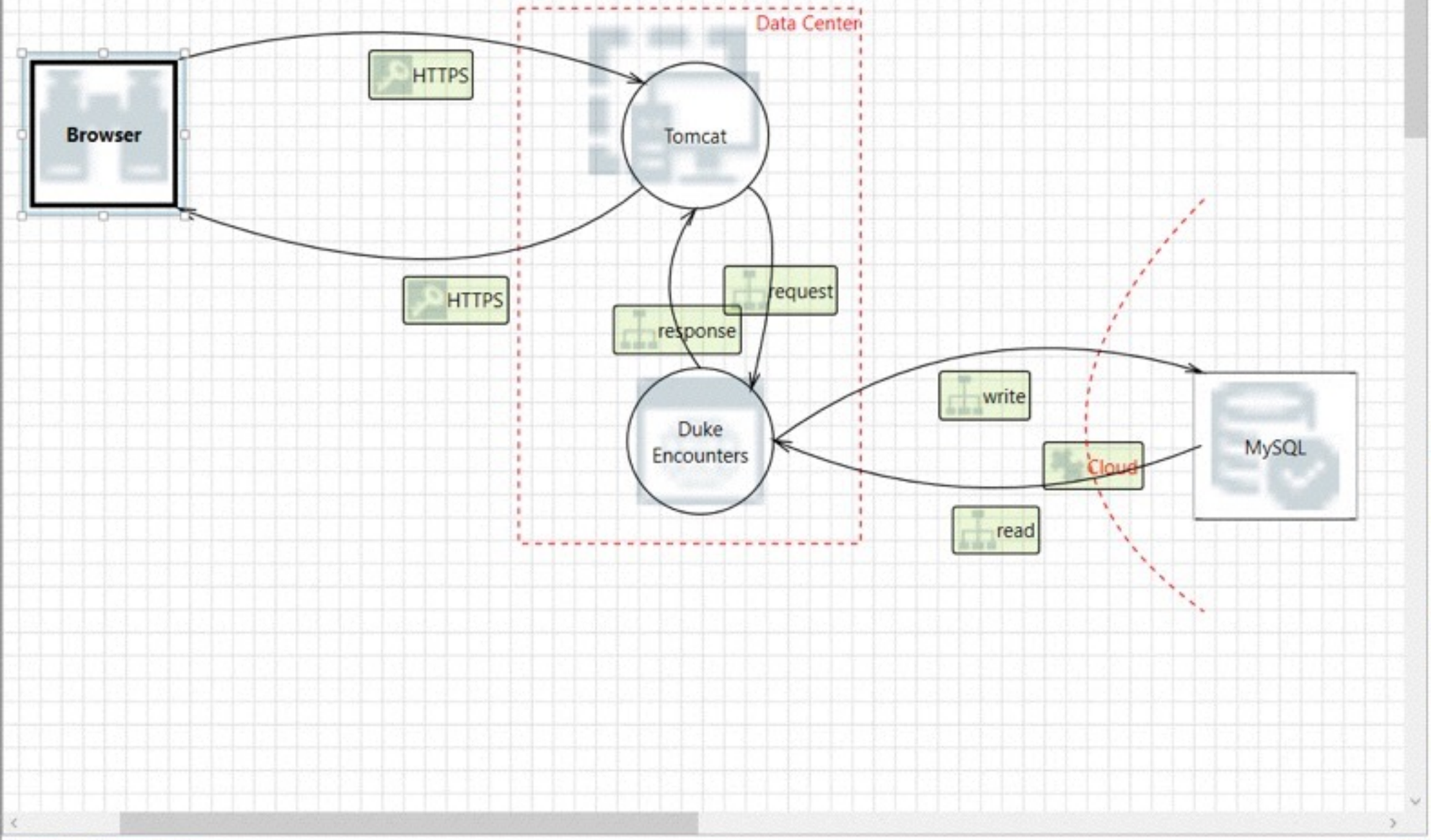
Threat Modeling in **Action**

Use one tool to threat model and version your models in a repo. Check and update them every time the application changes.

Duke Encounters - Threat Modeling Tool 2016

File Edit View Settings Diagram Reports Help

Duke Encounters



Stencils

- IOCTL Interface
- Generic Trust Line Boundary
  - Internet Boundary
  - Machine Trust Boundary
  - User mode or Kernel mode Boundary
  - AppContainer Boundary
- Generic Trust Border Boundary
  - CorpNet Trust Boundary
  - Sandbox Trust Boundary Border
  - Internet Explorer Boundaries
  - Other Browsers Boundaries
- Free Text Annotation

Element Properties

Browser

Name	Browser
Out Of Scope	☐
Reason For Out Of Scope	

Predefined Static Attributes

Type	Code

Configurable Attributes

As Generic External Interactor

Authenticates Itself	Not Applicable
Microsoft	No

Add New Custom Attribute

Messages - No issues found

Description	Severity	Diagram	Ignore

Demo

SeaSponge

mozilla.github.io/seasponge/#/draw

SeaSponge | File | Zoom-In | Zoom-Out | Web Application Sample | Refresh Sharable Link | Sha

Model Information

Diagram Information

New Threat

Threat Information

High SQL Injection

Stencils

Stencil Properties - Generic Trust Boundary

Boundary

Delete Element

Title

Database

Category

Boundary

Tags

Comma separated tags

Icon

images/icons/database11.svg

Code Type

Managed

Running As

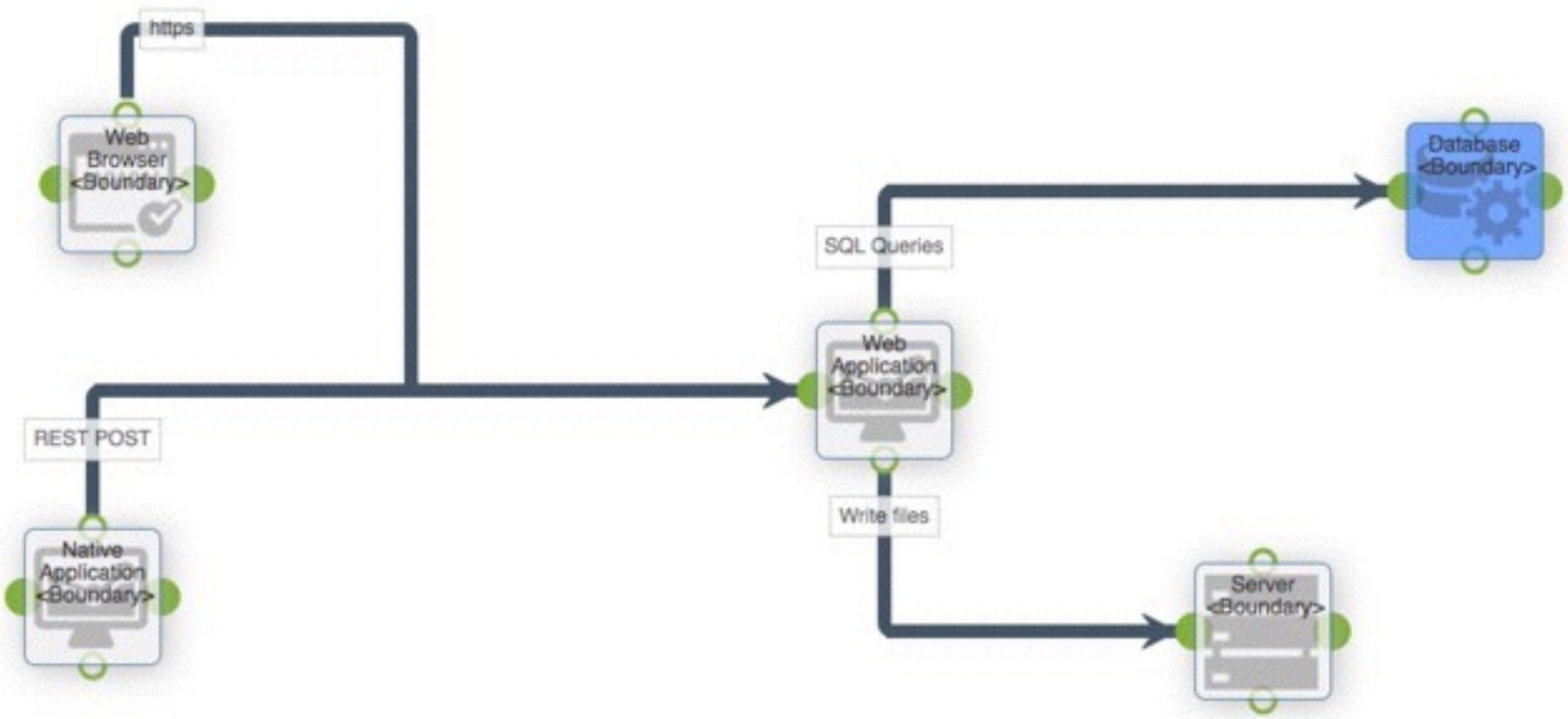
Kernel

Accepts Input From

Kernel, System, or Local Admin

Has Authentication Scheme

Web Application Sample | + Diagram



Update the model

A threat model is a living document

New threats might arise without a single change in the system. Update the model with every application change.

Test your threats

No threat (mitigation) without test

Write positive (normal usage) and negative (attack) tests for each threat.

```
@Test
@WithMockUser(username = "admin", password =
"admin", roles = "ADMIN")
public void verifyAdminAuthorizeOK() {
  mvc.perform(get("/admin"))
    .andExpect(status().isOk());
}
```

```
@Test
@WithMockUser(username = "user", password =
"user", roles = "USER")
public void verifyAdminAuthorizeNOK() {
  mvc.perform(get("/admin"))
    .andExpect(status().isForbidden());
}
```

Summary

Threat model before you start to code

Address every threat and test your solution

Remember there is no total security

bridging IT

BridgingIT GmbH
Königstraße 42
70173 Stuttgart

dominik.schadow@bridging-it.de
www.bridging-it.de/entwickler
blog.dominikschadow.de @dschadow

Avoiding the Top 10 Software Security Design Flaws
www.computer.org/cms/CYBSI/docs/Top-10-Flaws.pdf

Microsoft Threat Modeling Tool
www.microsoft.com/en-us/sdl/adopt/threatmodeling.aspx

Mozilla SeaSponge
air.mozilla.org/mozilla-winter-of-security-seasponge-a-tool-for-easy-threat-modeling

Threat Modeling: Designing for Security (Adam Shostack)
eu.wiley.com/WileyCDA/WileyTitle/productCd-1118809998.html

Pictures
www.dreamstime.com