VALUE DRIVEN THREAT MODELING

Security By Design

By Avi Douglen, CEO Bounce Security

Bounce SECURITY

Bounce
SECURITY

Summary

- Threat Modeling: most effective security activity
- Developers should Threat Model too!
  - *Not just the security team*

- Can be done quick and agile
- Prioritize by business value

Bounce SECURITY

About Me

- Email: AviD@BounceSecurity.com
- Twitter: @sec_tigger
- He / Him

- The important things:
  - *Whisky: smokey*
  - *Beer: stout*
  - *Coffee: strong*

- Software Security @ Bounce SECURITY
- Researcher / Developer / Architect
- Moderator Security.StackExchange
- Volunteer High School teacher
- OWASP Israel Leader
- Threat Model Project Leader

- Worldwide organization
- Dedicated to software security
- Free and Open source
- All volunteers
- Lots of projects
  - *Libraries*
  - *Tools*
  - *Guides*

- Israel chapter from 2006
- AppSecIL conference
  - *September 6th*
  - *Over 700 attending*
- Secure Coding Training
  - *September 5th*
  - *400 devs expected*
- FREE

Bounce
SECURITY

What is Threat Modeling?

- Structured security-based analysis
- Framework to understand threats
- Review of Design Elements
- Prioritize Mitigations by Risk

Bounce SECURITY

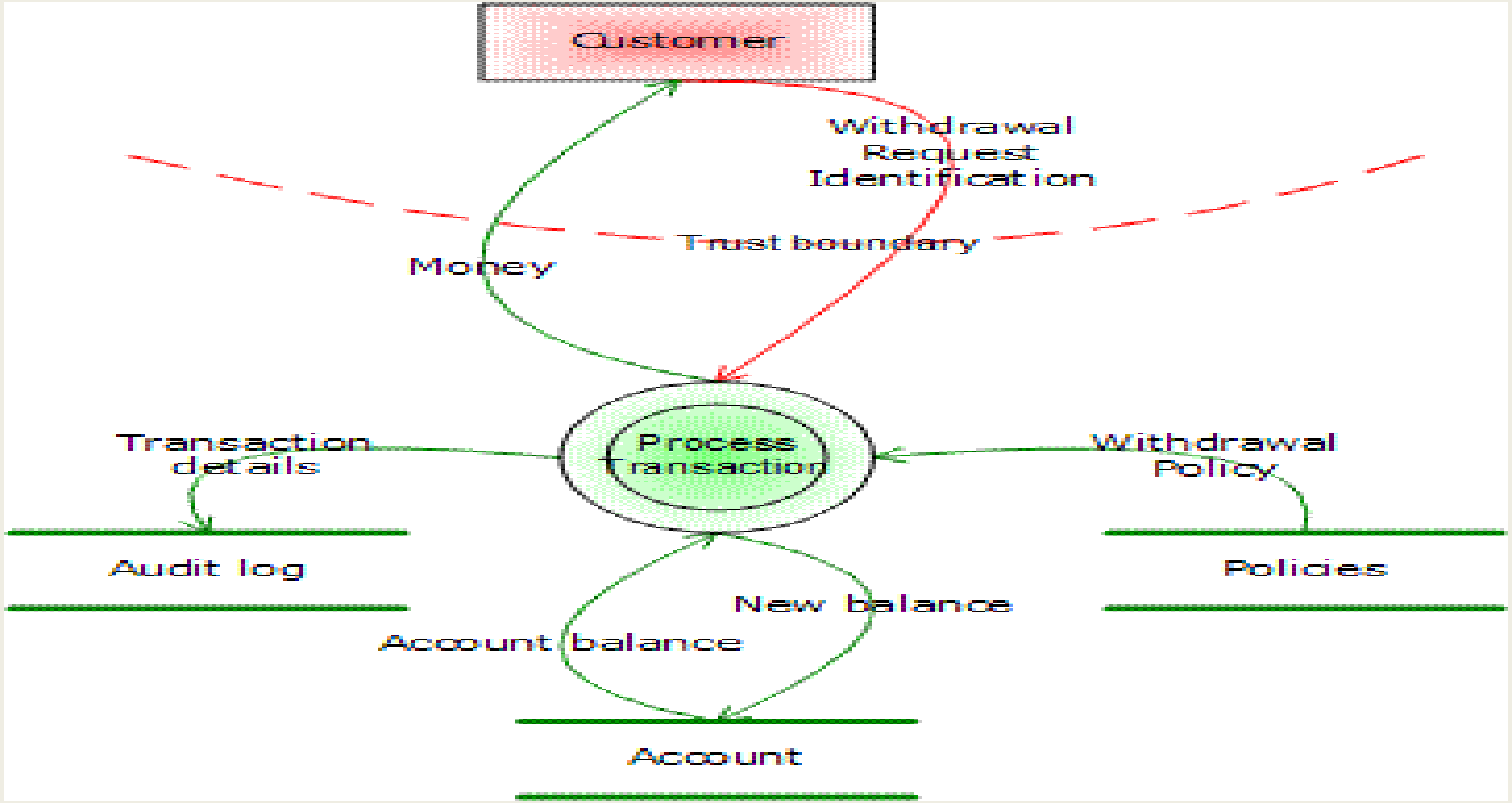
Why Threat Modeling?

- Discover unexpected potential attacks
- Understand risk and prioritize vulnerabilities
- Focus security efforts more efficiently
- Communication
- Documenting mitigations

“Classic” Threat Modeling

- Data Flows and Attack Surface
- Focus on Assets, Trust Boundaries
- Visually with DFDs or other diagrams

Data Flow Diagram



Bounce SECURITY

Process Outline

- Step#0: Scoping the Model
- Step#1: Decompose the Application
- Step#2: Identify the Threats (and risk level)
- Step#3: Determine the Countermeasures
- Step#4: Analyze Result

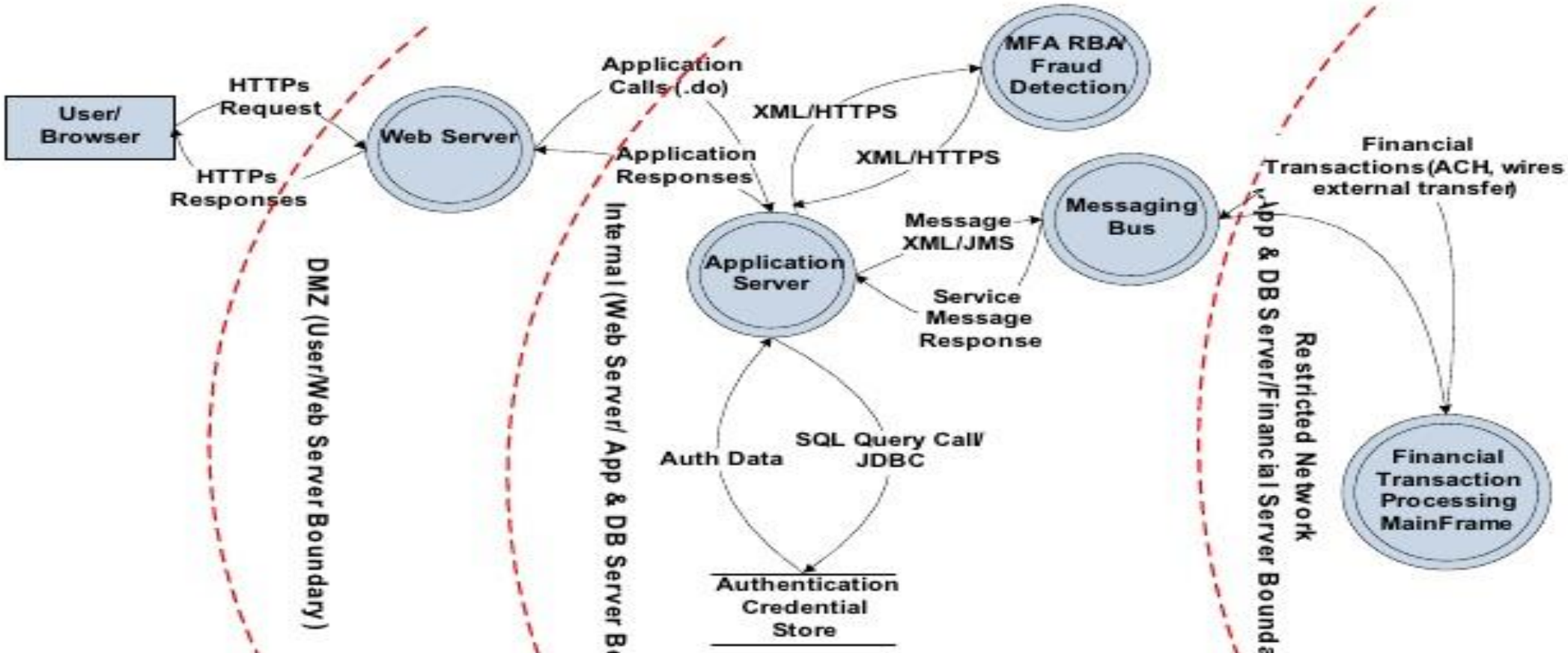
Classic Methodologies

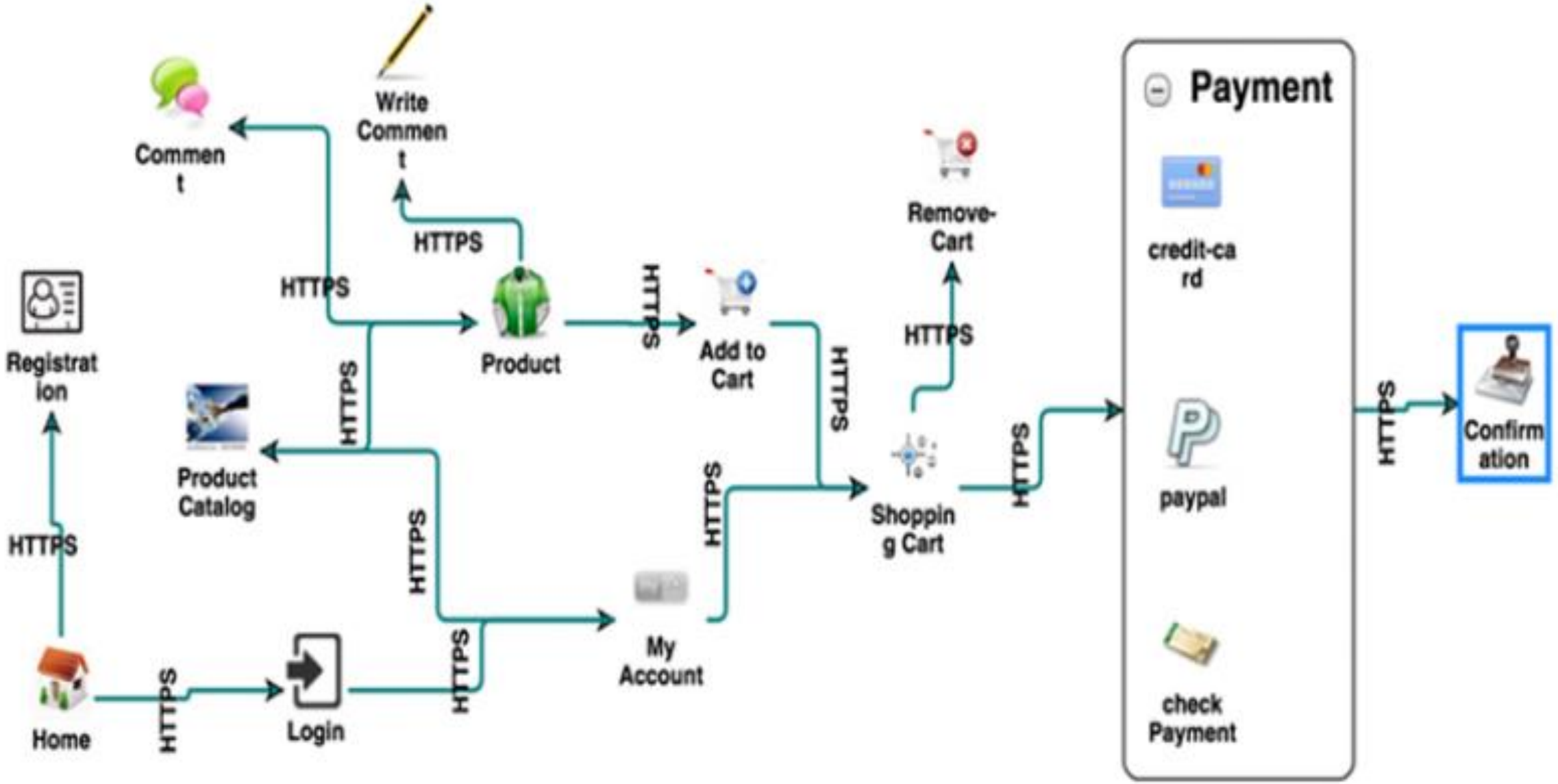
- STRIDE / STRIDE-per-element
- Attack Trees
- Asset-Focused
- Software-centric
- Attacker-focused
- Risk-Based

STRIDE Per-Element

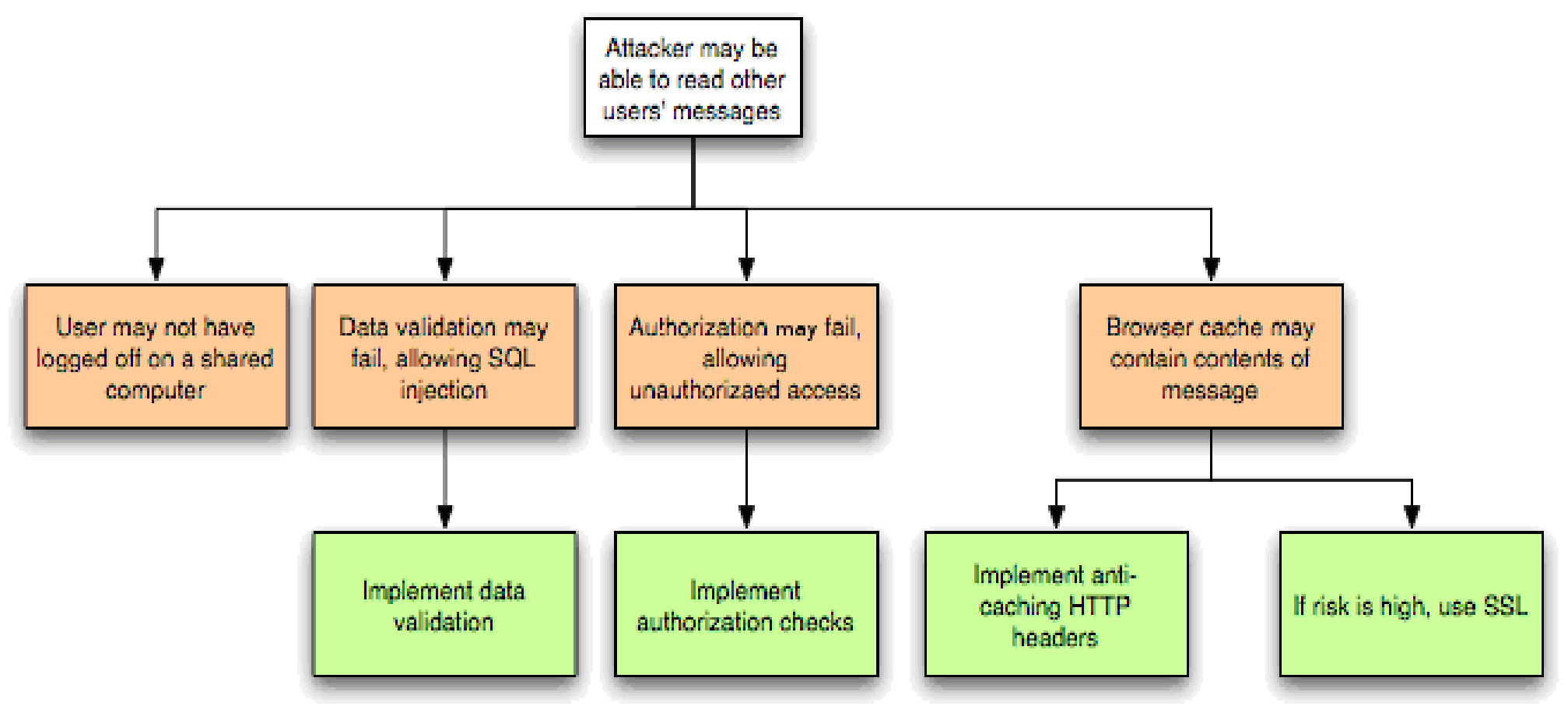
- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privileges

Data flow diagram-Online Banking Application




Comment
Write Comment
HTTPS
HTTPS
Product
HTTPS
Add to Cart
Remove-Cart
HTTPS
HTTPS
Payment
credit-card
Registration
Product Catalog
HTTPS
HTTPS
HTTPS
HTTPS
Shopping Cart
HTTPS
paypal
HTTPS
Confirmation
HTTPS
HTTPS
My Account
HTTPS
HTTPS
Home
HTTPS
Login
check Payment

Attack Trees



Bounce SECURITY

P.A.S.T.A

- Risk-Based Methodology for higher assurance
- **P**rocess for **A**ttack **S**imulation and **T**hreat **A**nalysis

- Seven stage process:


1. Define Objectives
Identify Business Objectives
Identify Security & Compliance Requirements
Business Impact Analysis
2. Define Technical Scope
Capture the boundaries of the technical environment
Capture Infrastructure | Application | Software Dependencies
3. Application Decomposition
Identify Use Cases | Define App Entry Points & Trust levels
Identify Actors | Assets| Services | Roles| Data Sources
Data Flow Diagramming (DFDs) | Trust Boundaries
4. Threat Analysis
Probabilistic Attack Scenarios Analysis
Regression Analysis on Security Events
Threat Intelligence Correlation & Analytics
5. Vulnerability & Weakness Analysis
Queries of Existing Vulnerability Reports & Issues Tracking
Threat to Existing Vulnerability Mapping Using Threat Trees
Design Flaw Analysis Using Use & Abuse Cases
Scorings (CVSS/ CWSS) | Enumerations (CWE/CVE)
6. Attack Modeling
Attack Surface Analysis
Attack Tree Development | Attack Library Mgt
Attack to Vulnerability & Exploit Analysis using Attack Trees
7. Risk & Impact Analysis
Qualify & quantify business impact
Countermeasure Identification & Residual
ID risk mitigation

Bounce
SECURITY

Common Objections

- “Security is everybody’s job”

Bounce SECURITY

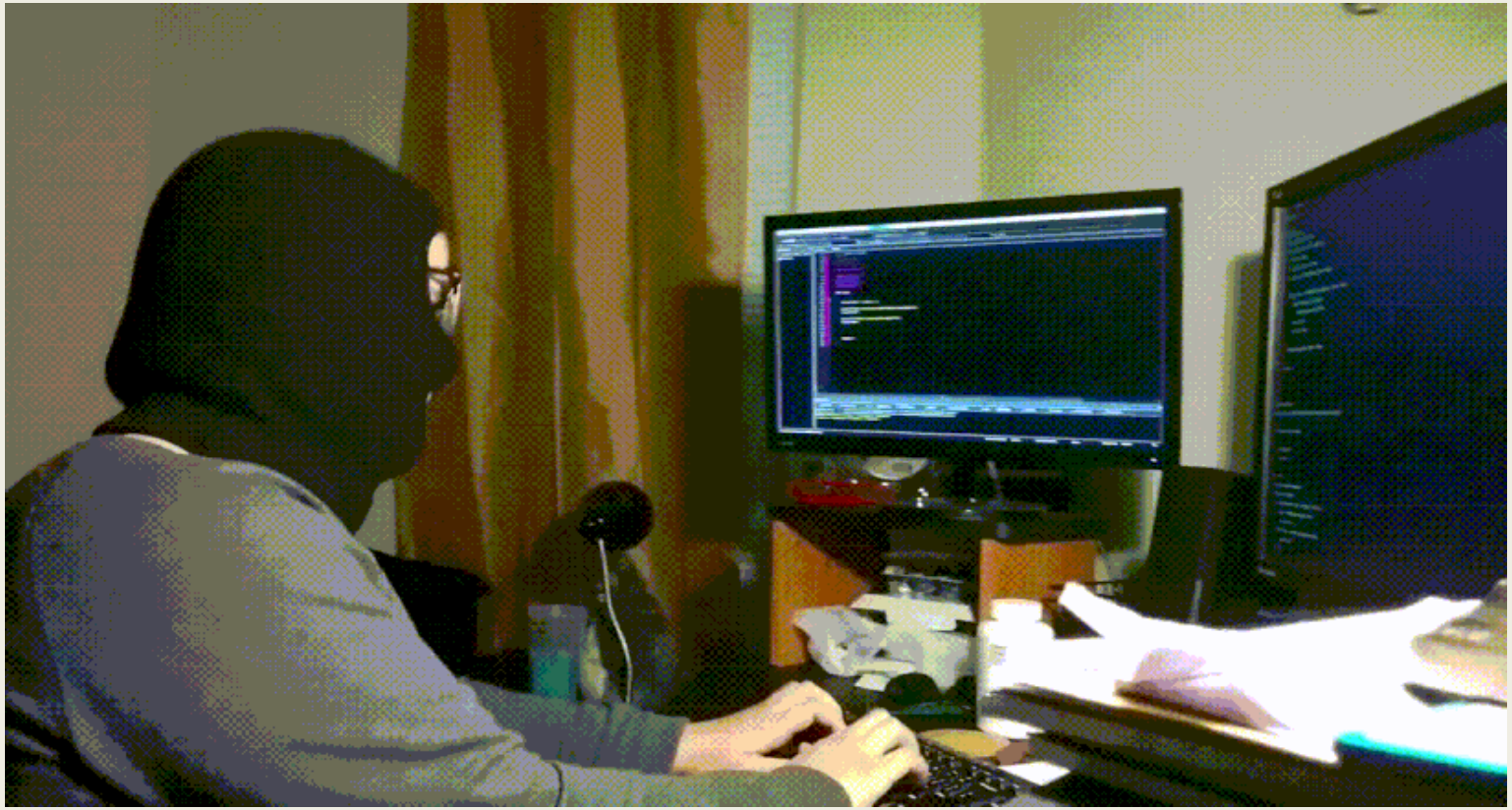
Common Objections

- Takes too much time!

Bounce SECURITY

Common Objections

- “Think like an attacker”



Common Objections

- Big Model Up Front

Bounce SECURITY

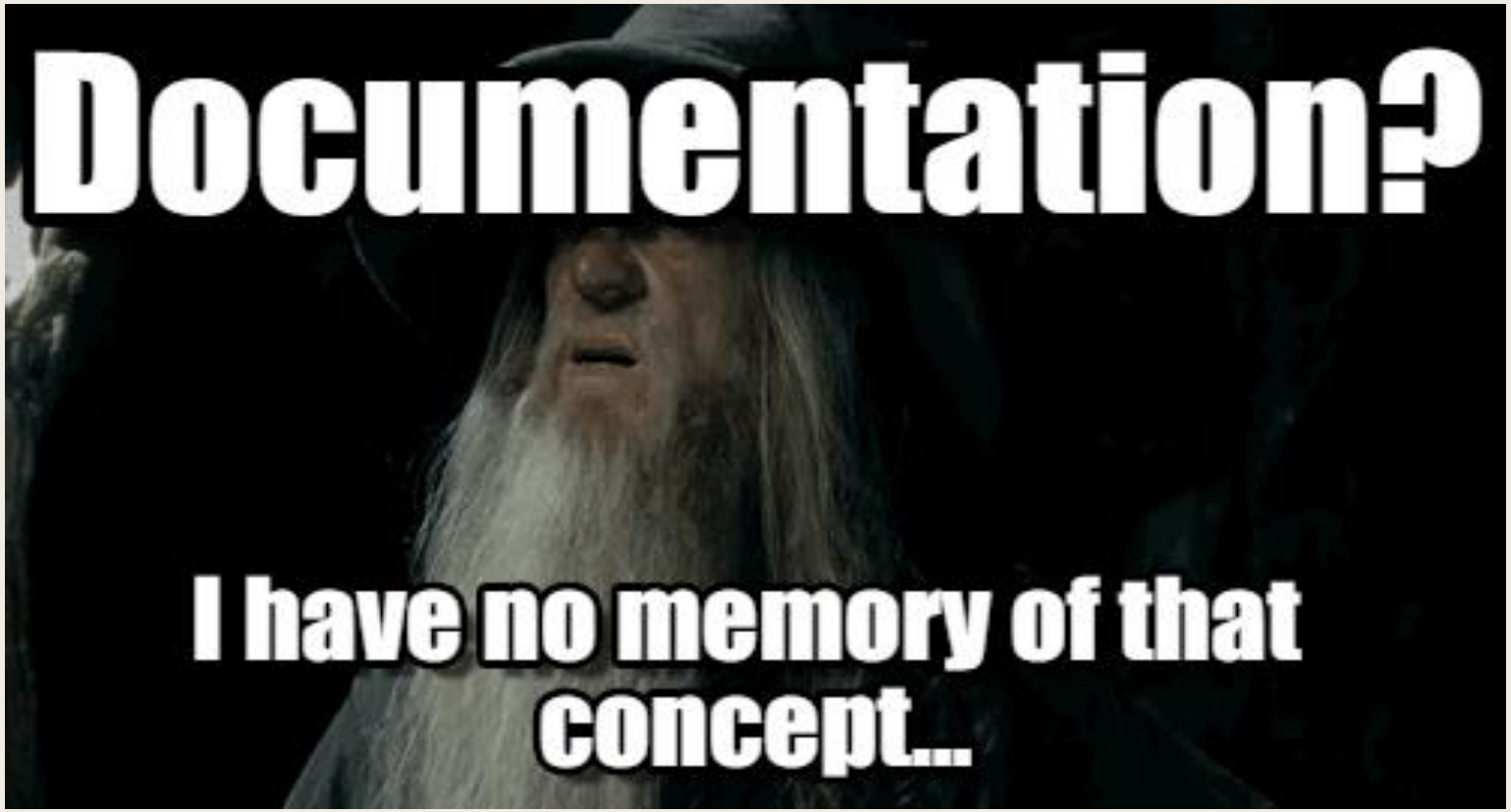
Common Objections

- Use case vs. user story approach

Bounce SECURITY

Common Objections

- Threat model separate from design



Bounce SECURITY

Common Objections

- Usually out of date, often before its complete

Bounce SECURITY

Common Objections

- Wasted time on unrealistic threats

Bounce SECURITY

Common Objections

- Mismatch with Security Team

Bounce SECURITY

Common Objections

- Security team doesn’t scale

Bounce SECURITY

Common Objections

- Security team drops in and out

Bounce SECURITY

Common Objections

Bounce SECURITY

Bounce
SECURITY

Bounce SECURITY

Bounce
SECURITY

Core of Classic TM

- 4 core questions of threat modeling:
  1. *What are you building?*
  2. *What can go wrong?*
  3. *What are you going to do about it?*
  4. *Did we do a good job?*

- "All Threat Models are wrong, some are useful"

Value Driven Threat Modeling

- Accept that it's wrong, focus on the usefulness

1. Why are we building this?
2. What needs to go right?
3. How do we make sure that happens?

Bounce
SECURITY

Value Driven Process

- Start from standard baseline
  - *Skip obvious threats (e.g. XSS, HTTPS)*
  - *Relies on basic code hygiene*
  - *Security training for all developers and testers!*
  - *Threat Library*

- Threat model each User Story / Epic
  - *During "Discovery" or Sprint Planning*
  - *Agile approach of "just enough" threat model*
  - *Threat model goes into the User Story*

Bounce SECURITY

Value Driven Process

- Find the value of each feature
  - *Follow the money!*
  - *How do people die?*

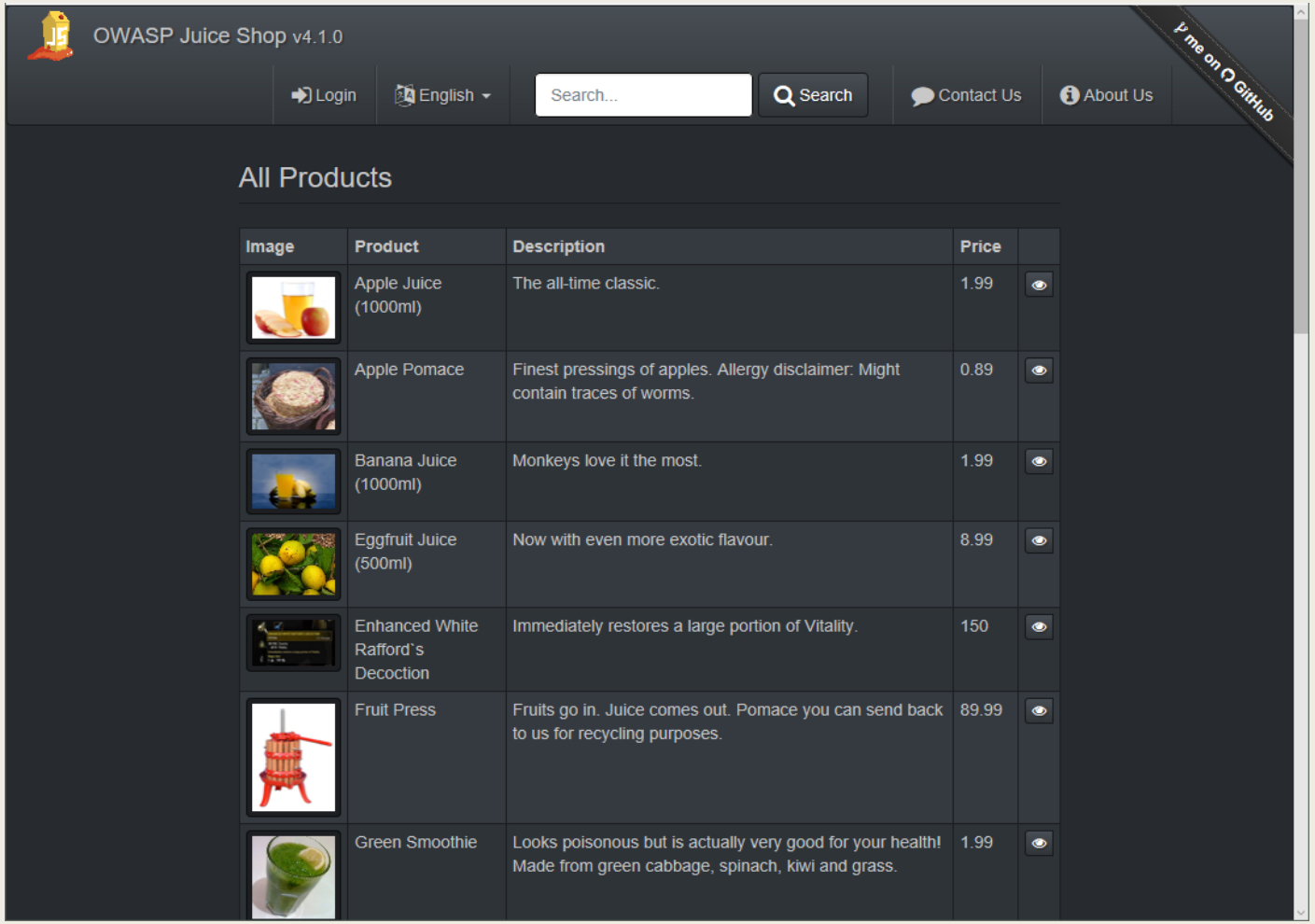
Bounce SECURITY

Value Driven Process

- State story goals
- Describe correct flow and conditions

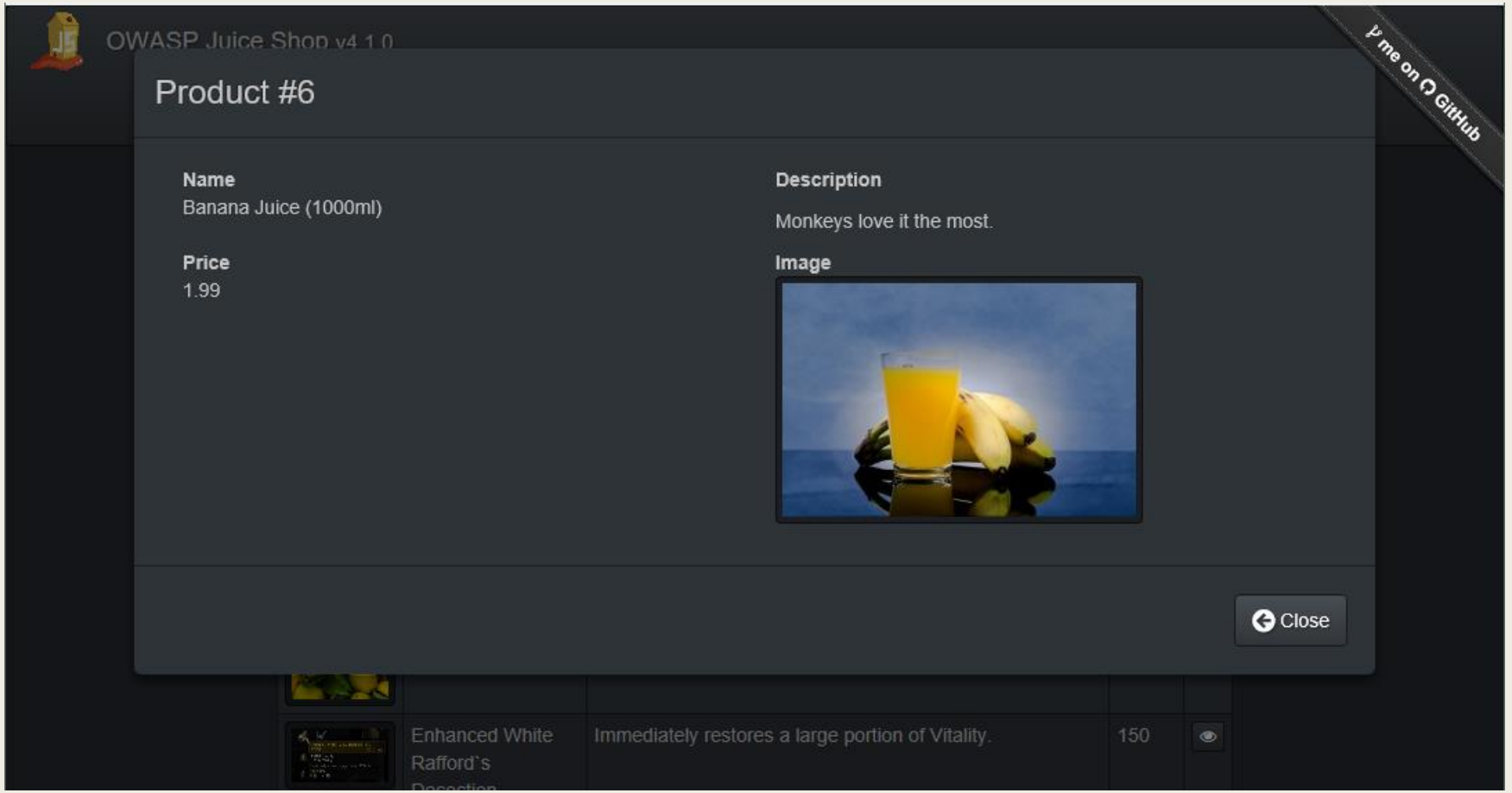
- Highlight assumptions and failure states
- Validate assumptions and enforce conditions
- Explicitly handle failure states

OWASP Juice Shop



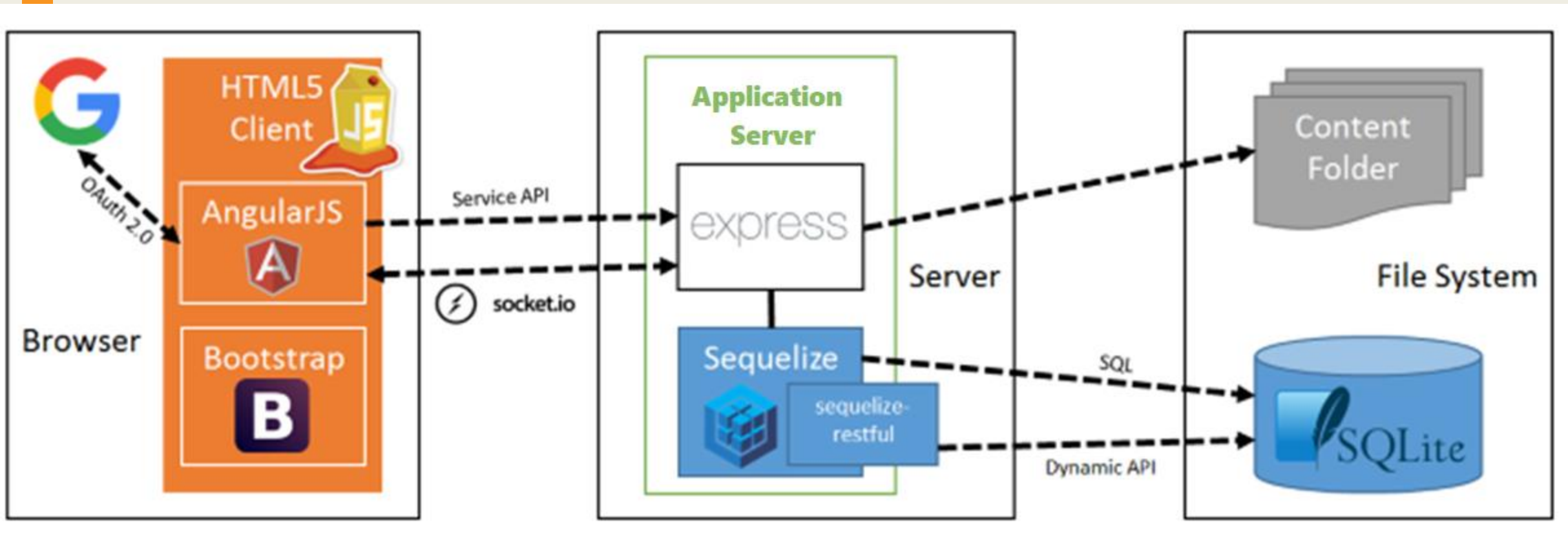
Bounce SECURITY

OWASP Juice Shop

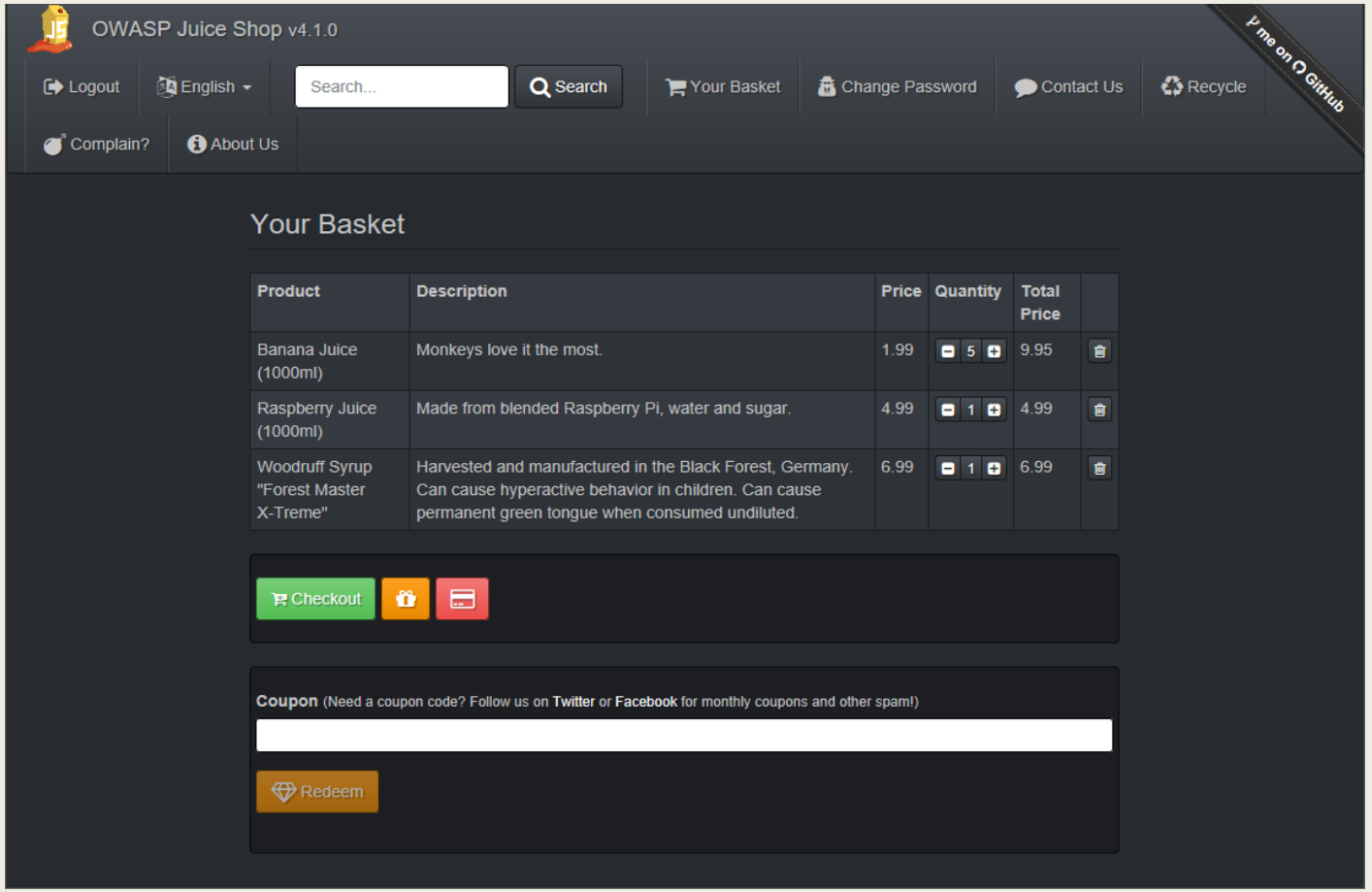


Bounce SECURITY

OWASP Juice Shop



OWASP Juice Shop



Bounce SECURITY

OWASP Juice Shop

Bounce SECURITY

Value Driven Techniques

- Definition of Done
- Acceptance Criteria

When I login with a wrong password,

I should be locked out after X times.

Value Driven Techniques

- Security unit tests

Test that user accounts are locked after X attempts

Test that locked user accounts are unlocked after Y time

Value Driven Techniques

- Abuser stories

As an attacker,

I want to try a large number of passwords,

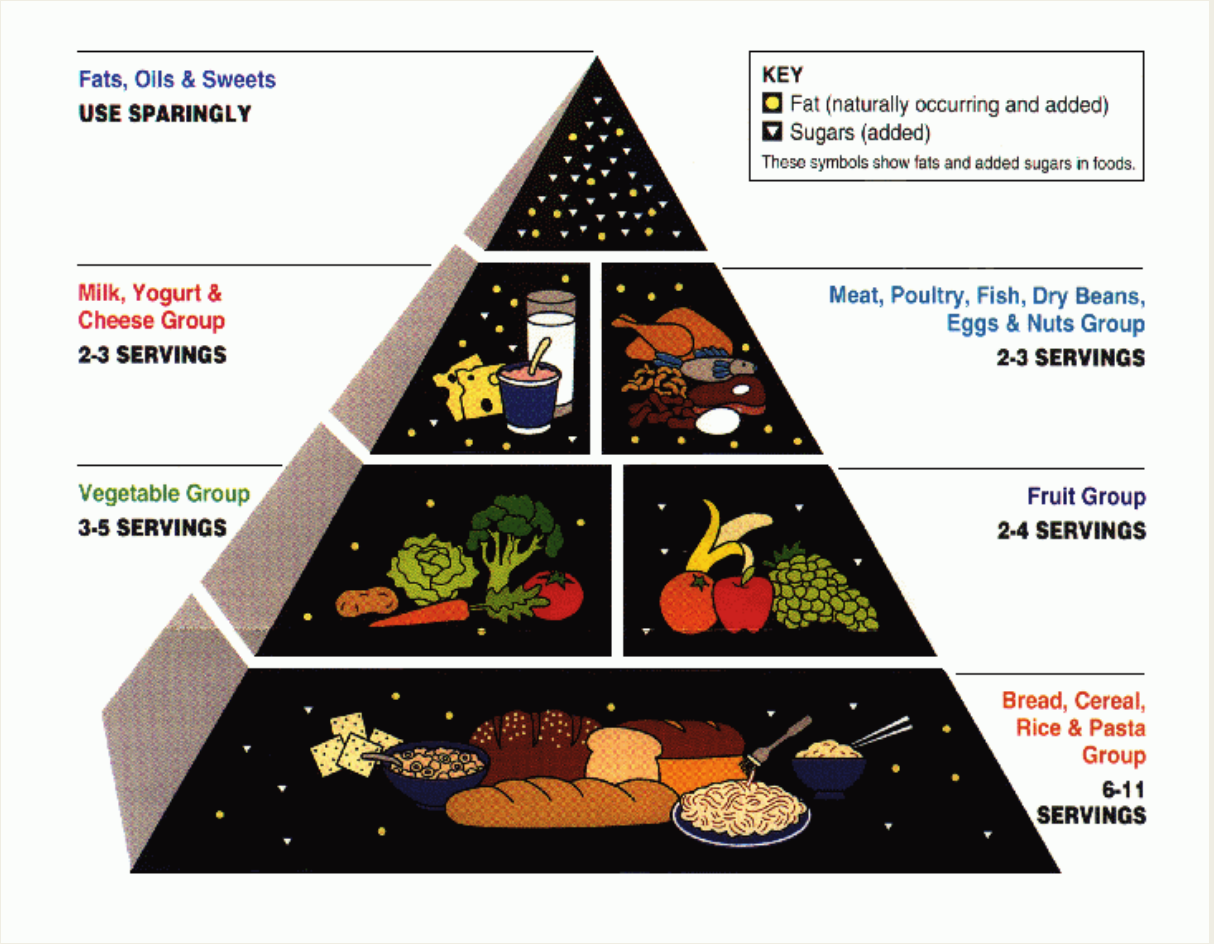
so that I can impersonate another user

and steal their juicebox

Bounce SECURITY

Value Driven Techniques

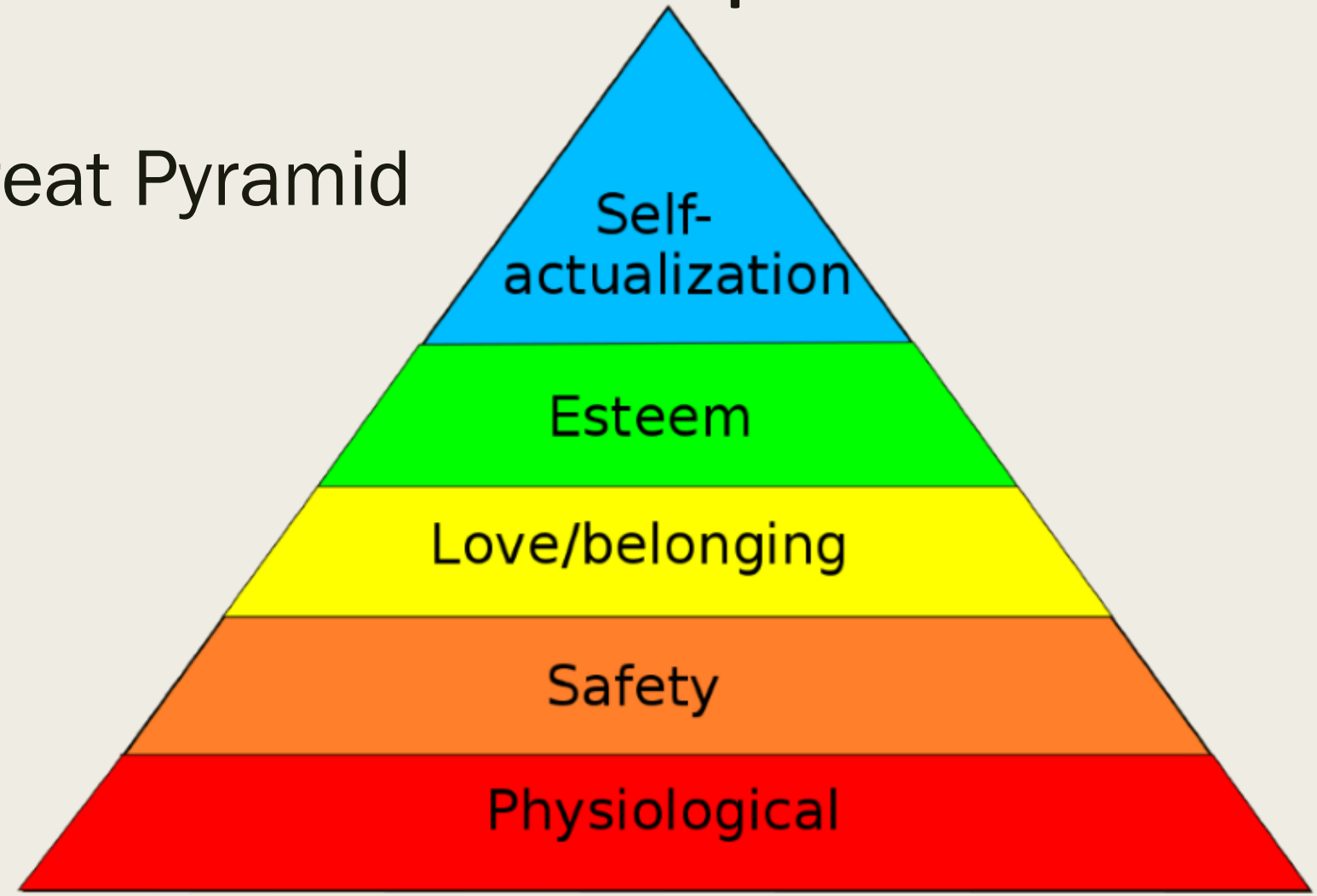
- Threat Pyramid



Bounce SECURITY

Value Driven Techniques

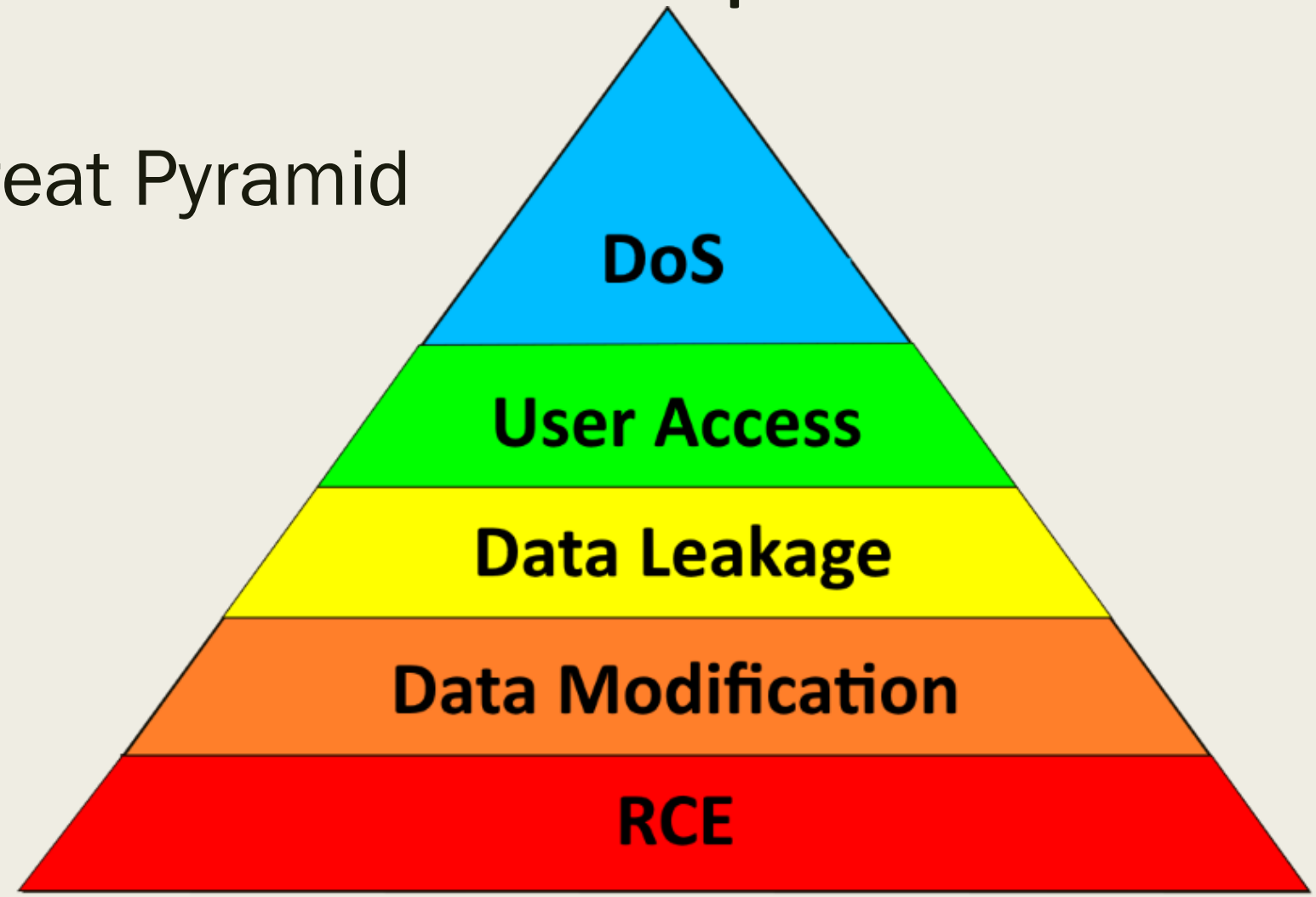
- Threat Pyramid



Bounce
SECURITY

Value Driven Techniques

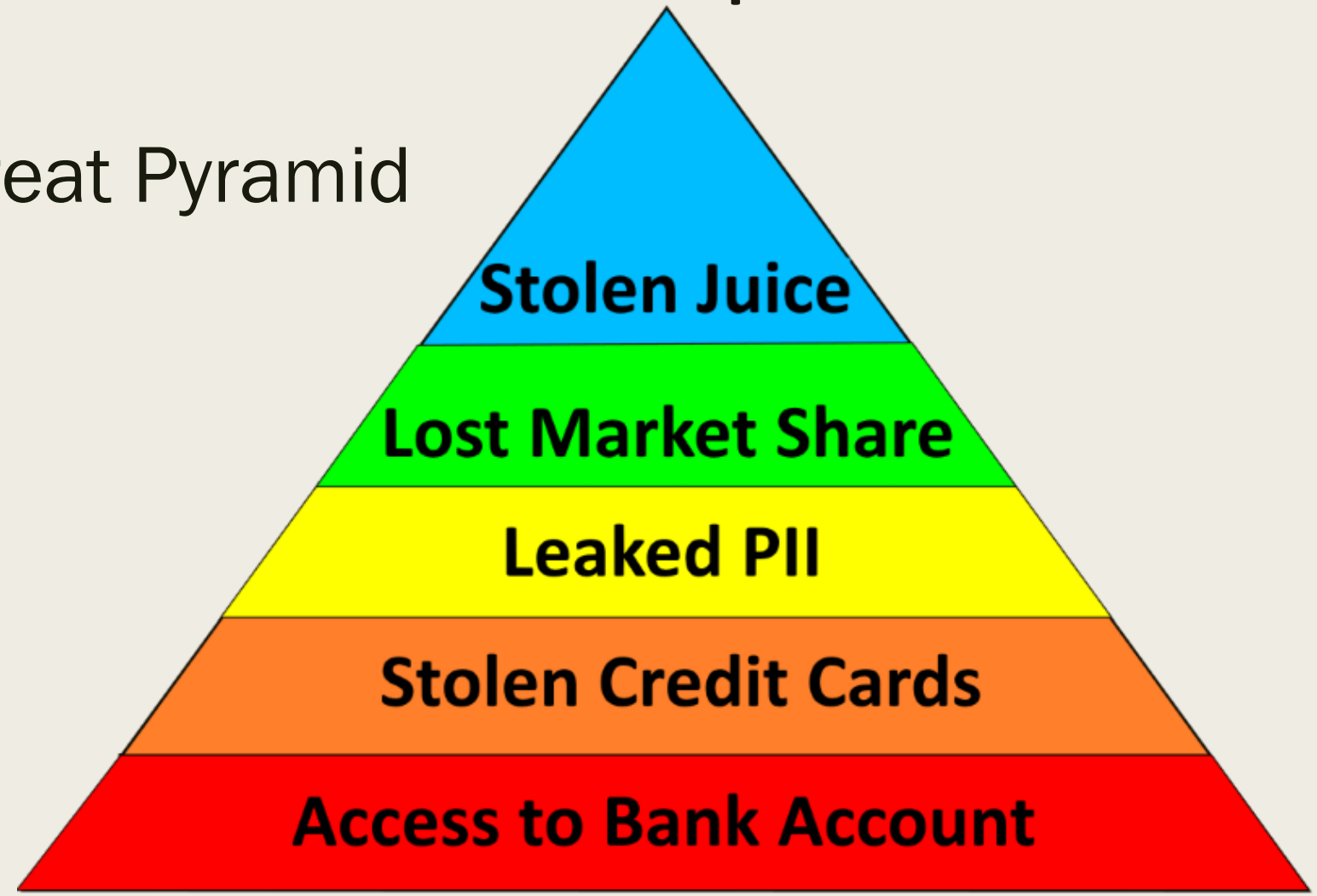
- Threat Pyramid



Bounce SECURITY

Value Driven Techniques

- Threat Pyramid



Bounce SECURITY

Value Driven Techniques

Story Points

Relative estimate of effort

↓

Sorry Points

Relative estimate of impact

“What if it goes horribly wrong?”

Bounce SECURITY

Benefits

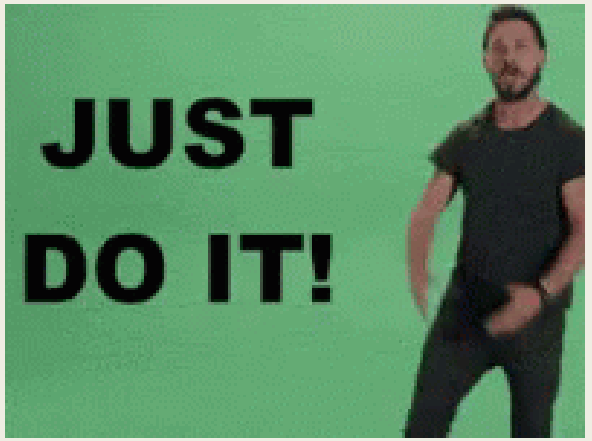
- Enables protection of application
- Efficient security investment
- Can integrate as part of agile design
- Justify NOT implementing security feature
- Don’t need piles of expensive consultants
- Take ownership of security requirements

Limitations

- Not complete
- Misses a LOT of threats
- Relies on developer experience
- Security champion must be part of team
- Low assurance for high risk systems

Summary

- Developers – start threat modeling!!
- TM should be part of dev workflow
- Focus on business value
- Start with the useful part of TM – and stop there
- Skip the overkill – until you really need it



Bounce SECURITY

Bounce SECURITY

THANKS FOR LISTENING!

Find me on Twitter: @sec_tigger

Bounce SECURITY