Common Architecture Flaws

Problems in Identity, Authentication, & Authorization

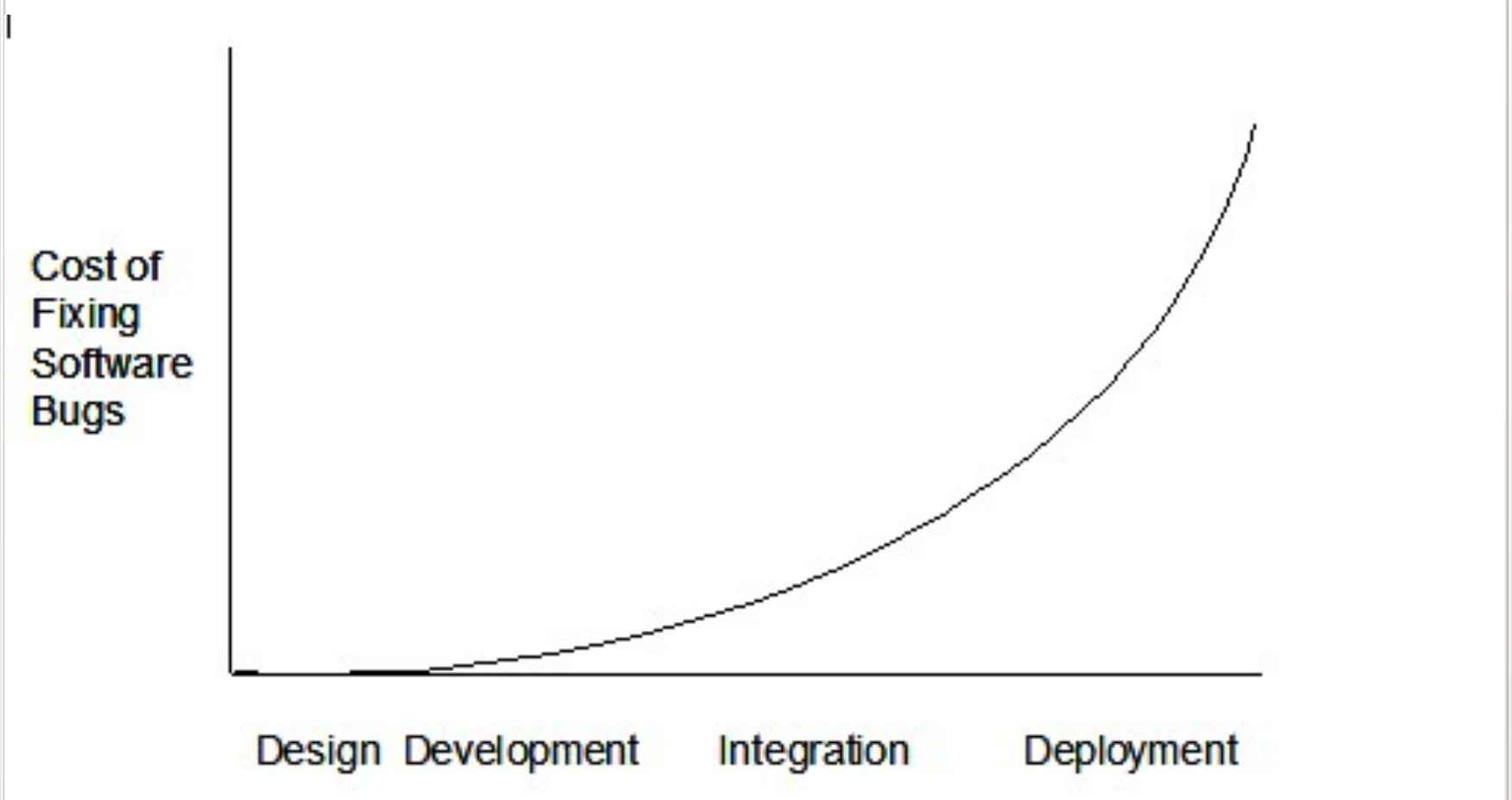
-jOHN (Steven)
Internal CTO, Cigital Inc.
Principal Consultant
@m1splacedsoul

Why are Flaws Important?

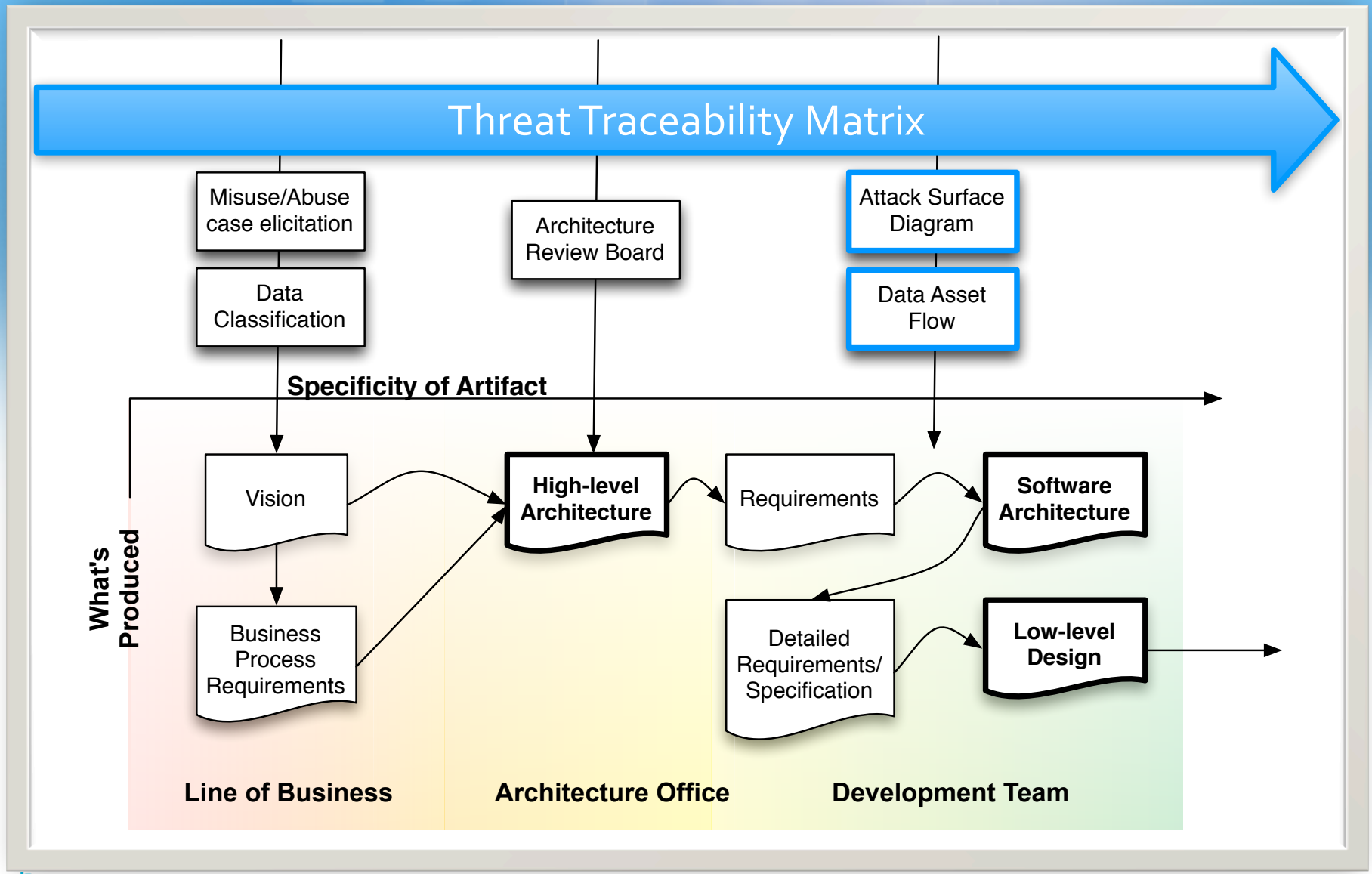
The "NASCAR EFFECT" -- It's Fun

- Study how systems fail
- Build for failure
- "Moving left"
  - Time
  - Money
  - Scale

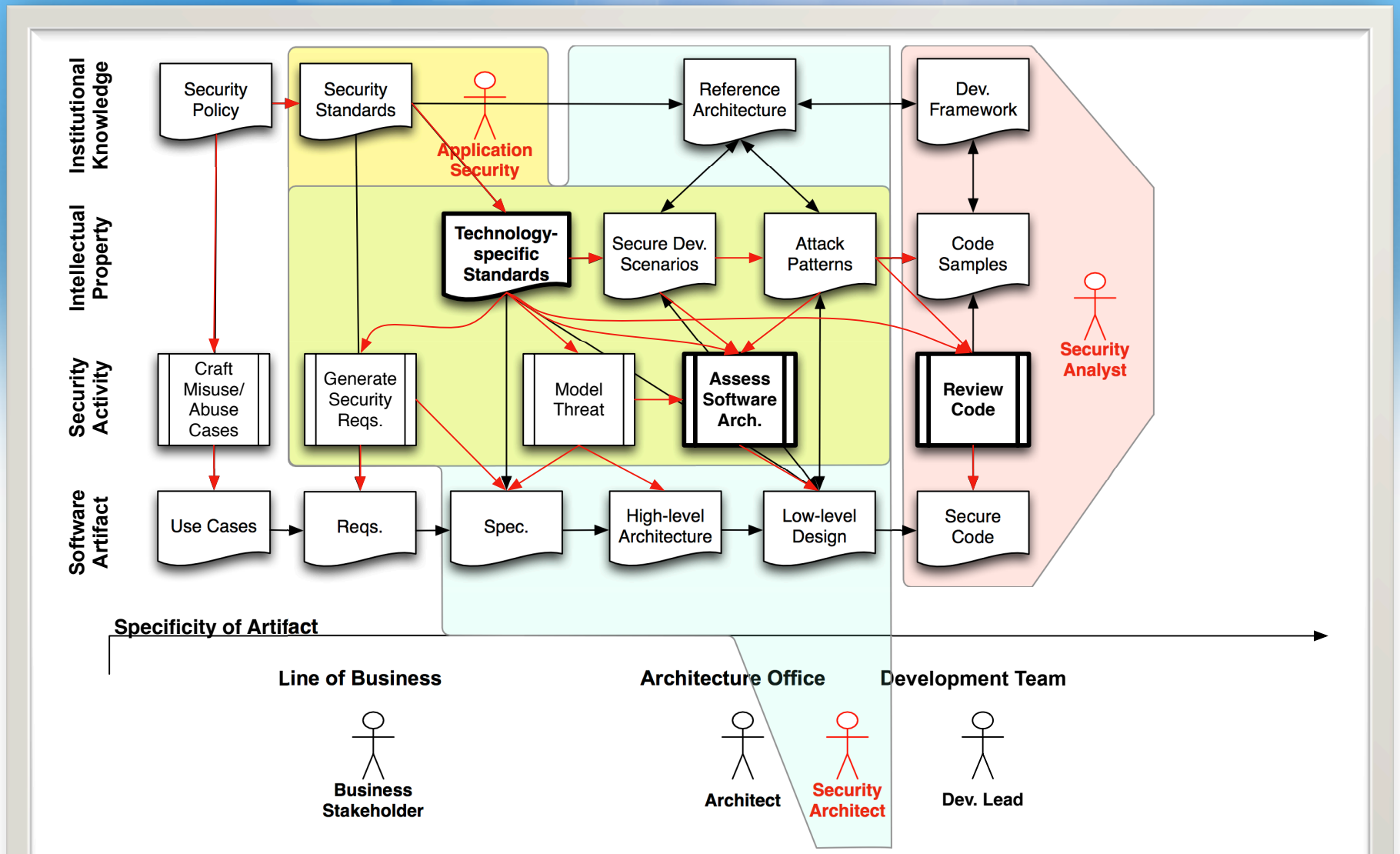
Cost to fix



Moving Left (Conceptually)

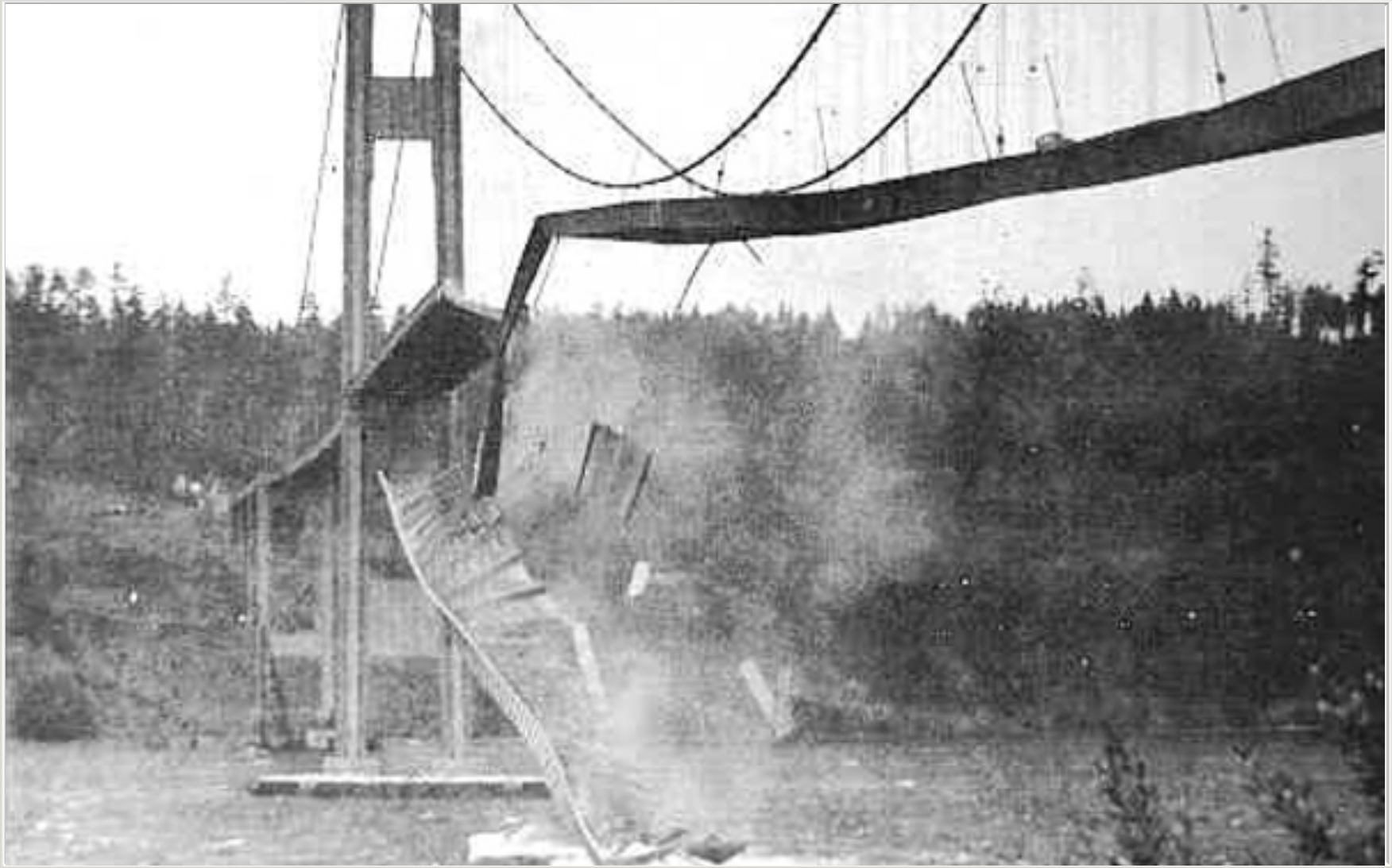


Moving Left (Large Org)

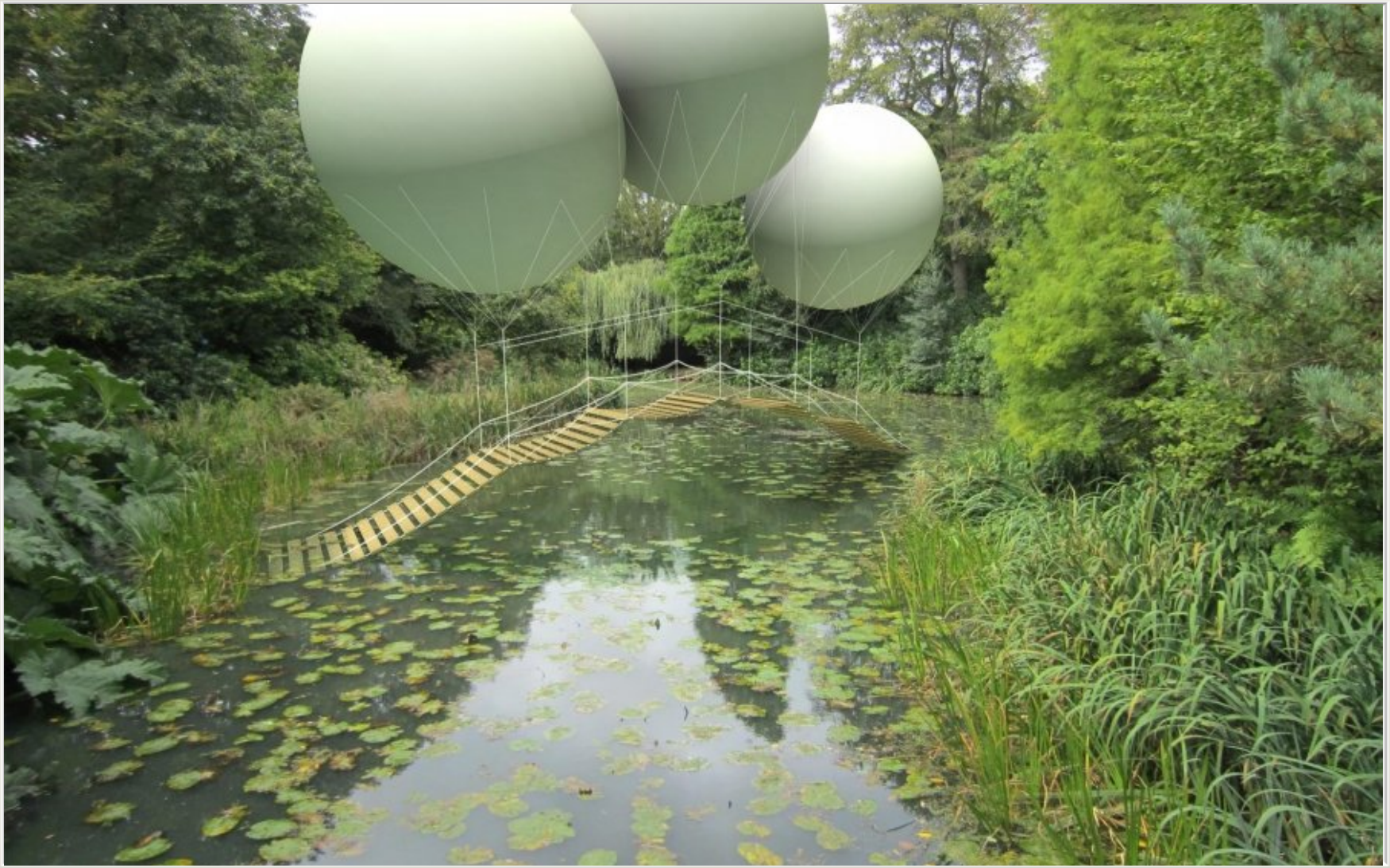


Bug

Flaw

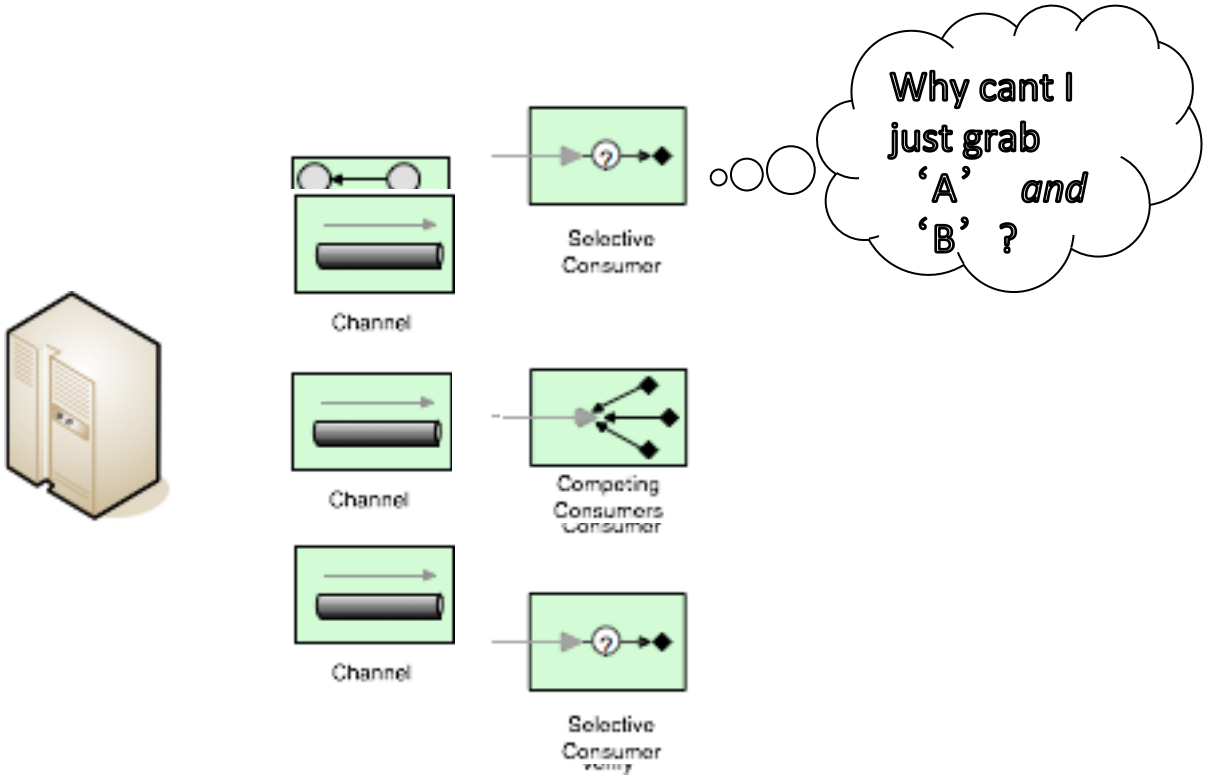


Flaw

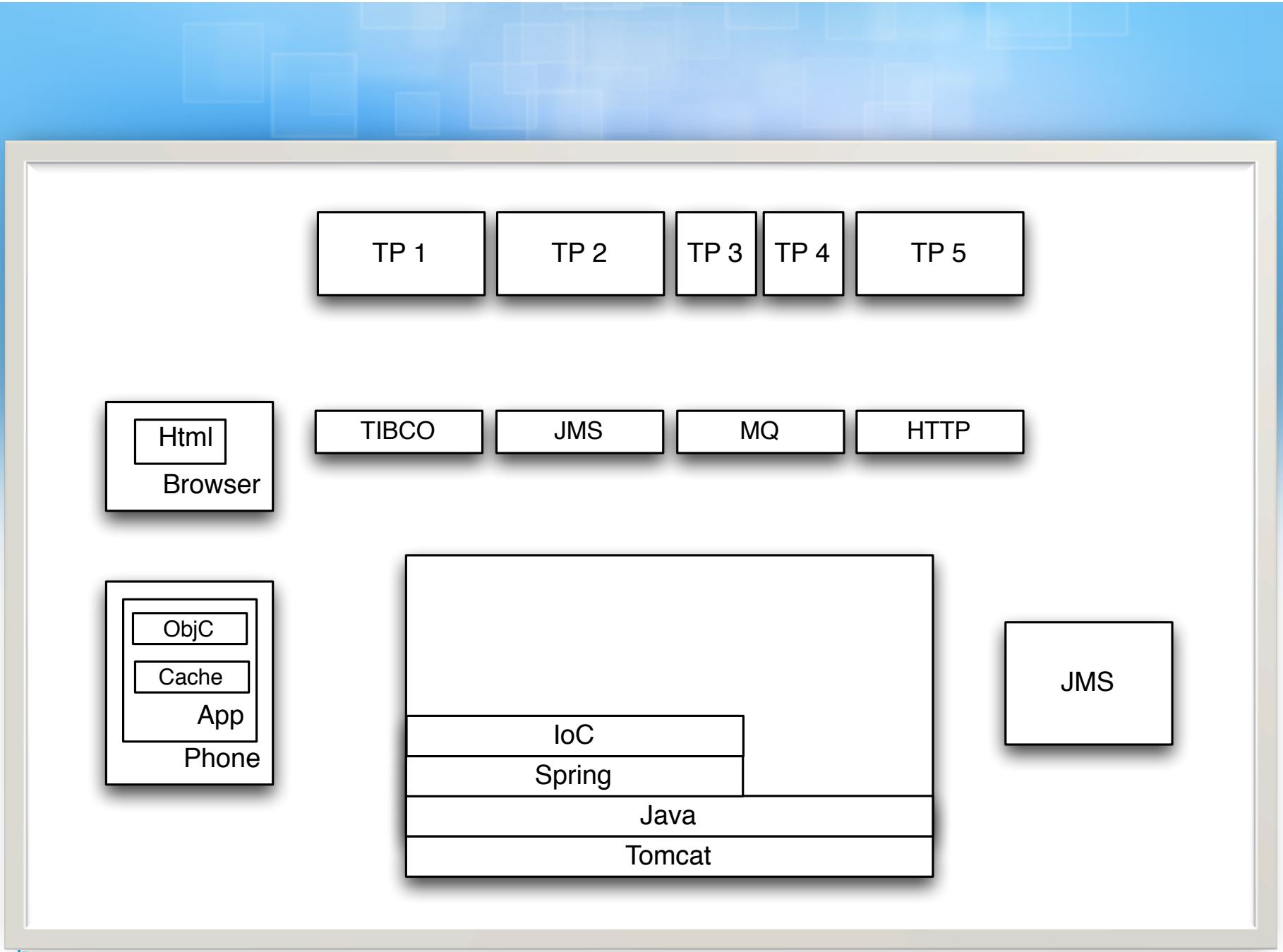


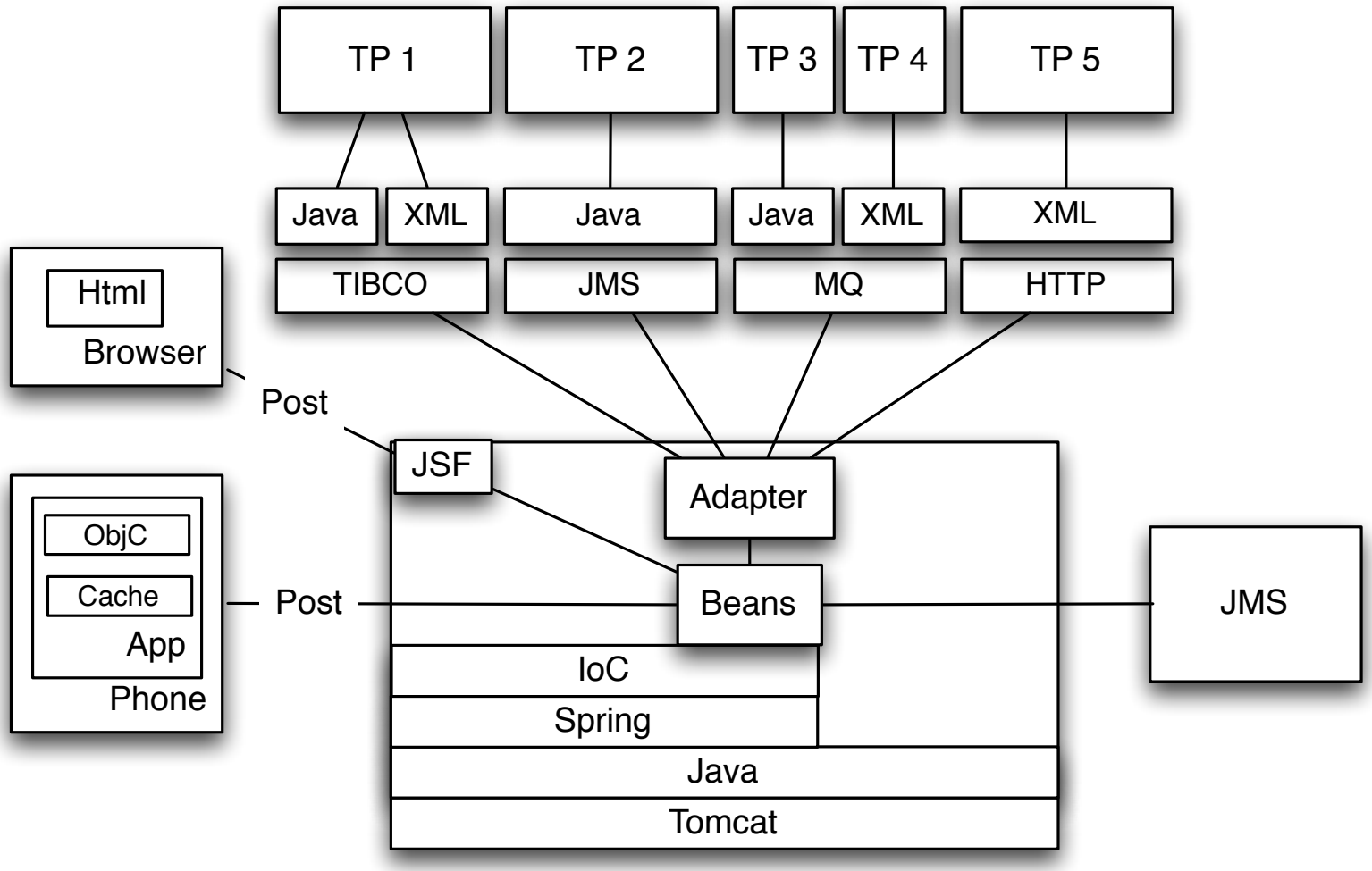
Flawed?

System Flaw: Principal Mapping



cigital


TP 1
TP 2
TP 3
TP 4
TP 5
Html
Browser
TIBCO
JMS
MQ
HTTP
ObjC
Cache
App
Phone
IoC
Spring
Java
Tomcat
JMS


TP 1
TP 2
TP 3
TP 4
TP 5
Java
XML
Java
Java
XML
XML
TIBCO
JMS
MQ
HTTP
Html
Browser
Post
JSF
Adapter
ObjC
Cache
App
Phone
Post
Beans
JMS
IoC
Spring
Java
Tomcat

Output Encoding Bugs

Contextually Aware Output Encoding

ESAPI

```
<%-- Must escape content (even in user names!) --%>
Hello <%= ESAPI.encoder().encodeForHTML(user.getName()) %>!

<%-- Must escape 3 different contexts correctly --%>
<img src="/profile-photo?user=<%= ESAPI.encoder().encodeForURL(user.getId()) %>"
   alt="<%= "Photo of "+ESAPI.encoder().encodeForHTMLAttribute(user.getId()) %>"
   onclick="<%= "openProfile('"+ESAPI.encoder().encodeForHTMLAttribute(
            ESAPI.encoder().encodeForJavaScript(user.getId())) + "')" %>" />

<%-- Outputting unescape, is however, easy: --%>
<%= user.getProfileHtml() %>
```

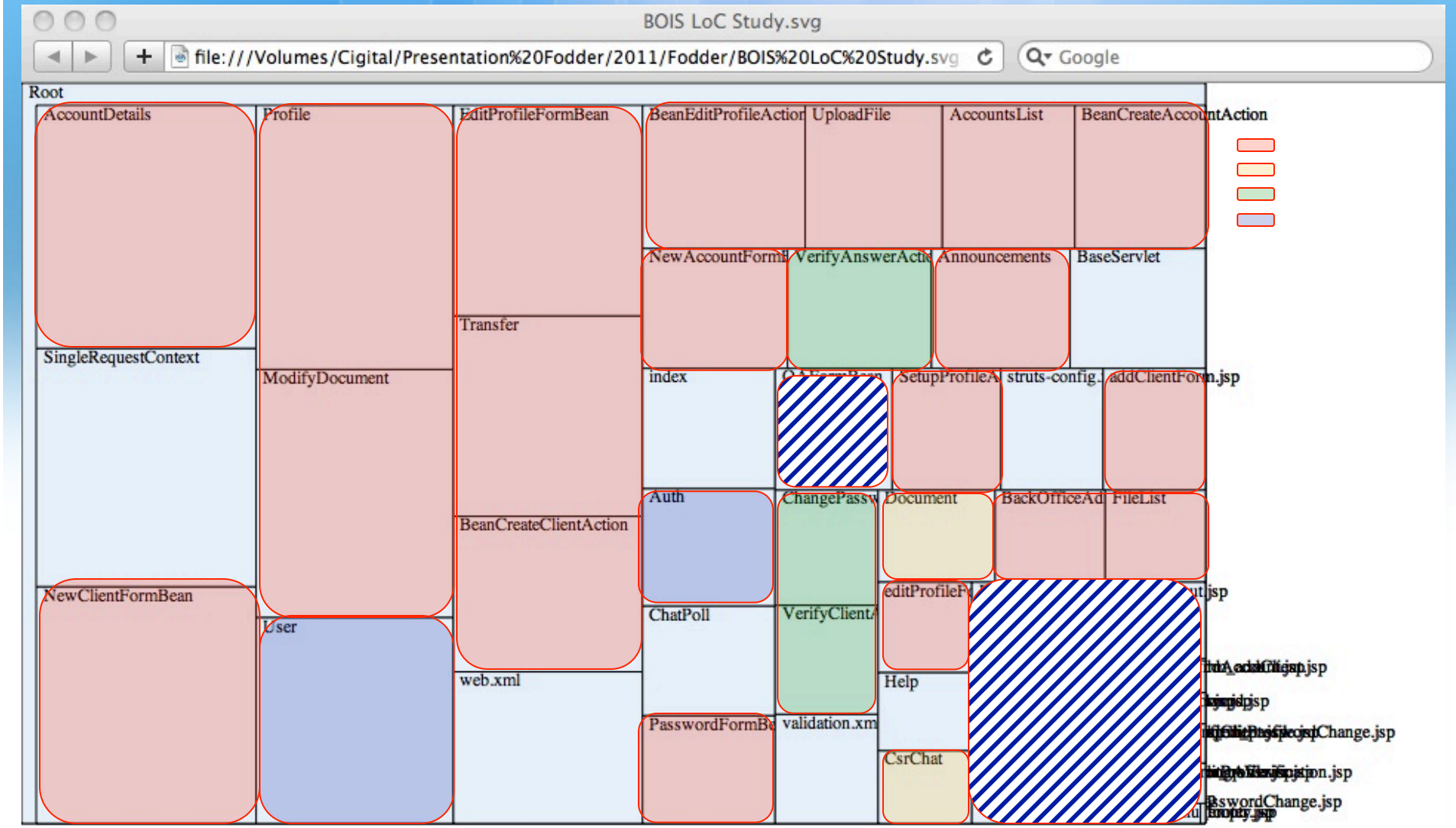
JXT

```
<!-- Automatically escaped content -->
Hello ${user.getName()}!

<!-- Example tag with 3 different contextual encoding requirements -->
<img src="/profile-photo?user=${user.getId()}"
   alt="Photo of ${user.getName()}"
   onclick="openProfile('${user.getId()}')" />

<!-- Override the default escape, rare, but occasionally needed: -->
<jxt:out value="${user.getProfileHtml()}" escape="none"/>
```

cigital

Fixing Output Encoding with OWASP ESAPI

("Blast Radius" – Struts1)



Change 70% of files, representing 78% of KLOC

cigital

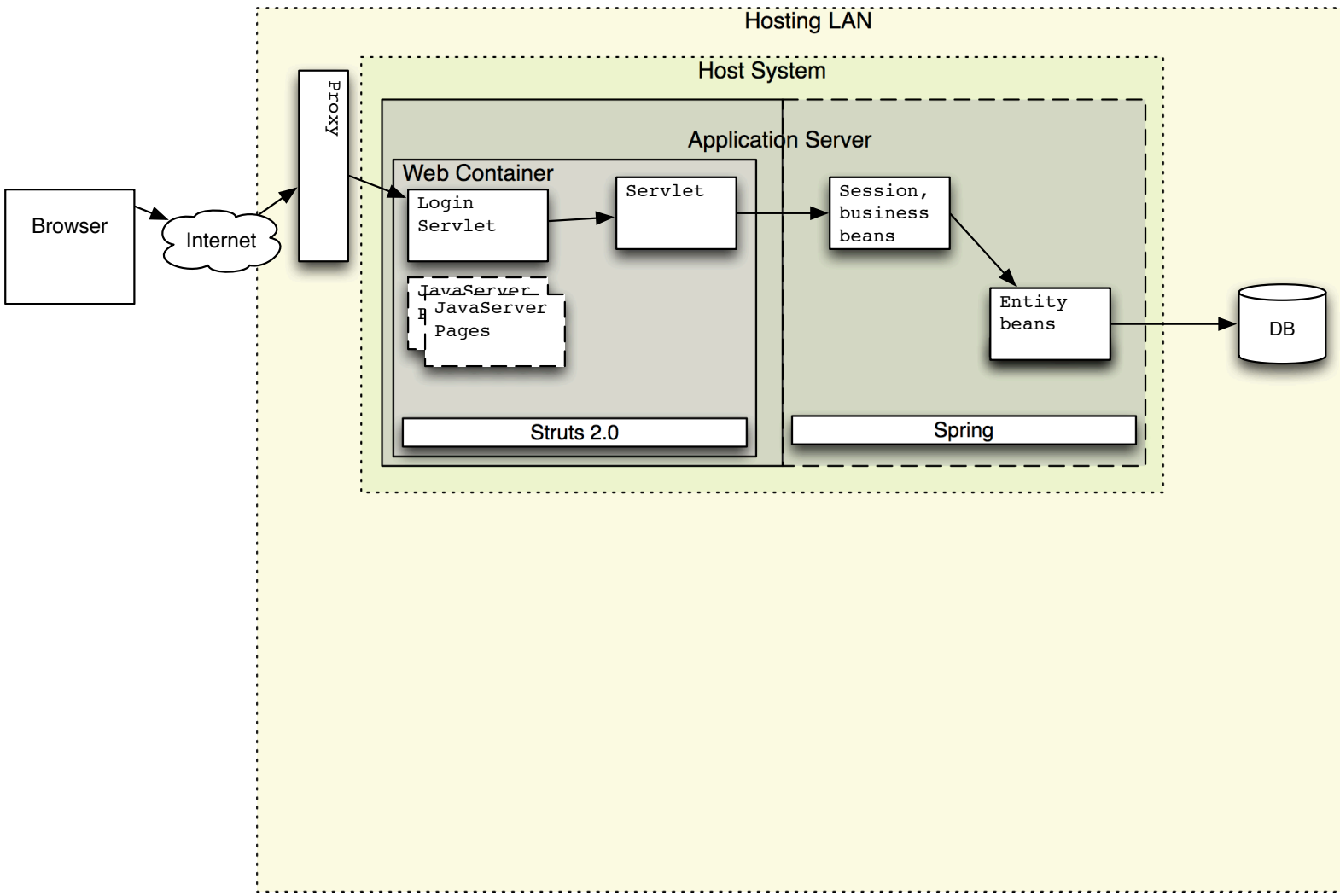
Bugs vs. Flaws

- Names are not important
- What is important is the:
  - Stakeholders engaged in the fix
  - Techniques used to fix the problem
  - Scope/scale at which the fix is applied
- If fixing a bug entails improving *how* something is implemented, fixing a flaw improves *what* it is.
  - …opening a new set of implementation bug opportunities;-)

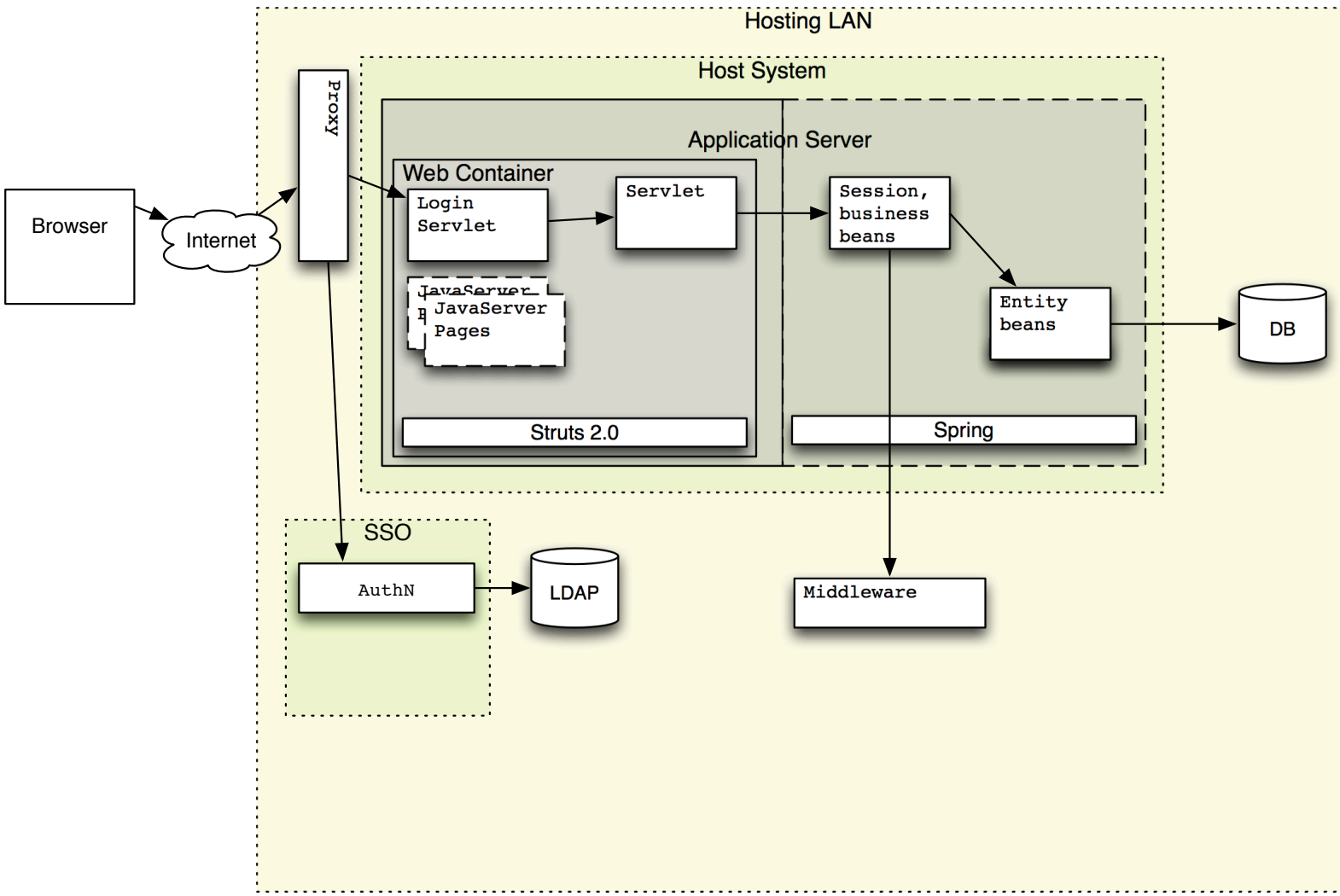
Common IAM/Auth[N|Z] Flaws

Principal Identity

Propagating Principal: Most Basic Form



Federated Systems



Federated Systems

Browser → AuthN

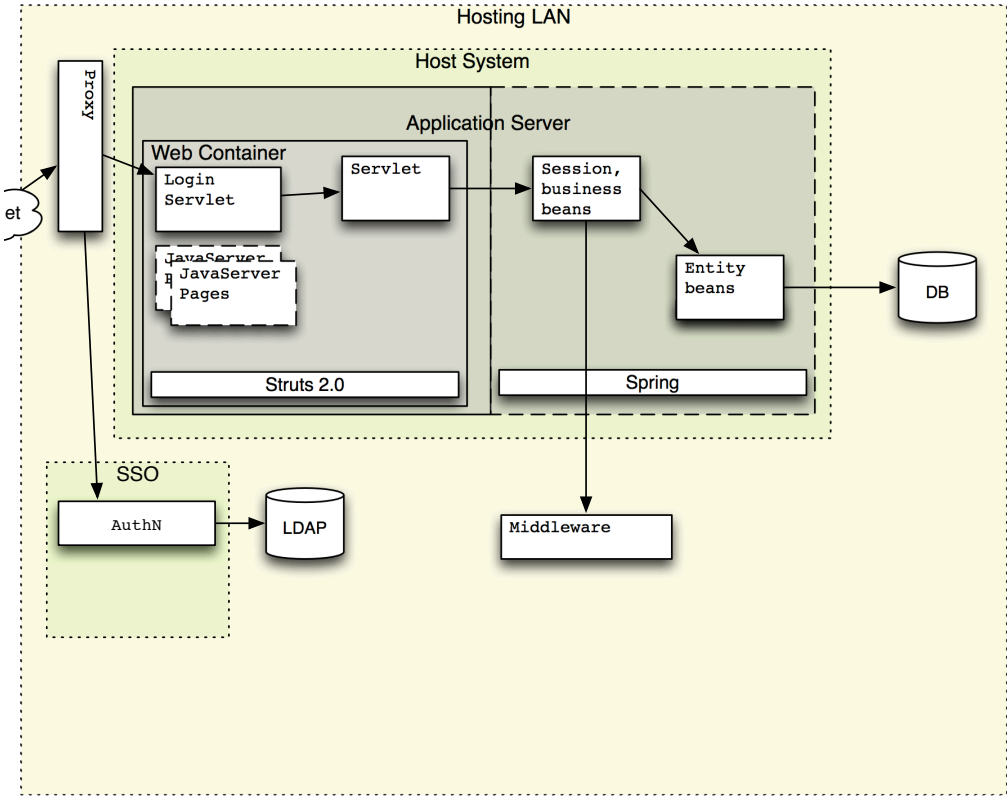
- User-level: UN/PW
- Creds → UID + Session

Browser → Container

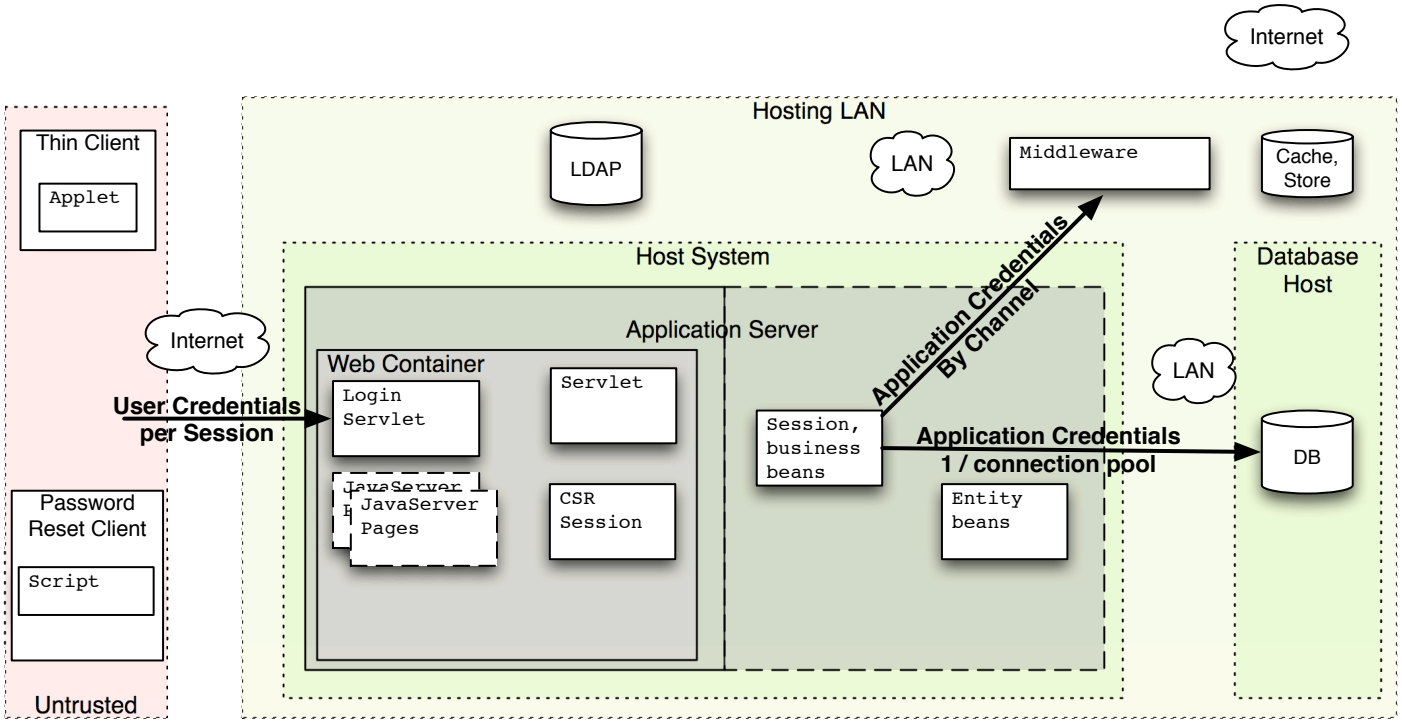
- Binary AuthN: session
- Optional RBAC

Container → DB

- Host-level AuthN
- Optional RBAC



Principal Resolution Changes...



cigital

Consequences: AuthZ Foiled

1. Authenticated requests can access anything
    1. Forced browsing
    2. Parameter tampering, pollution, and so forth
    3. Replay attacks

2. Containers lack info required for AuthZ
    - Role is too coarse to mitigate account access
    - UID lacks user context
    - Access control list lies in directory or DB
    - Requests carry no claims-based info

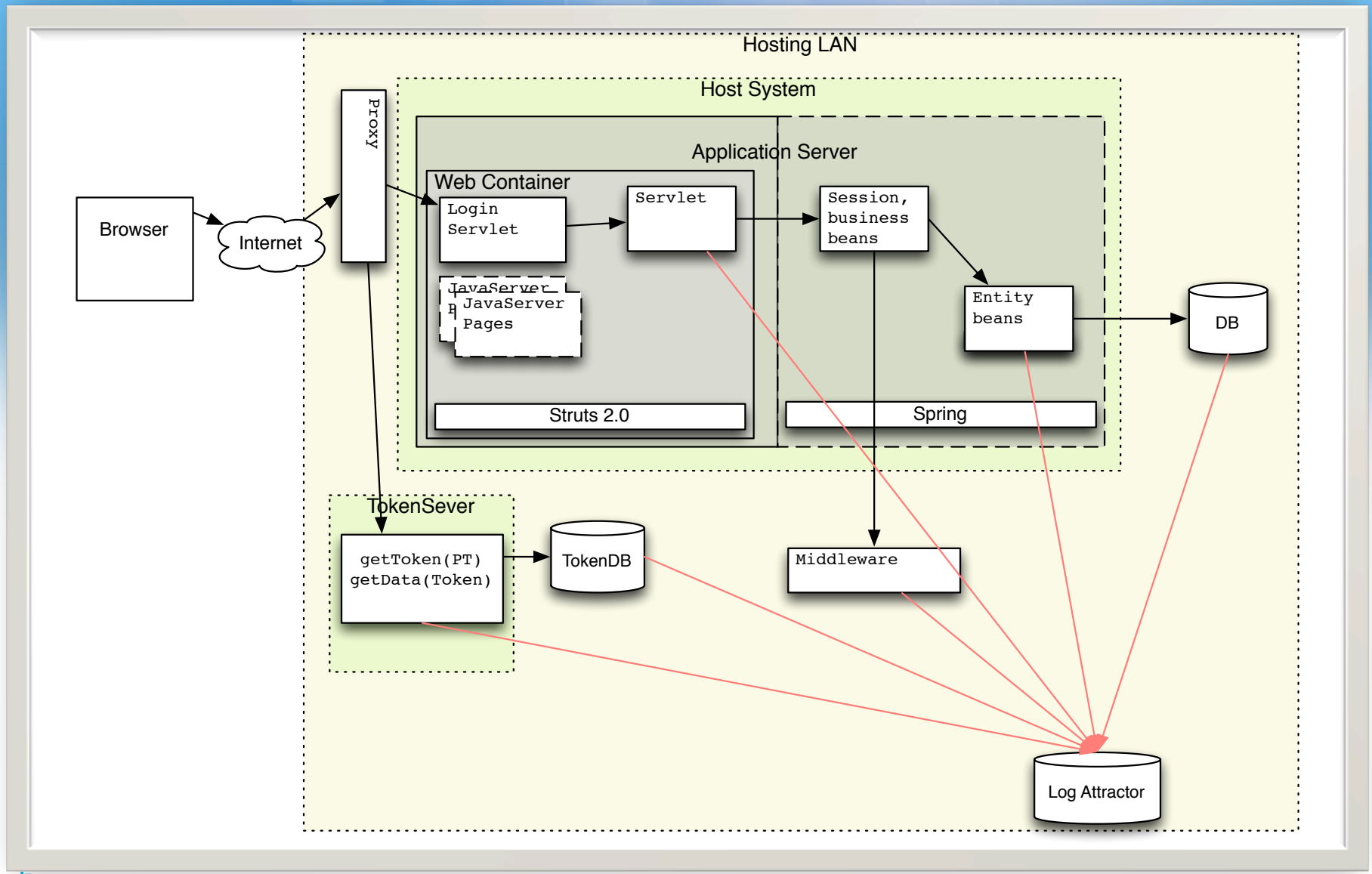
Principal Resolution counter Privilege

Asset access & privilege increase moving 'right'

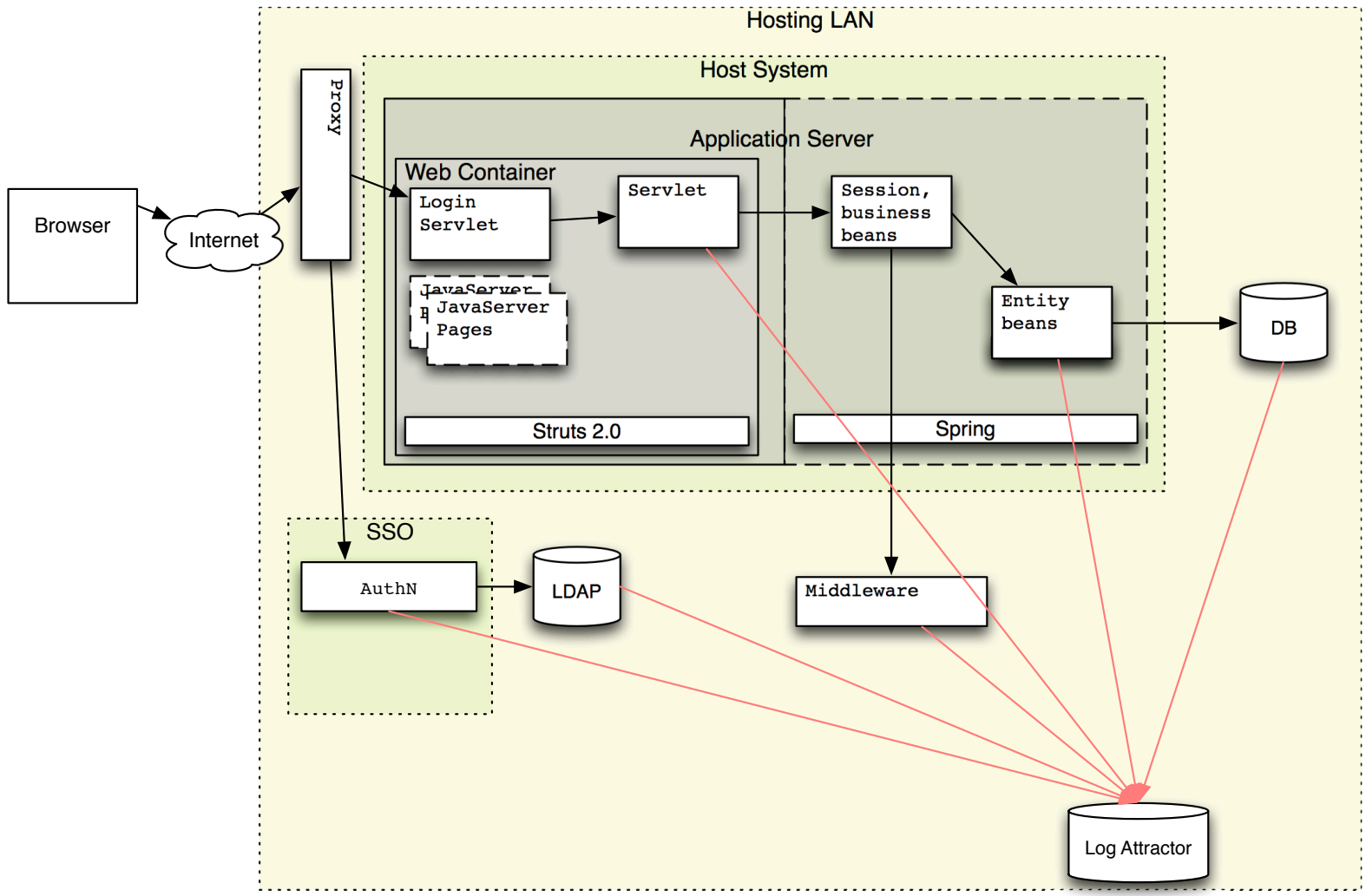
Information available to assert privilege decreases

…this, BTW, drove CC# as ID years ago.

Tokenization



Tracking Users



Solution: DMV?!

Centralize identity provision

- Force requests to carry ID
- Multiple verifiable elements

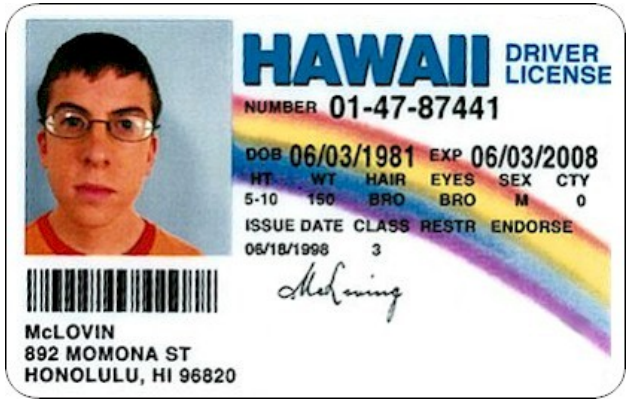
- Accepted everywhere w/in federation
- Accepted at foreign crossings as well

Verify

- Principal and ID match
- Principal is expected (e.g. guest list)

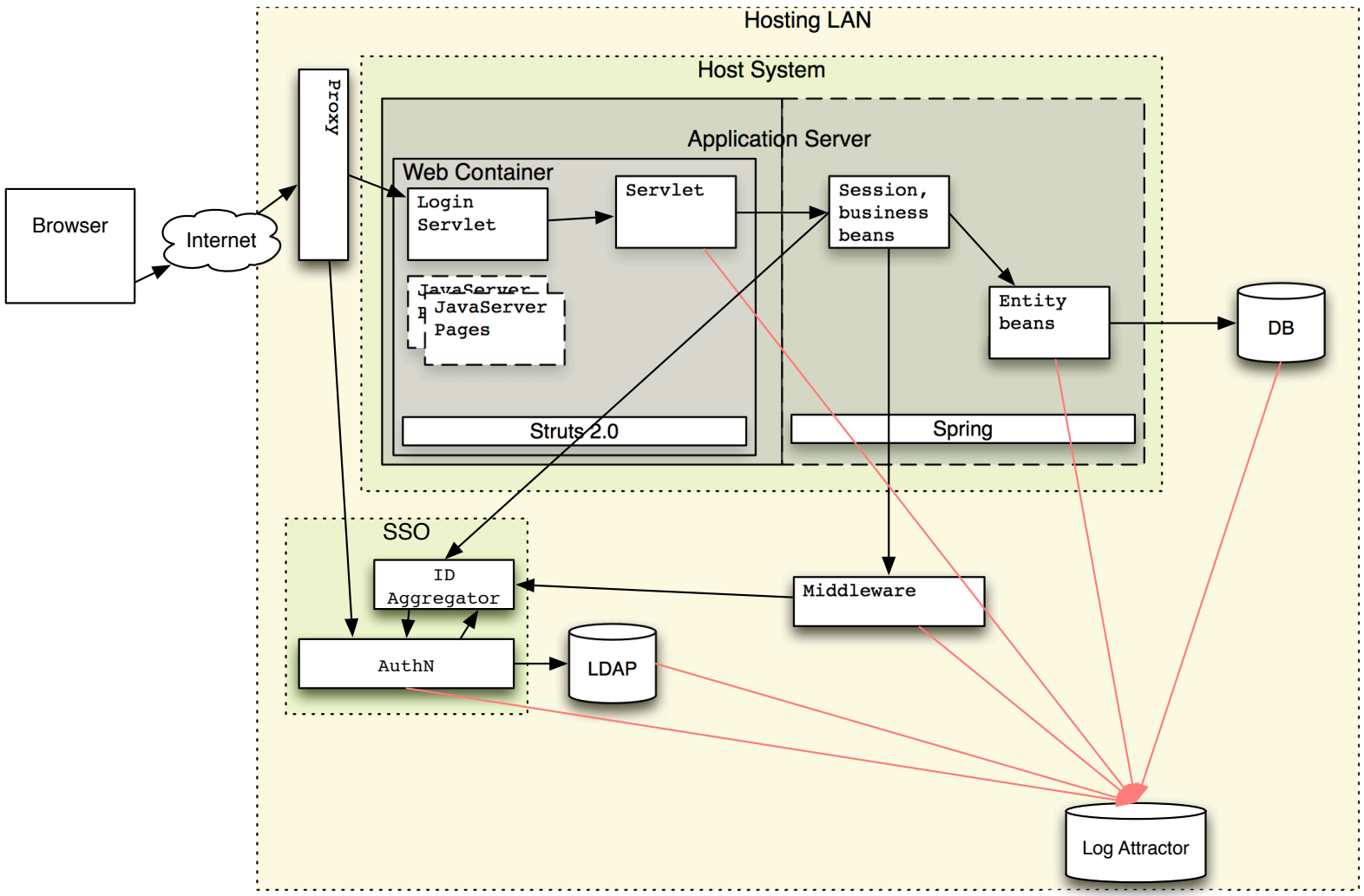
Quick verify

Costly creation/provision

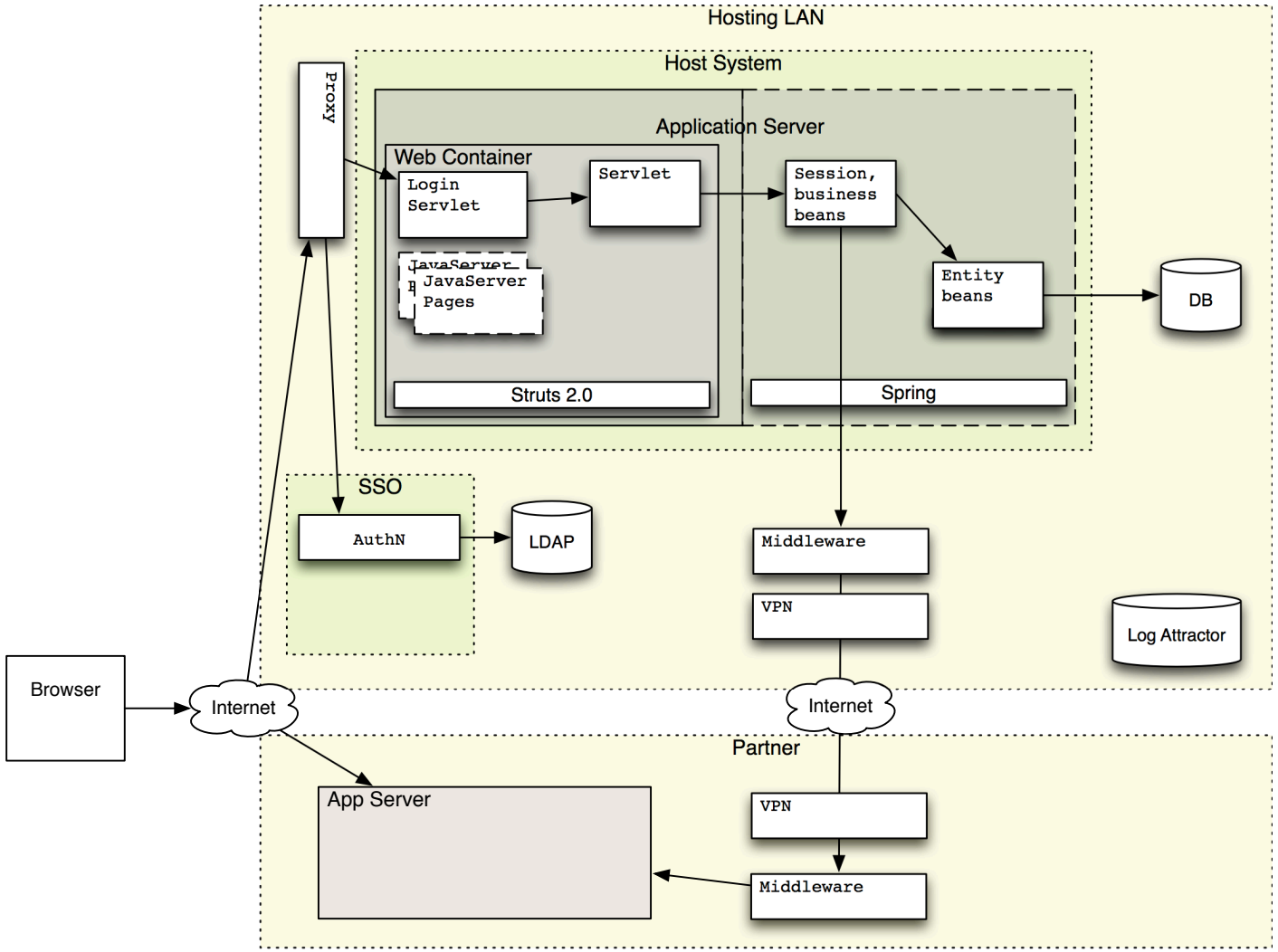


May carry (optional) endorsements as necessary / appropriate

cigital

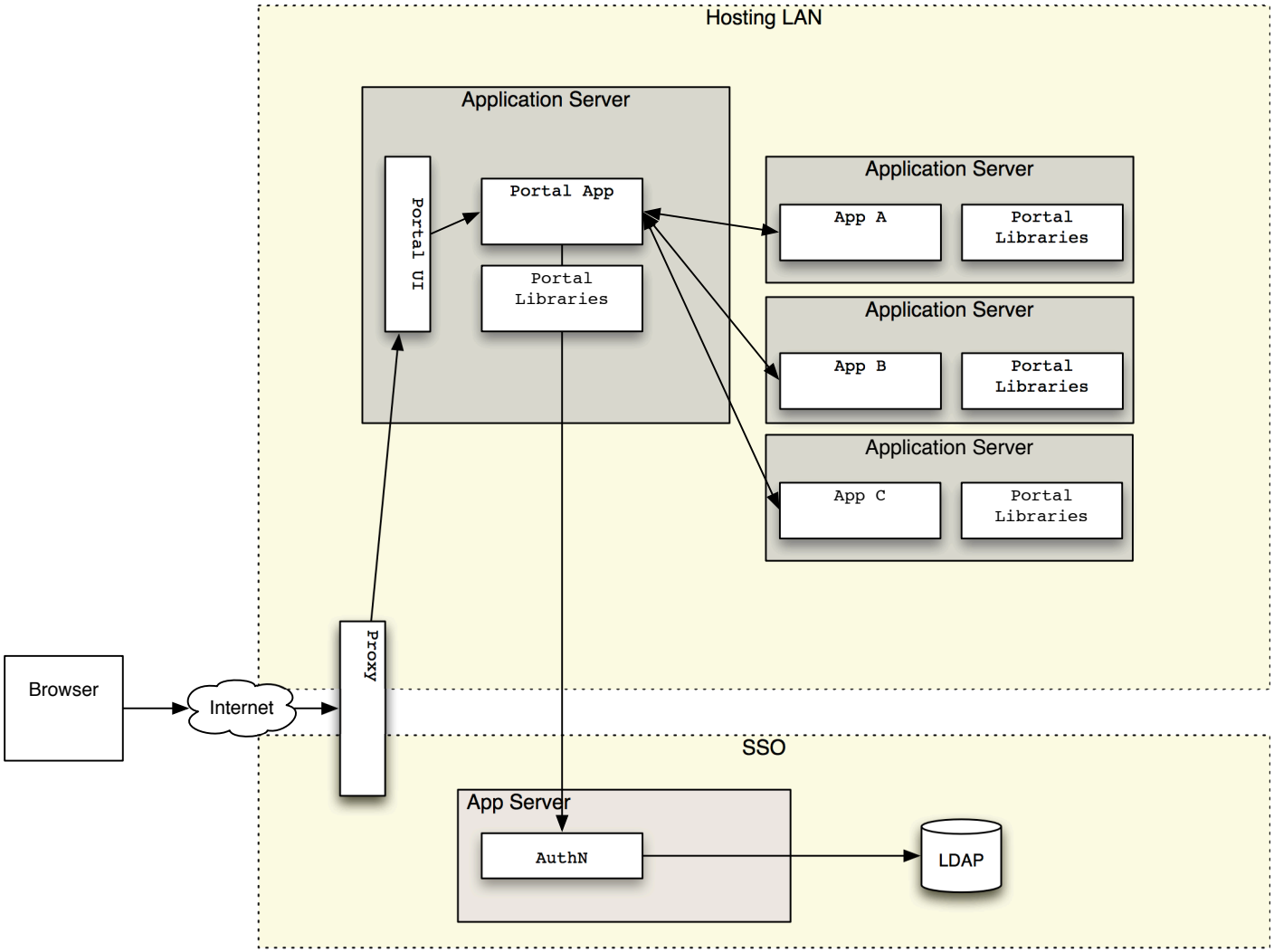


Identity extends beyond org. boundaries

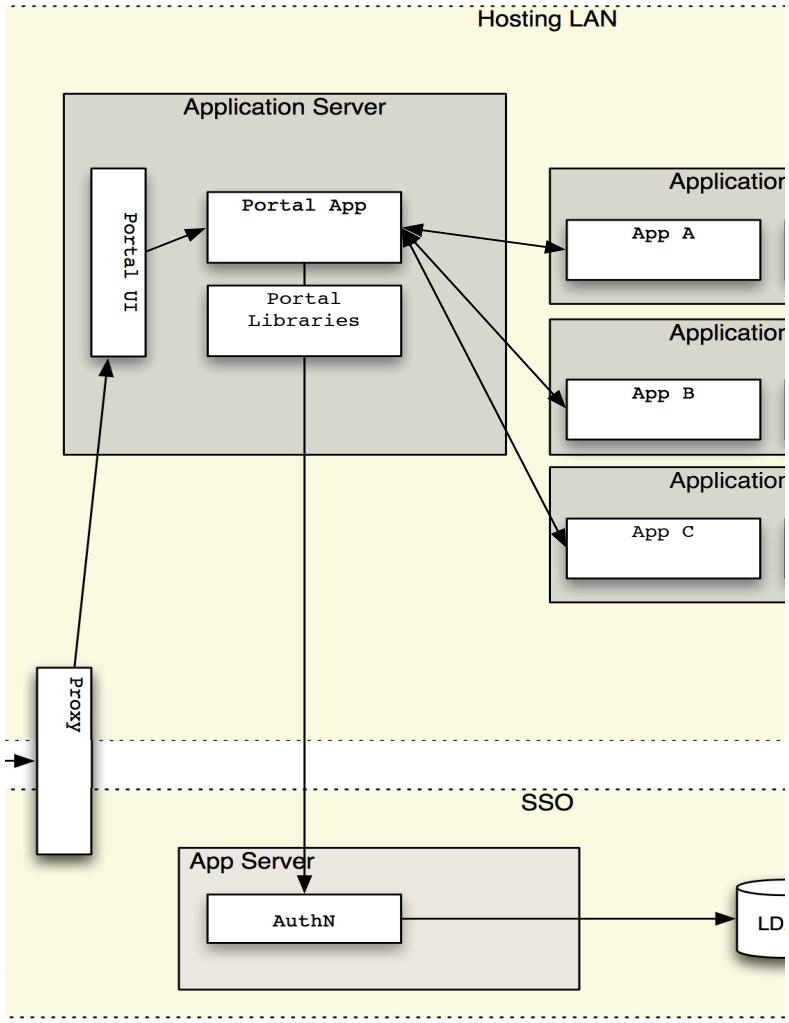


Scope & Termination

Context: Common Portals & Mash-up Sites



Context: Common Portals & Mash-up Sites



AuthN & Portal UI collaborate

- Conduct login workflow
- Associate session w/ UID

User navigates to App X

- Portal calls AuthN
  - Check session validity
- Checks UN valid for realm
- Hands control to App X

App X

- Checks UN valid for App

Consequences

- Decoupling Session Management Log-in/out means
  - Application doesn't know about:
    - Timeout
    - Logout (sometimes)
    - User Termination/Deletion events
  - App can't participate in work flows

cigital

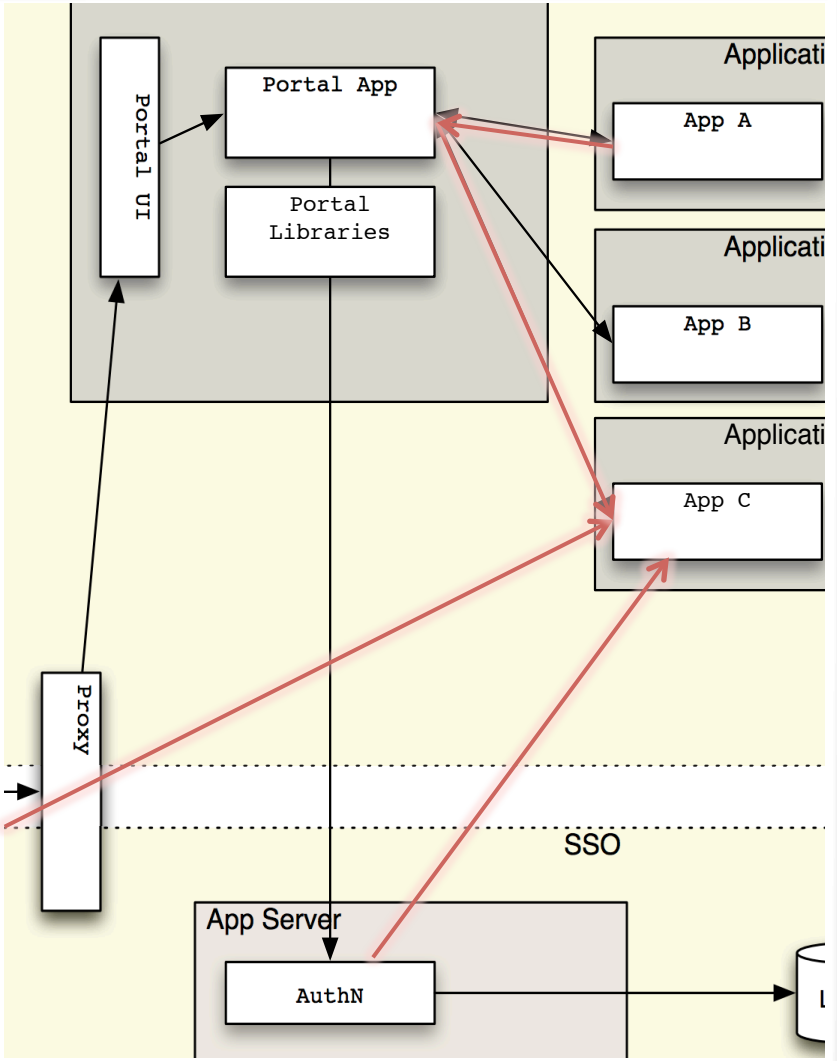
Visually…

AuthN can't talk to AppC

AppC must replicate behavior

- AuthN (Session)
- Portal (User maps, workflow)

Portal Can't talk to AppC w/o valid request



Generate Single Scope Handles

AuthN system generates:

- Application-specific sessions, in concert with
- Portal-specific identity

AuthN system formats specific sessions

- <session ID> ':' <app ID>

Unfortunately, existing products don't support this out of the box

Solution: Callbacks w/ UUID

AuthN system communicates with App

- (Pull) Application polls AuthN for session properties
- (Push) AuthN makes requests 'pushing' session events

The application can:

- (pull) Query AuthN for session tuple get back answer
  - Centralizes ACLs, PDP
- (push) AuthN annotates request
  - Annotation sufficient to make decisions
  - UUID → APP_SESSION_UUID
  - XACML, JSON, etc.

Coopt the User for Fraud Detection

Context

AuthN workflows have become complex

- Discern computer/human
- Implement Multi-"factor" authentication
- Apply 'risk-based' workflow based on client
  - *** Known clients get 'easier path'

Fraud systems interact with the login workflow

- Systems involve users in workflow
- Systems support notifications

Problem

Complexity breeds errors

- Workflow state machines often broken
- Confusing end-point registration systems proves easy
- Multi-factors are redundant

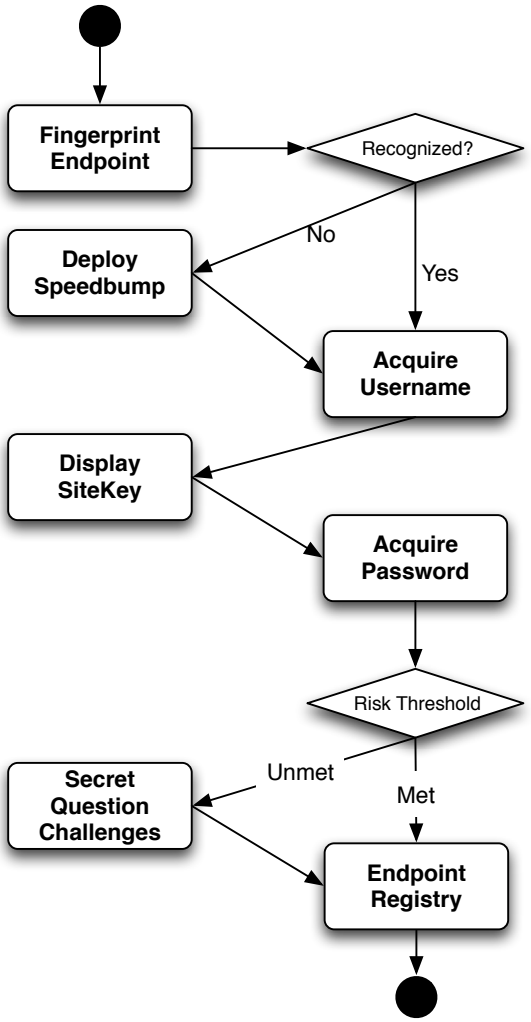
Attackers always pick "shortest path"

- Attack a registered end-point
- Spoof a common end-point (IOS)

Privilege / Trust are sticky

- How long is trust appropriate?
- Is there a way to revoke it?

Common Practice



Intended Purpose

- Identify client endpoint
- Prevent brute force attack
- Identify user
- Validate server (anti-phishing)
- Validate user
- Evaluate risk
- Validate user (further)
- Ease login process

cigital

Solutions → Problems: Fingerprint

Fingerprint efficacy based on device

- IOS is low entropy (almost always matches)
- Firefox, Opera are so unique they give you away

Browser fingerprint is a biometric misnomer

- Something you have vs. something you are
- Control becomes liability w/ mobile device
  - Specially w/ Safari

Solutions → Problems: Speedbumps

Remove these for a mobile device?

- Keyboard & Autocorrect too annoying…

Remove for registered fingerprints?

- Server has seen this device, associates it w/ user…

Differentiate human vs. script

- Control becomes liability w/ mobile device theft
- Many schemes vulnerable to mining attacks

SiteKey: designed to assure user speaking to server directly

- Again: mining attacks

Solutions → Problems: Secret Questions

Another multi-factor conflation

- Duplicate "something you know"

Conflates

- Additional assertions about the user vs. endpoint

Key Scheme Improvements

Improve Fingerprinting

- Focus around only device, not user
  - This can't replace computer/human detection or theft
- Use access patterns
  - Telemetry, location (change is as useful as value)
  - Time, speed, etc.

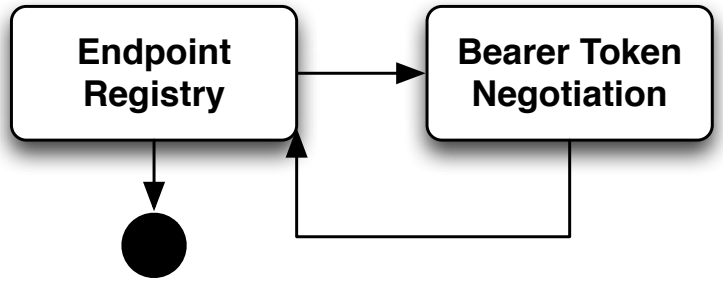
Trust once…

Many systems are add only

- No audit list
- No removal

This is bad for fingerprints

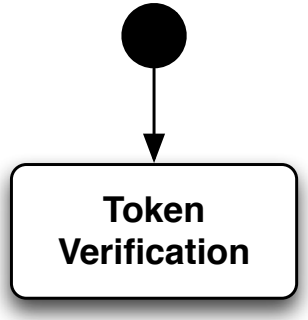
This is fatal for bearer tokens



"Trust" should not be binary

…and not for multiple purposes

- Fingerprinted mobile device != OOB Channel



cigital

Key Scheme Improvements (2) - Involve User

Provide the user the ability to label endpoint

Provide a list of end-points, enable user disposition

- Do not think of as a sliding bar (black, grey, white)
- Actions may include:
  - Do not allow
  - Notify
  - Request addl. verification
  - Reduce access
  - Omit some verifications

Provide OOB notification, include:

- Fingerprint data
- Time
- Actions taken

Binary vs. Rings of Trust
The "Jelly Donut Architecture"

Castles, like me, are misunderstood

http://www.flickr.com/photos/sugarmonster/

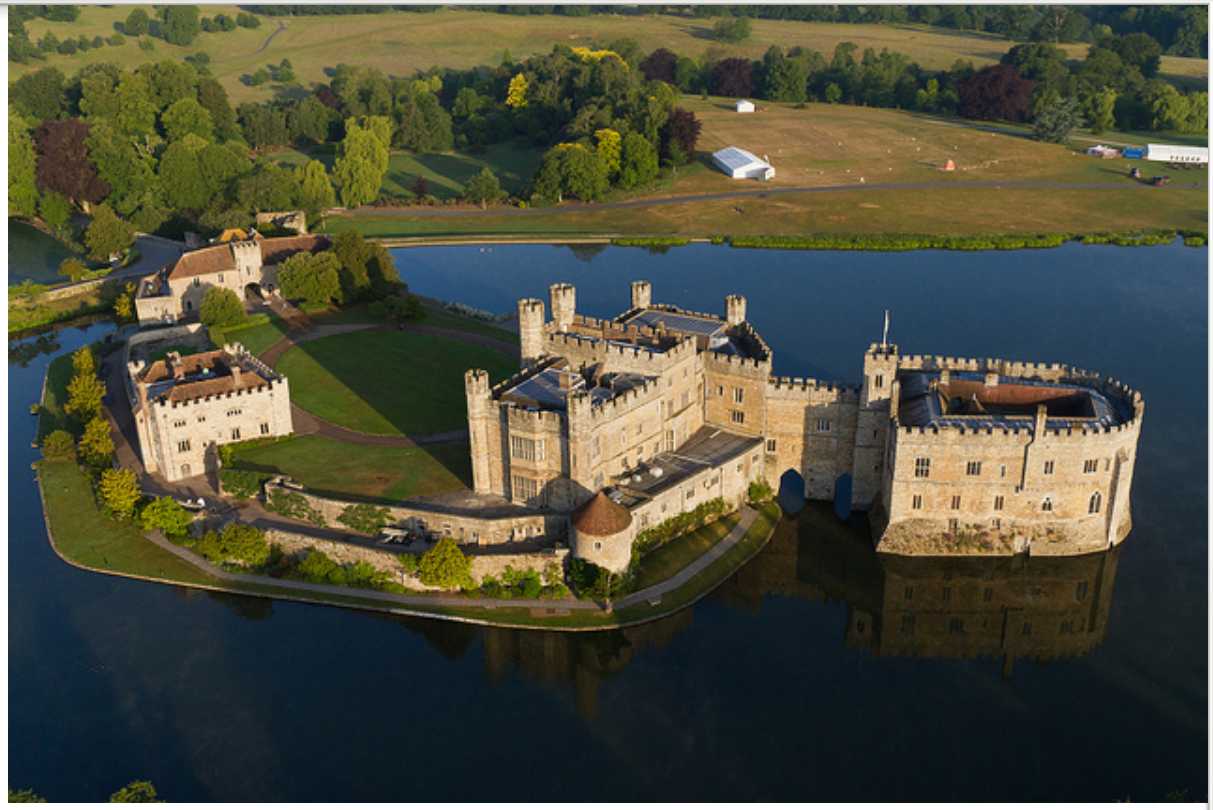
Barbican

Town

Bailey

Building

Keep

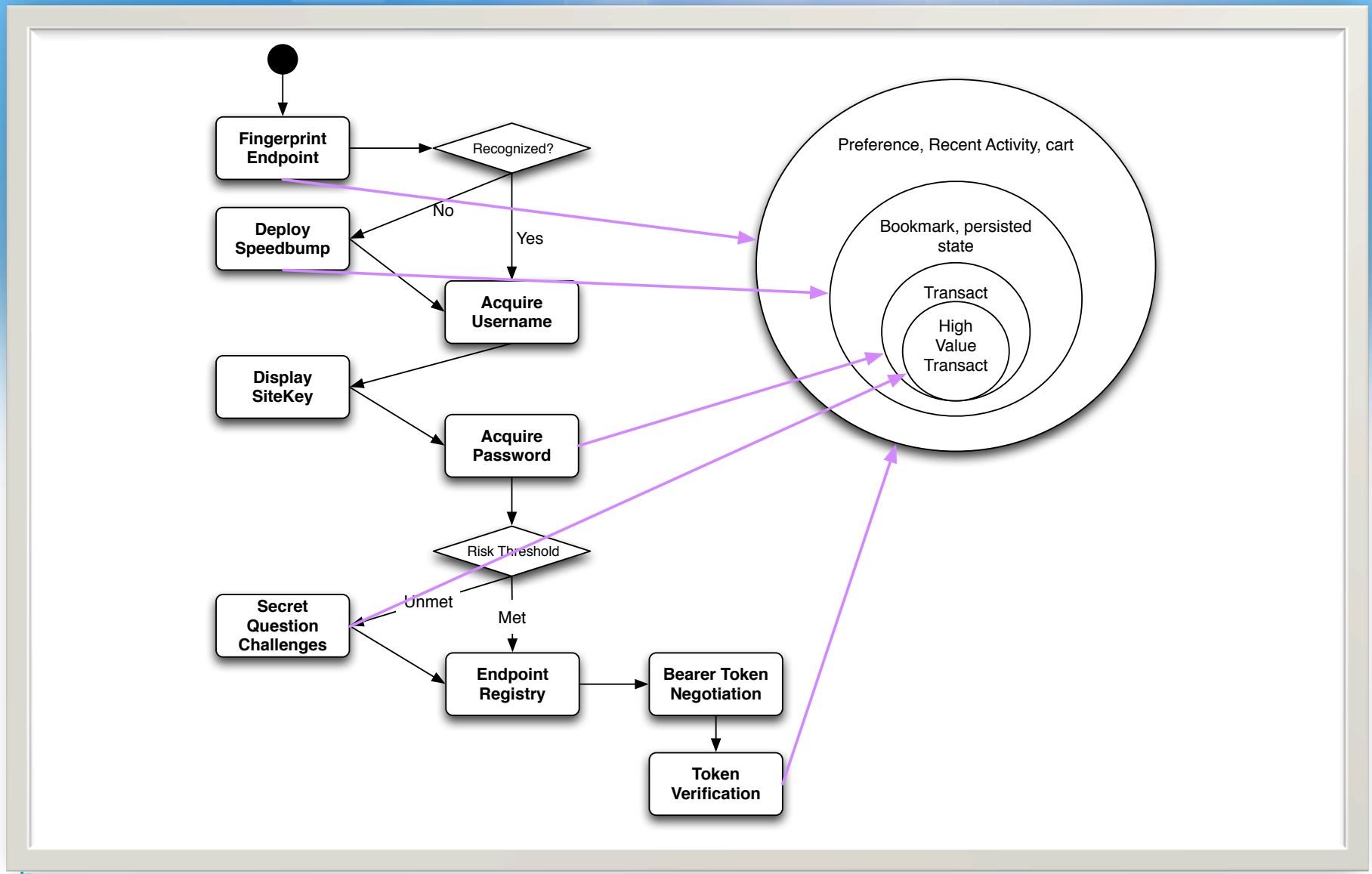


Consider a small bank's "castle"

Consider as alternative: Amazon.com

cigital

Castles, Entitlements, and so forth



Thank you for your attention