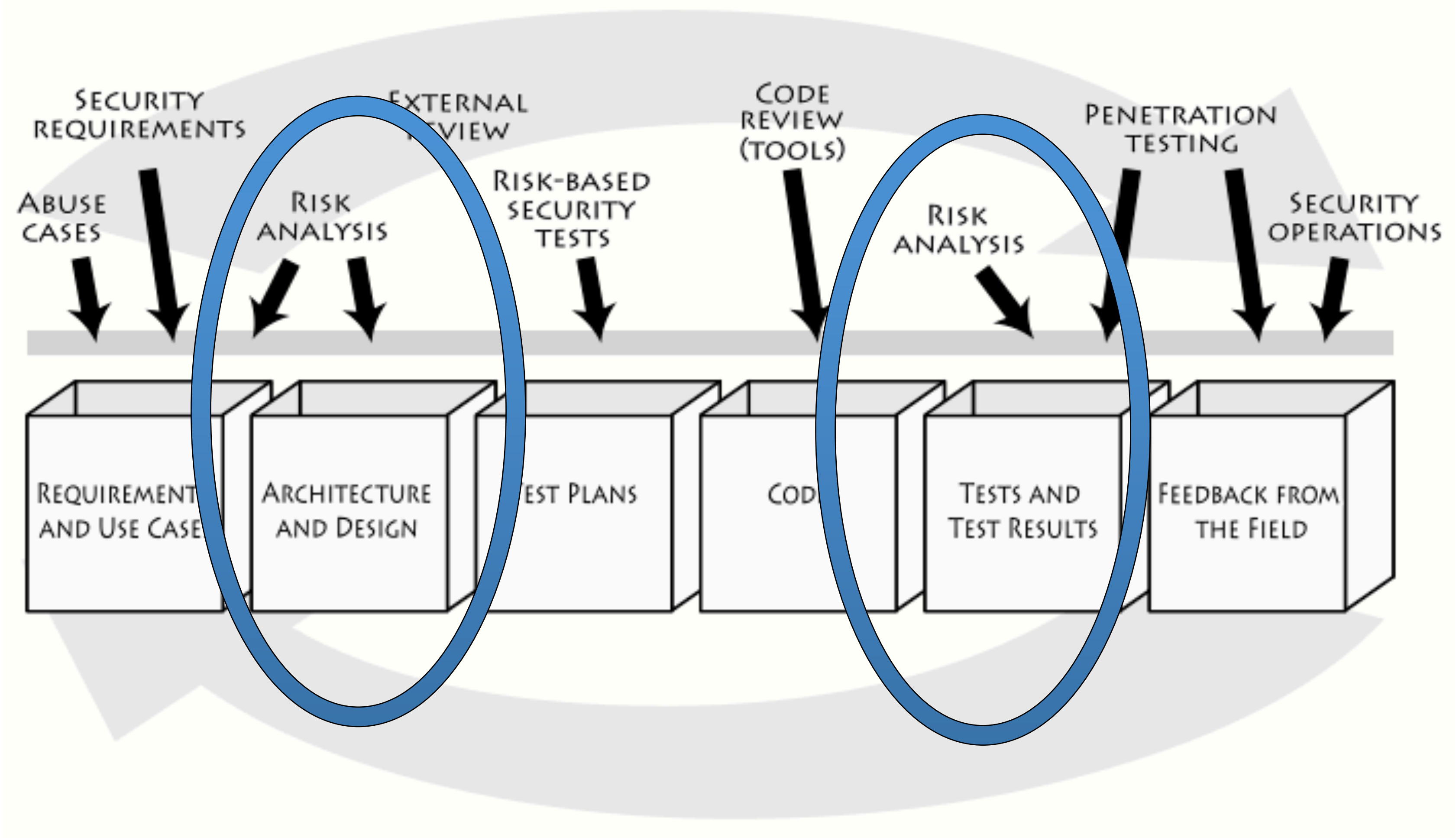
Architecture Analysis

JIM DELGROSSO
PRINCIPAL CONSULTANT

@JIMDELGROSSO

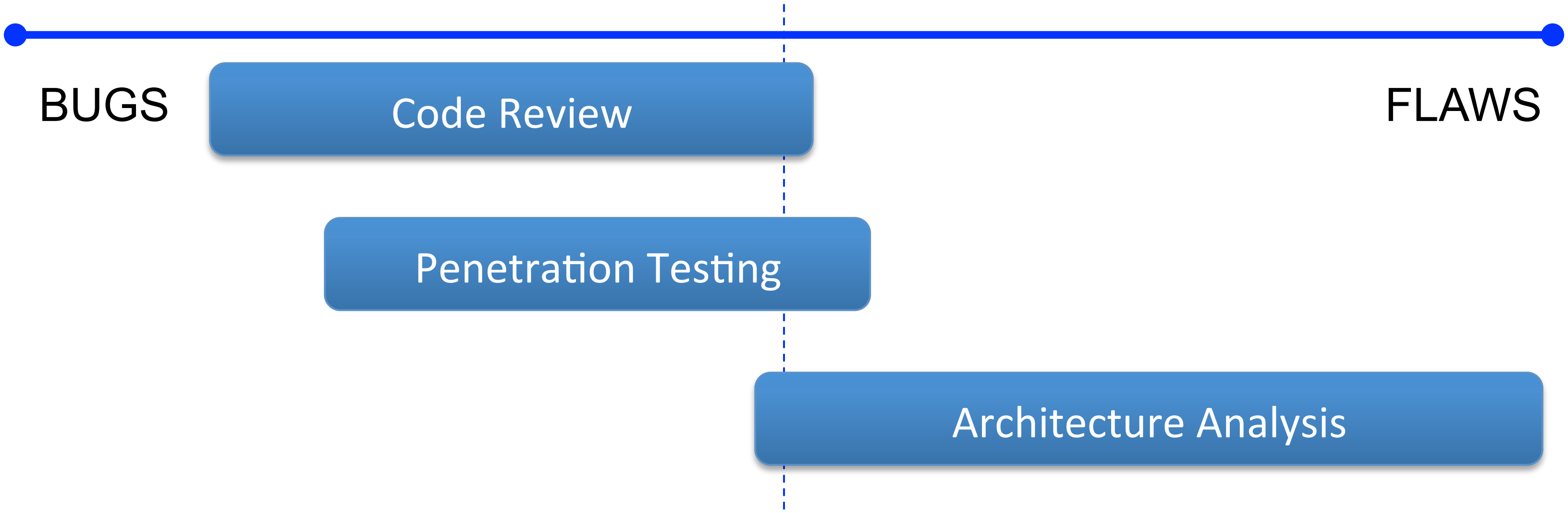
Software Security In The SDLC



The Software Defect Universe

Cross Site Scripting

Weak security control



BUGS

FLAWS

Code Review

Penetration Testing

Architecture Analysis

Bugs vs. Flaw Comparison

Cryptography

Bug

- Bug in open-source or COTS software

Flaw

- Use a confidentiality control where an integrity control is necessary
- Using crypto to hide poor design choices
- Poor key management design

Authentication

Bug

- LDAP Injection

Flaw

- Two-step authentication process with hidden user account, performed on client side

Logging Activities

Bug

- Log Injection
- Writing sensitive data to logs

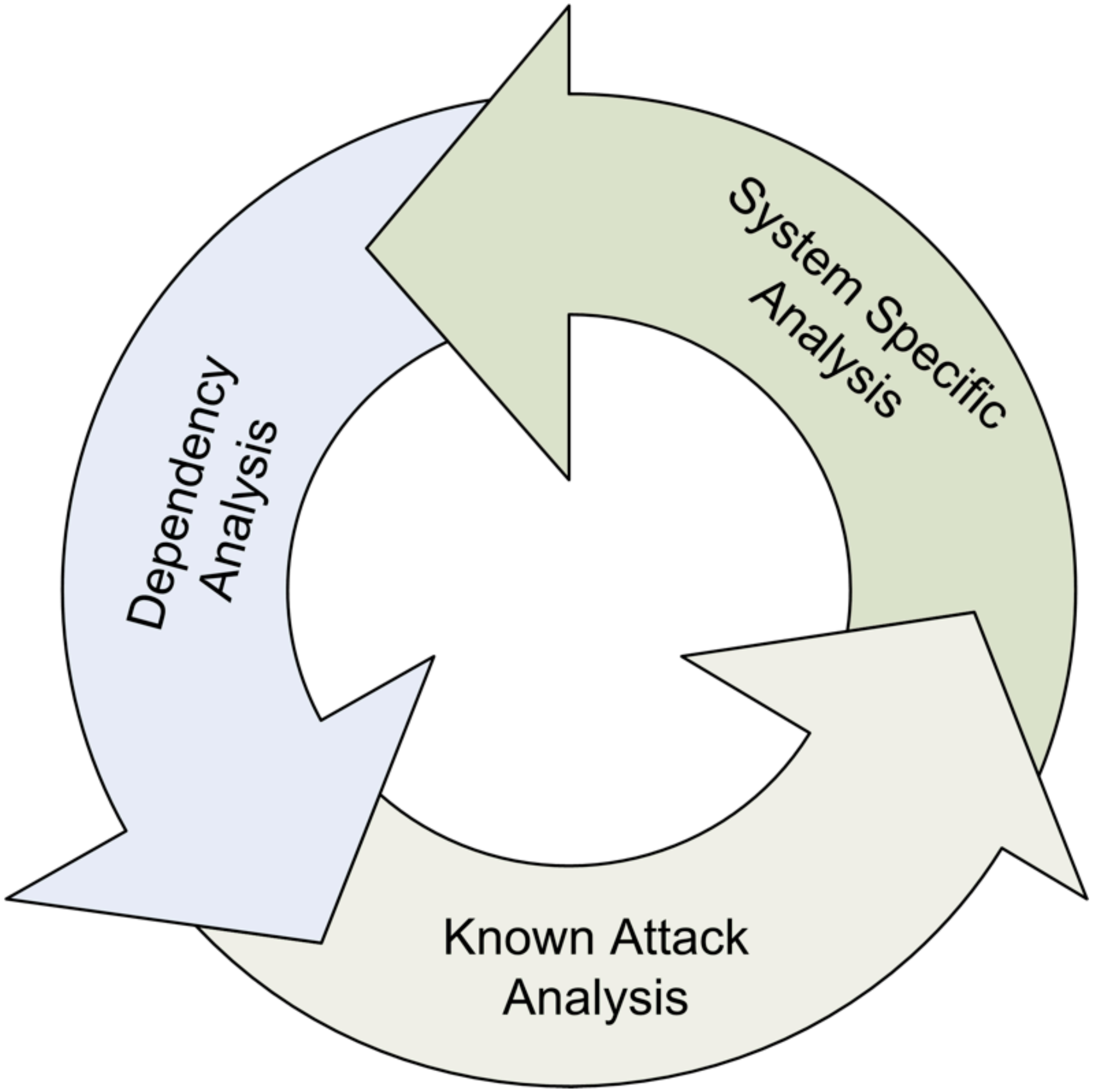
Flaw

- No standard tokenization of sensitive data to allow (easy) log aggregation
- Allow logs to be altered without detection

How To Find Flaws?

- Code review?
  - Unlikely with tool; maybe by manual review
- Pen-testing?
  - Unlikely without deep knowledge of system and possibly a lot of test time
- Need something else…
  - A type of analysis that is not code-based
  - A type of analysis focusing on how we design a system

How To Find Flaws?



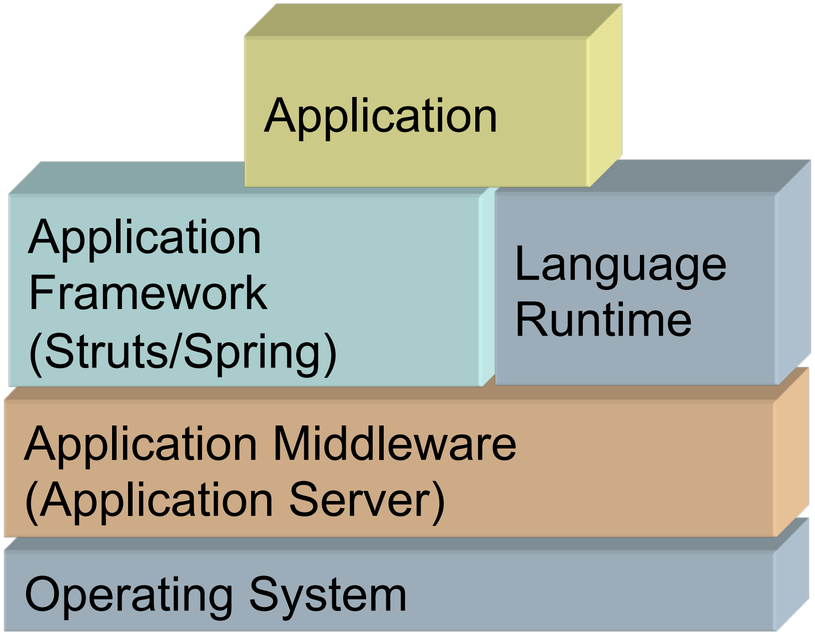
Finding Flaws

DEPENDENCY ANALYSIS

cigital

Dependency Analysis

Software is built upon layers of other software


Application
Application Framework (Struts/Spring)
Language Runtime
Application Middleware (Application Server)
Operating System

What kind of flaws are found?

- Known vulnerabilities in open-source or product versions
- Weak security controls provided with the framework
- Framework features that must be disabled or configured to their secure form

Dependency Analysis

The application environment provides controls. What are the limitations?

- Cryptography
  - Example: JCA
- Authentication and Authorization
  - Example: JAAS
- Input Validation and Output Encoding
  - .NET validateRequest
  - OWASP ESAPI
- Sandboxing
  - JavaScript Same Origin Policy

Finding Flaws

KNOWN ATTACK ANALYSIS

cigital

Known Attack Analysis

Understanding known attacks provide insight

- Designers – what controls are needed to prevent them
- Attackers – what to try again

OWASP
The Open Web Application Security Project
OWASP Top 10 - 2013
The Ten Most Critical Web Application Security Risks

Known Attack Analysis

What flaws show up “often”?

- Client-side trust
- Missing or weak control
  - ○ XSS
  - ○ CSRF
  - ○ Logging and auditing
- Session management
- Replay attacks

cigital

Known Attack Analysis

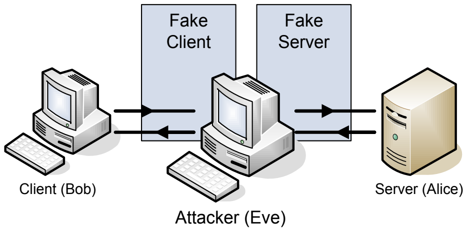
Identify design elements historically vulnerable to attack

- Distributed architecture
- Dynamic code generation and interpretation
- APIs across stateless protocols
- Rich Internet Applications
- Service-oriented Architecture

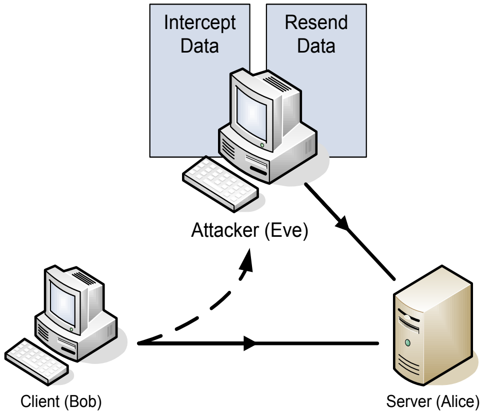
cigital

Distributed Architecture

- Distributed systems are susceptible to network-based attacks
  - Eavesdropping
  - Tampering
  - Spoofing
  - Hijacking
  - Observing



Interposition Attack



Replay Attack

Dynamic Code Generation and Interpretation

- Languages and programming environments are moving more decisions from design-time to run-time
- Many attacks involve misinterpretation of data as code in these environments
- When and how will user input be used by runtime language interpreters?

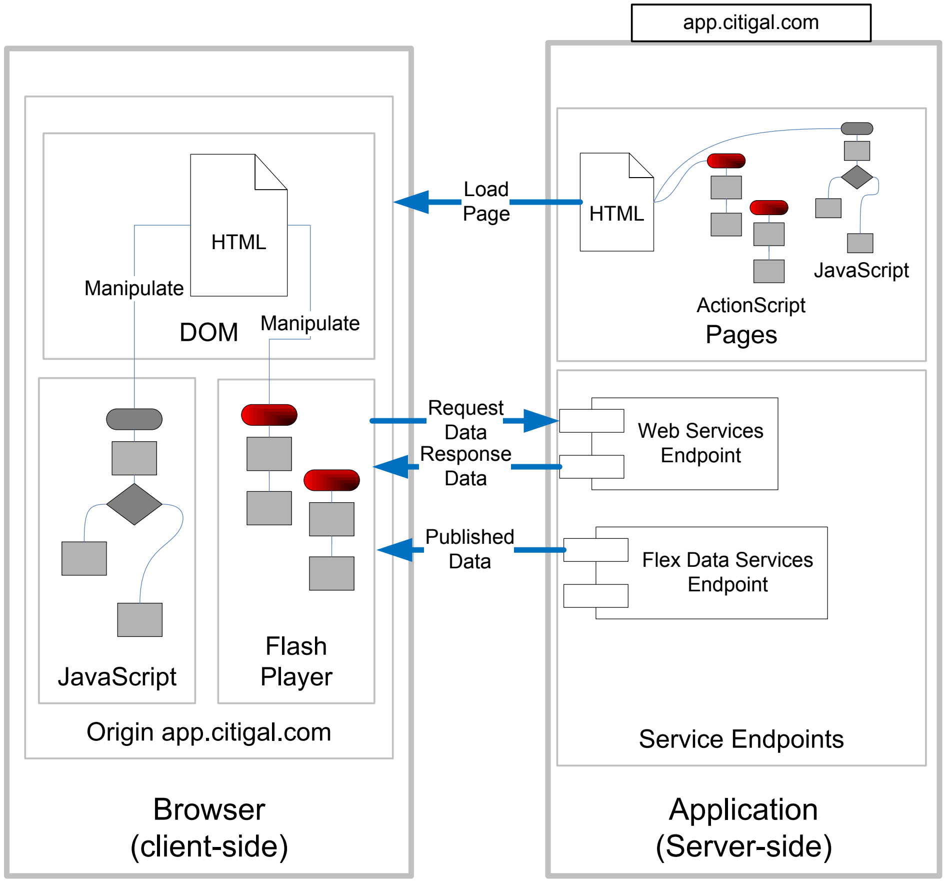
APIs Across Stateless Protocols

- Identifiers representing state can be abused
  - Prediction
  - Capture
  - Fixation
- State sent to the client between requests is altered or replayed

cigital

Rich Internet Applications

- Processing moves to the client



Service-Oriented Architecture (SOA)

- Security needed for SOA components
  - Web-services: SOAP/WSDL/UDDI
  - Message-oriented Middleware
  - Enterprise Service Bus

- Common Problems
  - Exposing backend code to dynamic attacks
  - Channel versus Message security

Finding Flaws

SYSTEM SPECIFIC ANALYSIS

System Specific Analysis Flaws

- Weakness in a custom protocol
- Reusing authentication credentials
- Not following good software security design principles

Thirteen Design Principles

- Secure the weakest link
- Defend in depth
- Fail securely
- Grant least privilege
- Separate privileges
- Keep things simple
- Be reluctant to share
- Be reluctant to trust
- Assume your secrets are not safe
- Mediate completely
- Make security usable
- Promote privacy
- Use your resources

http://searchsecurity.techtarget.com/opinion/Thirteen-principles-to-ensure-enterprise-system-security

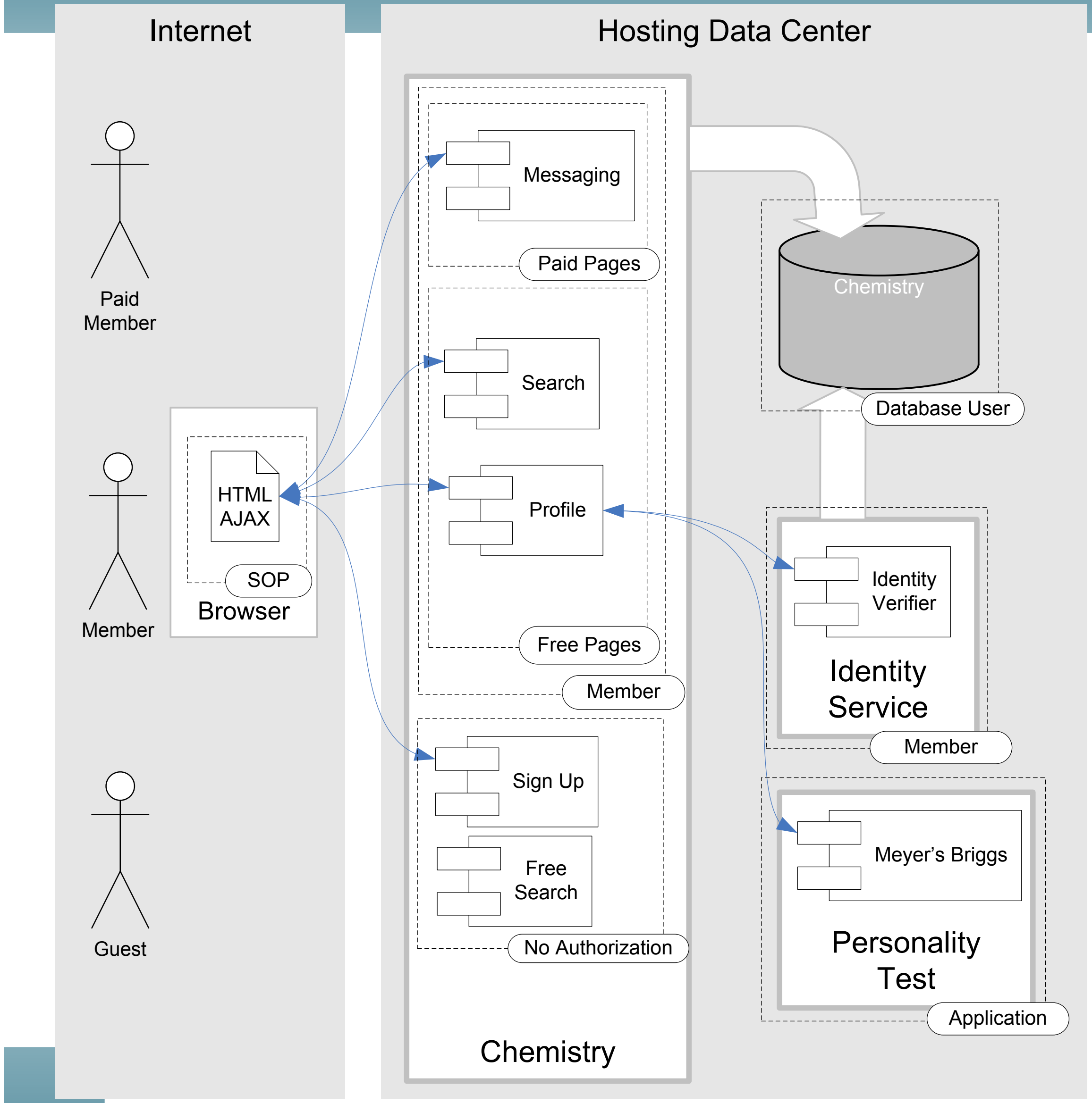
cigital

Threat Modeling

- Model the software by understanding
  - Threat agent
  - Asset
  - Attack
  - Attack surface
  - Attack goal
  - Security control

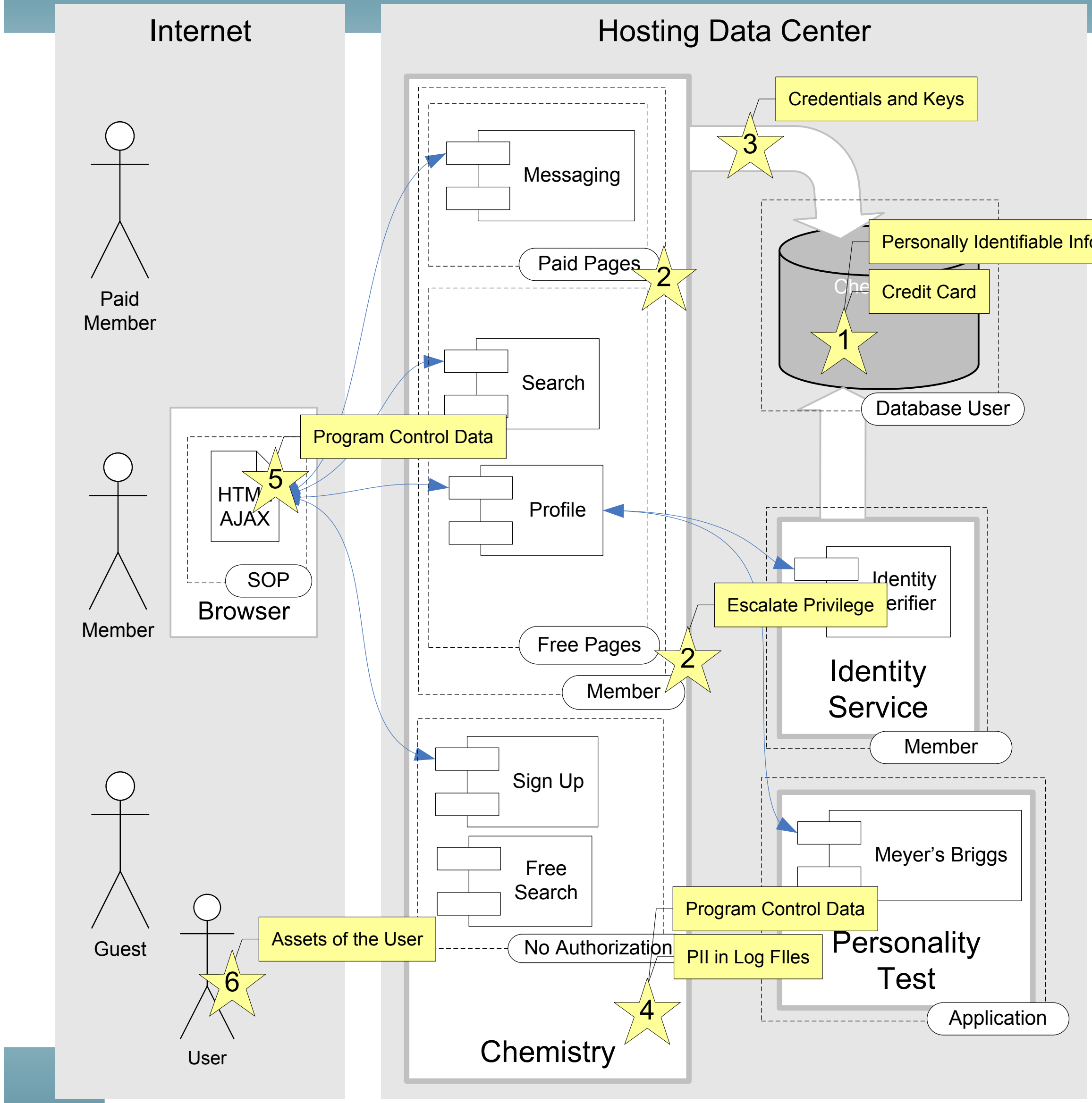
cigital

Who Is Attacking You?



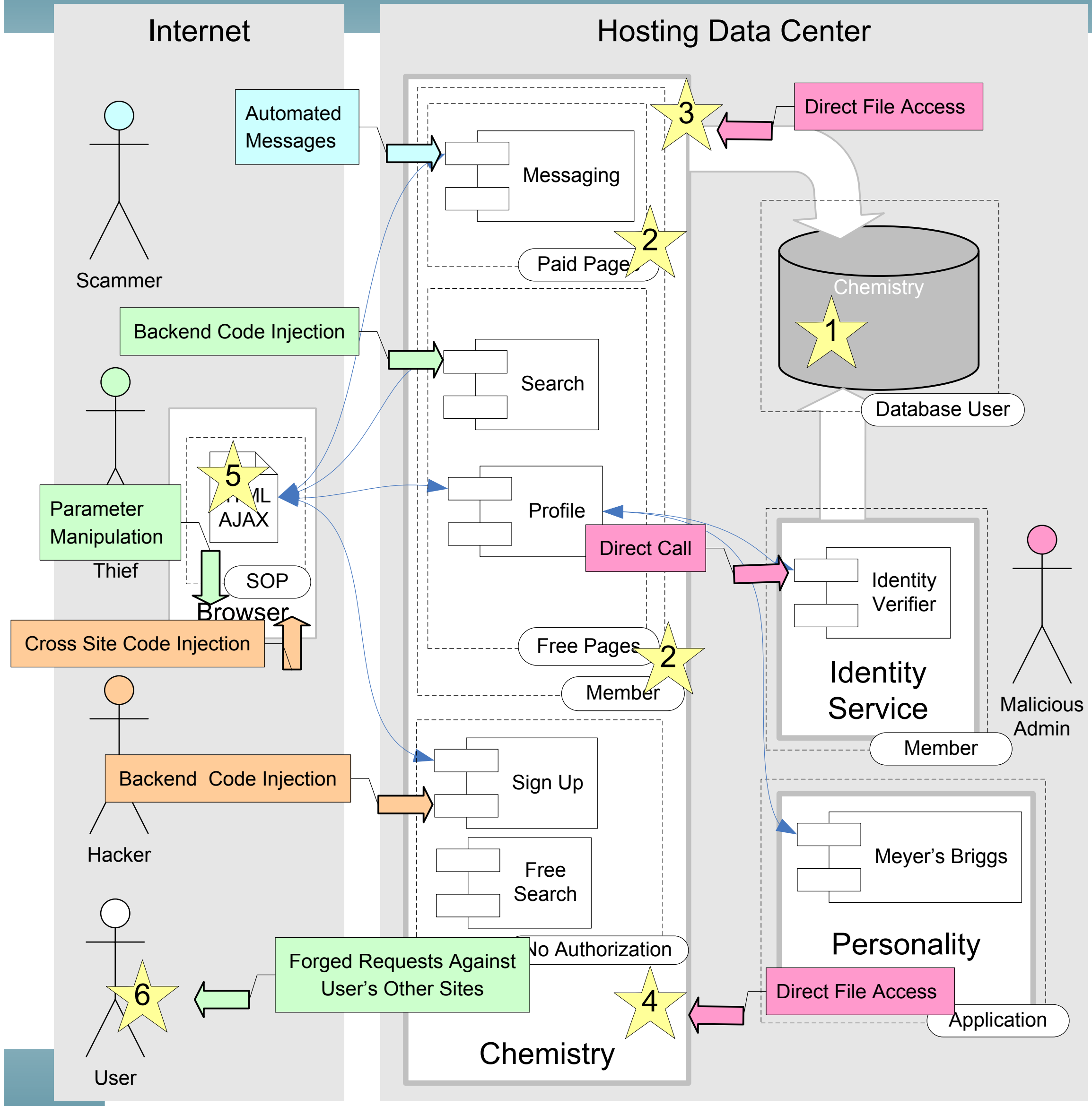
- Threat Agents are users that have malicious intent
- Like users they have capabilities within the system
- Threat Agents have a goal that usually involves subverting a security control

What Are You Trying To Protect?



- Assets are the application's functions
- Assets are the application's sensitive data
- Assets are the application's users, and assets of other systems the user can access

How Will You Be Attacked?



- Associate a Threat Agent with an Asset and determine how to can get to it
- Threat Agents will attack nearest, easiest targets first
- Designers: look to place controls around assets
- Attackers: start with direct attacks and graduate to multi-step

Why Architecture Analysis Is Necessary

Proper Solutions Require Proper Point Of View

Also viewed as tunnel vision, or not seeing the big picture, the wrong perspective, etc.

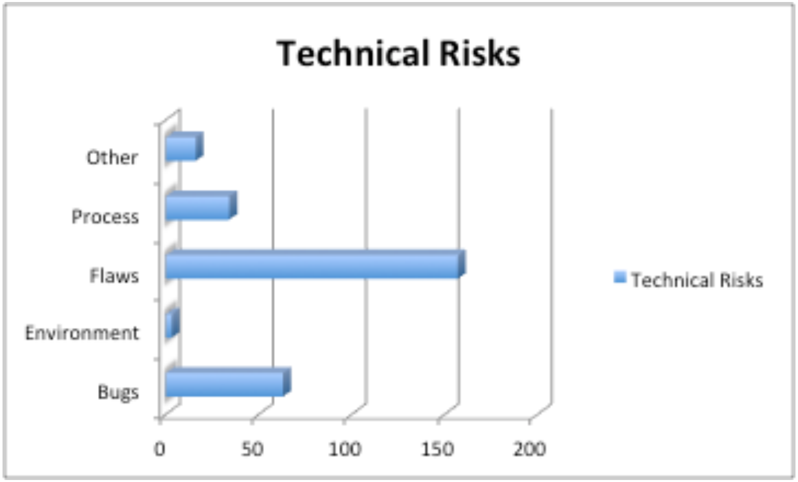
Poor key management example

- PT would likely miss this
- SCR would probably flag it as a key management issue
- AA would fix the design

cigital

History Repeats Itself

ARA Findings Spreadsheet (12 ARAs, 8 clients, mostly web applications, 273 risks identified)



Still screw up client-side trust

Still screw up proper use of crypto

Still fail with password management

Challenges

Challenge – Assumptions Are Evil

Assuming systems are hardened

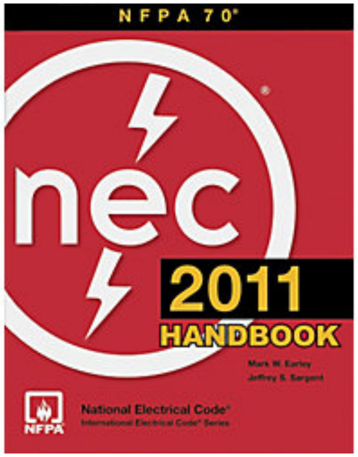
Assuming nothing sensitive is sent to the client

Assuming the fundamentals are done well

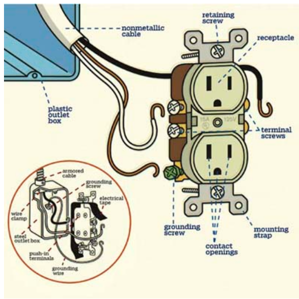
Assuming the overall design has been looked at after the initial design – maybe years ago

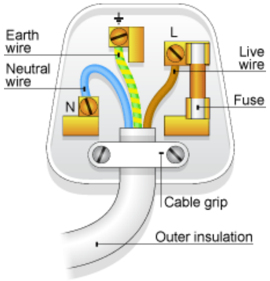
Assuming that because the client has a good process defined, that that process is followed

Challenge – **Some** of This Is Hard Stuff


NFPA 70®
nec
2011
HANDBOOK
Mark W. Earley
Jeffrey S. Sargent
National Electrical Code®
International Electrical Code® Series
NFPA

✔





X

Not just "book" training

Some of this requires apprenticeship

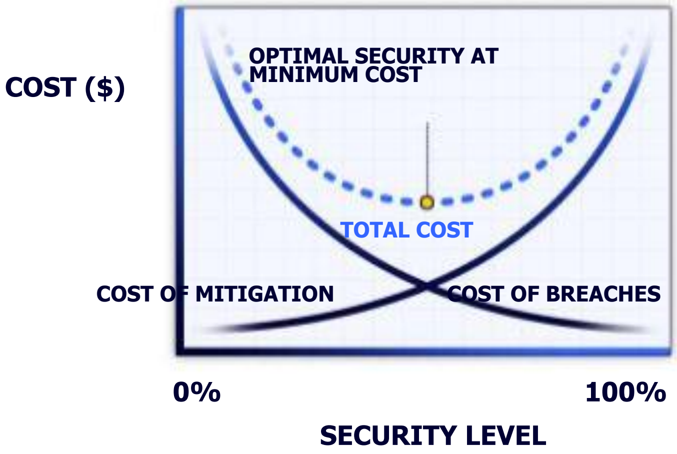
Challenge – Too Much Too Soon

Wrap-Up

Modern Security Is About *Managing* Risks

- There is no such thing as 100% secure
  - Must make tradeoffs
  - Should be business decisions
- Proactive security is about building things right
  - Software security
  - Security in the SDLC
- Security is not a *function*

- Most security problems are caused by software bugs and flaws
- We must build secure software



cigital

Architecture Analysis Wrap-Up

- Helps you find flaws
- Does **NOT** replace other techniques
- Human-driven analysis (minimal tool support)
- Some fixes require long-term solutions
  - Risk mitigation is key
- Apprenticeship

cigital

Thank You