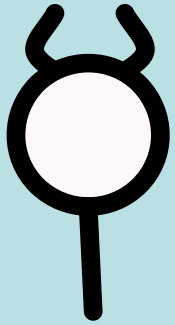
Simplifying Threat Modeling

Mike Ware
Cigital, Inc.
1.703.404.9293, x1251

OWASP
9/23/2011

Today's Threat Modeling Theme

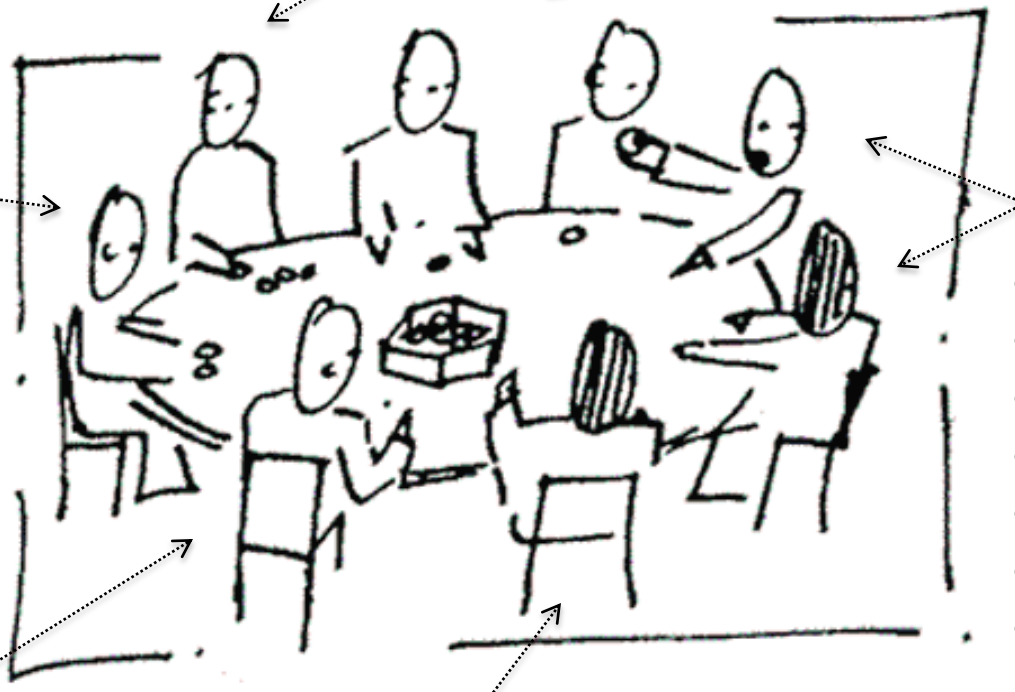


Keep it simple.

Objective:
Provide a framework to facilitate a threat modeling roundtable

Builders

- Developers
- Vendors



Gluers

- Enterprise Arch
- CTO
- Shared Services

Owners

- Program
- Product
- Project
- Business
- Requirements
- ISO
- IRM

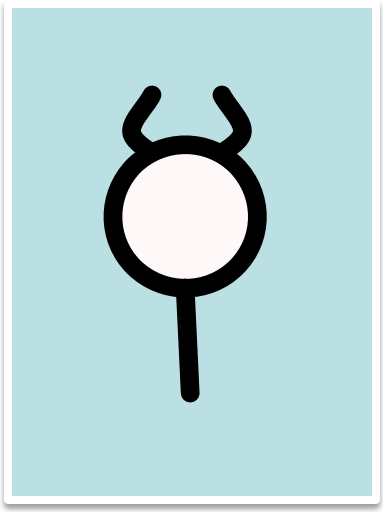
Defenders

- Infrastructure
- Ops

Breakers

- SSG
- External Pen Testers

What is a Threat?



- Anything (e.g., object, human) capable of performing unauthorized actions against a software system
- Possess **skills**, **access**, and **resources**

OWASP_NoVA_Chapter: https://groups.google.com/forum/#!forum/novaowasp_threatmodeling

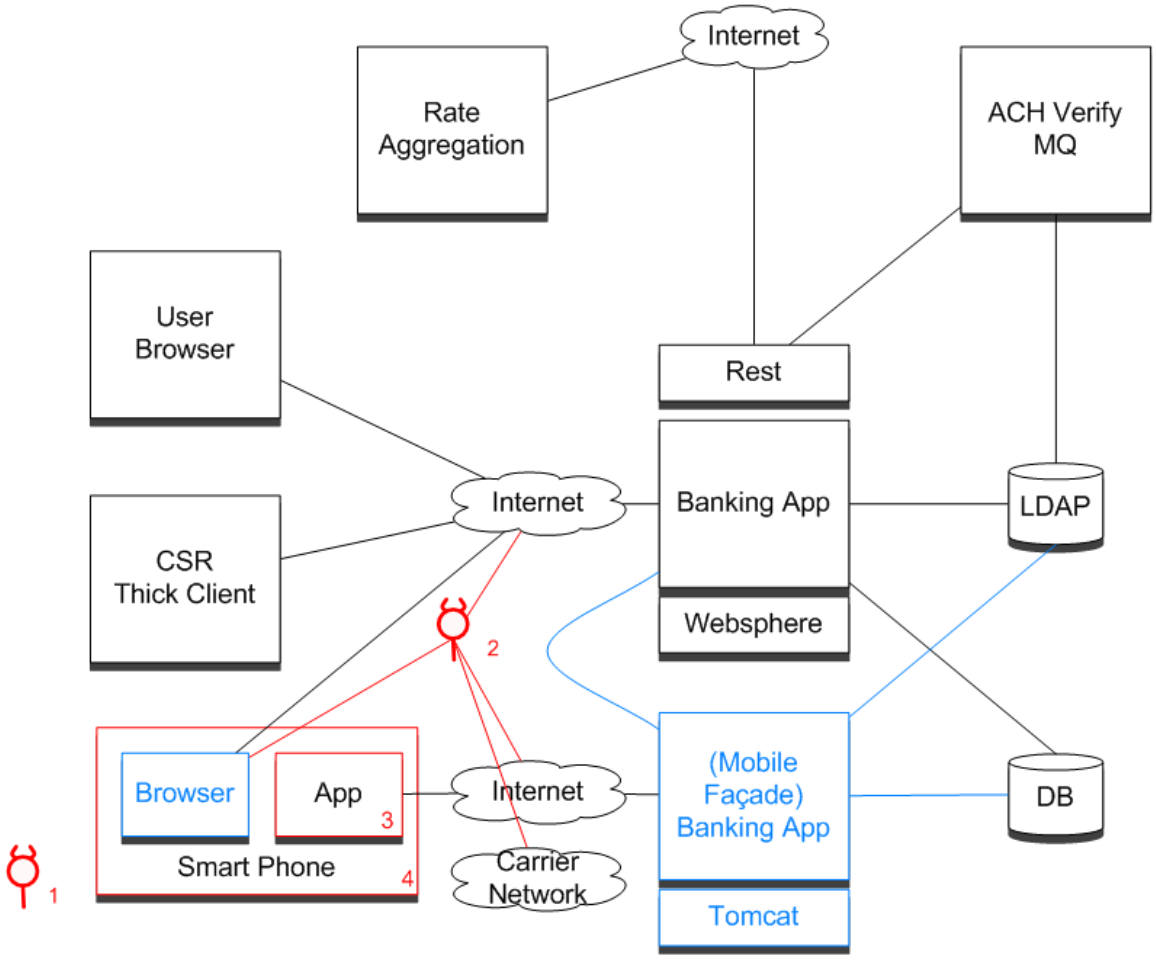
Threat Example – Mobile Architecture

Malicious Device User (1)

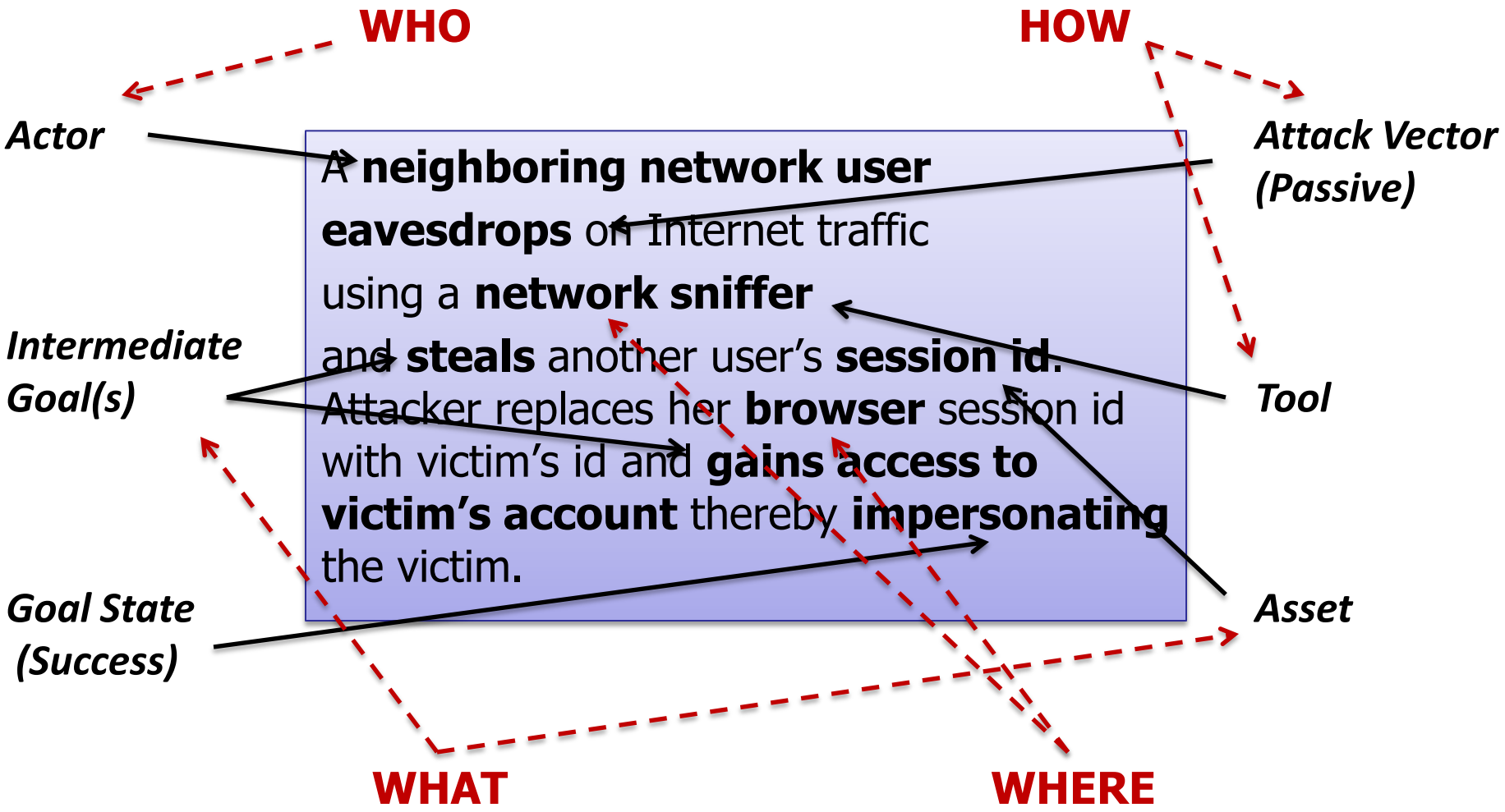
Skills
- Jailbreak device
- Reverse engineer software
- Install/modify software

Access
- Access to device
- Access to apps/browsers
- Access to device SDK

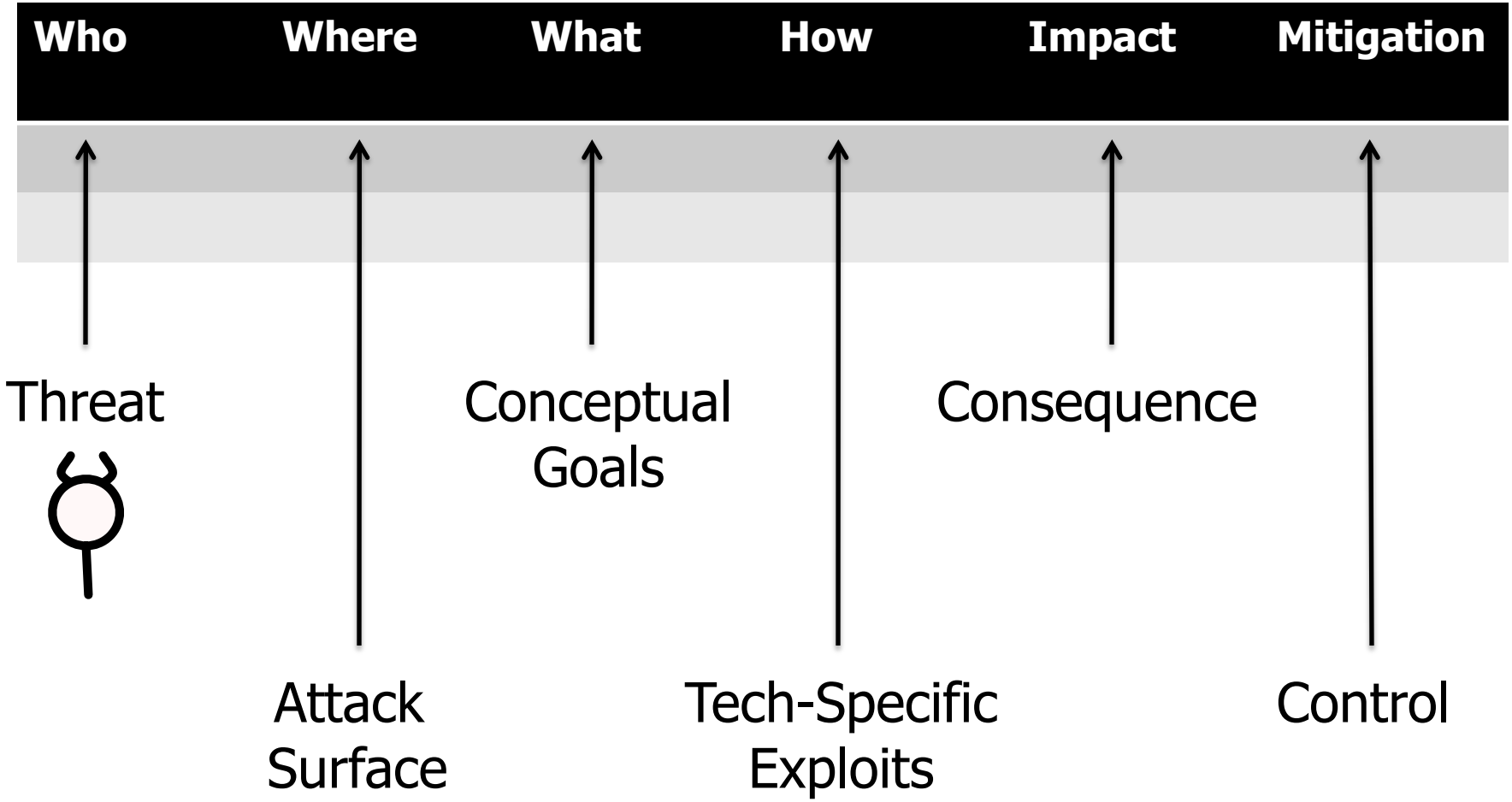
Resources
- Possess device/app credentials
- Disassemblers, proxies



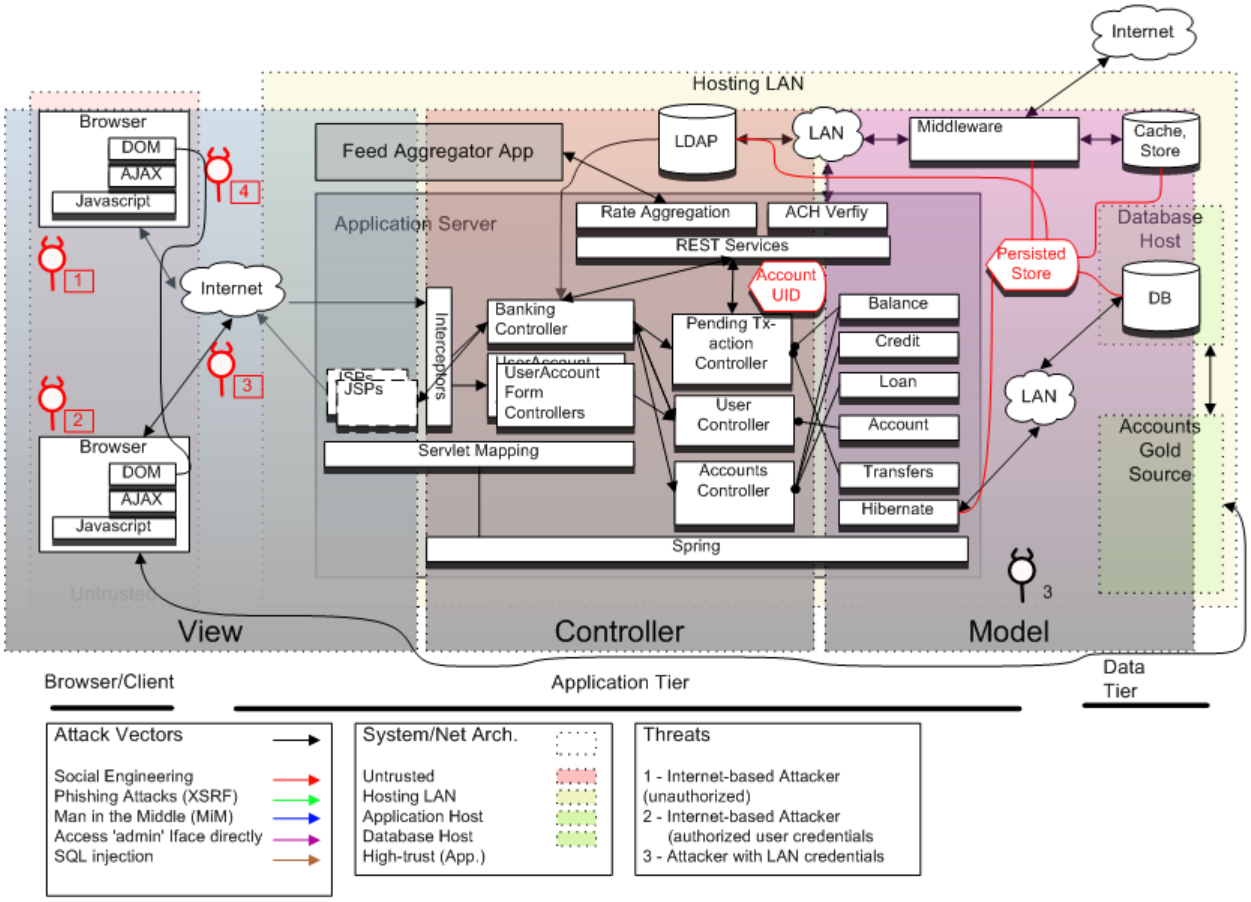
Anatomy of an Attack



Threat Traceability Matrix

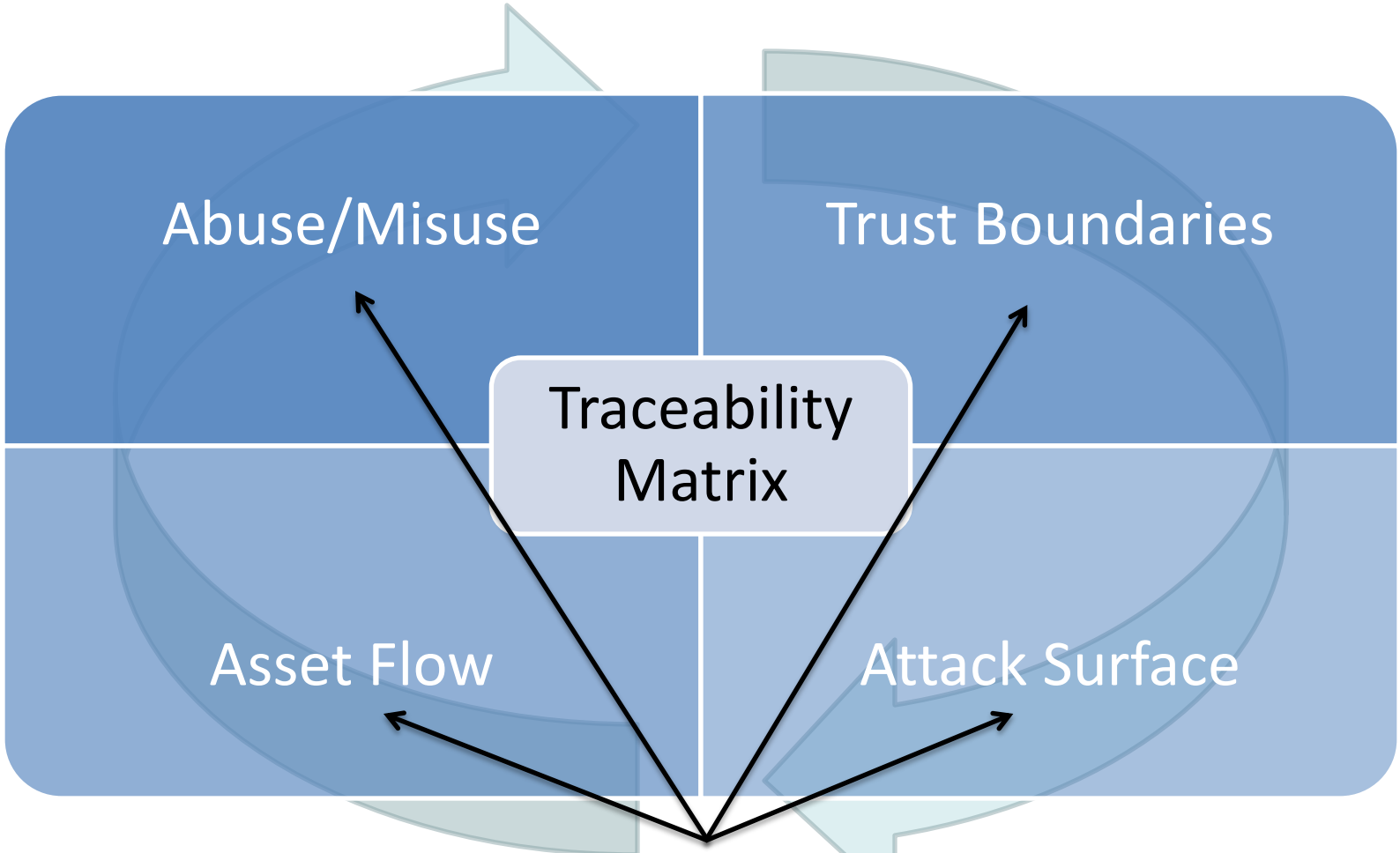


Elements of a Threat Model

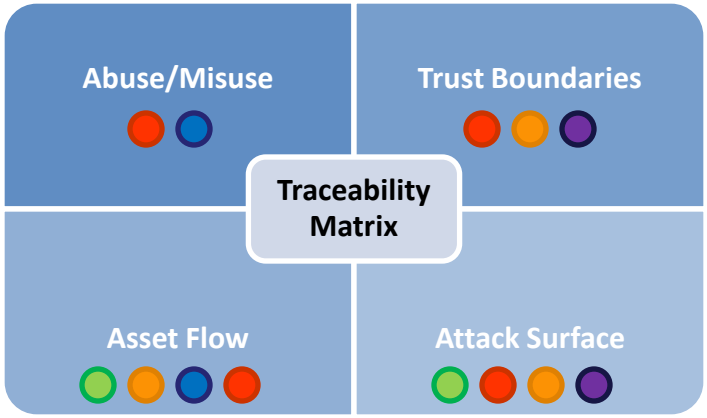


- **Software architecture** – structure, interaction, control flow, frameworks, services, design patterns
- **Threats**
- **Assets** (data and function)
- **Attack Vectors**
- **Security Controls**
- Notion of '**trust**'

Simplified Threat Modeling Framework



Views of a software system

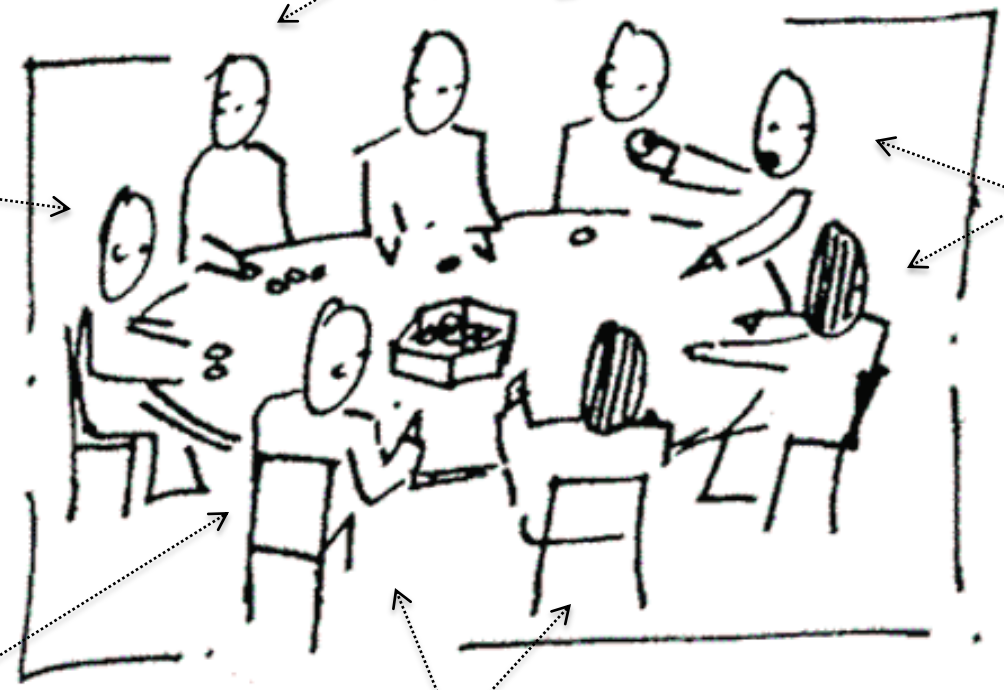


- Developers
- Vendors

Builders

Gluers

- Enterprise Arch
- CTO
- Shared Services



Owners

- Program
- Product
- Project
- Business
- Requirements
- ISO
- Risk

Defenders

- Infrastructure
- Ops

Breakers

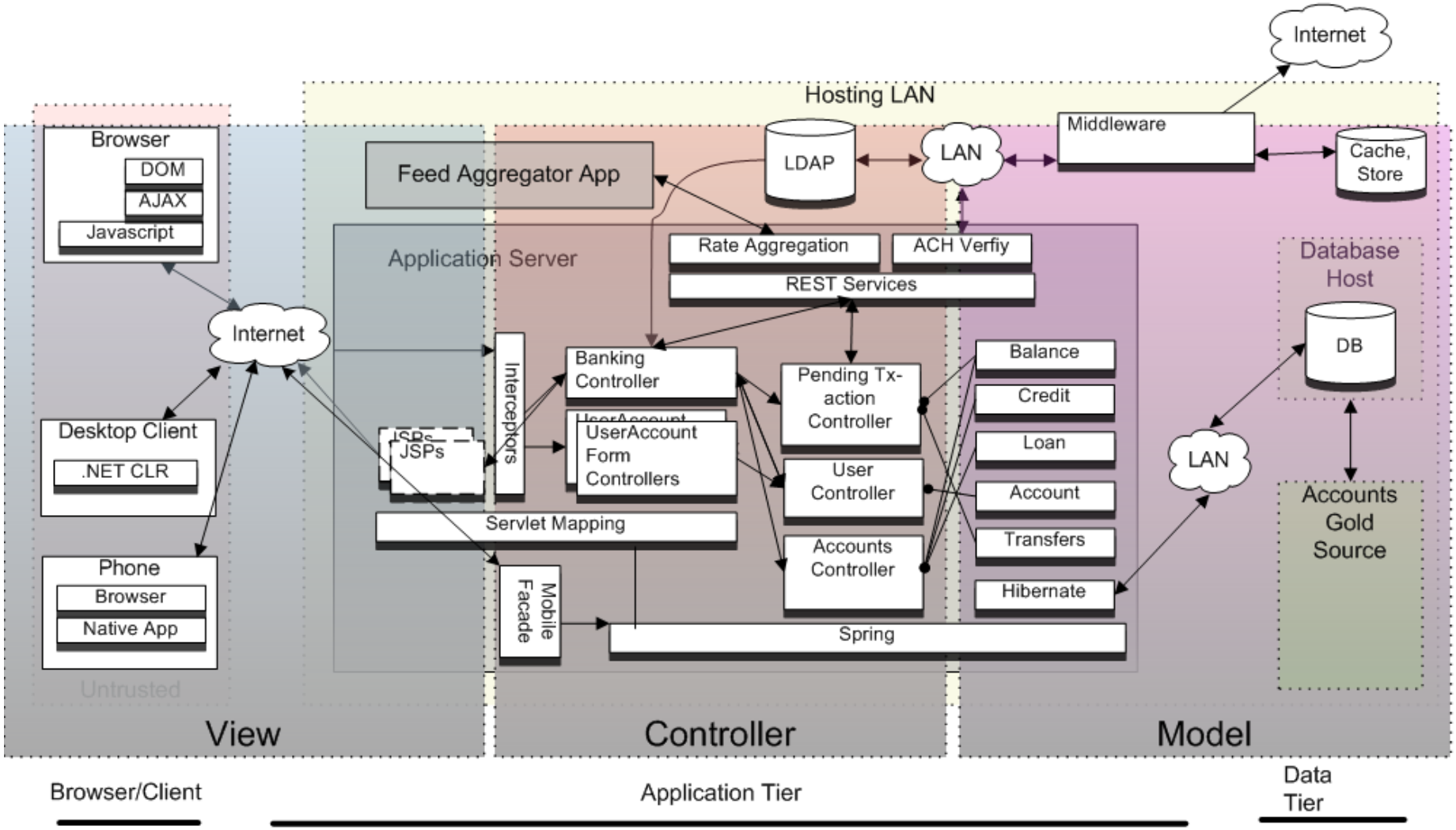
- SSG
- External Pen Testers

Keep it simple.

7+1 Threat Modeling Steps

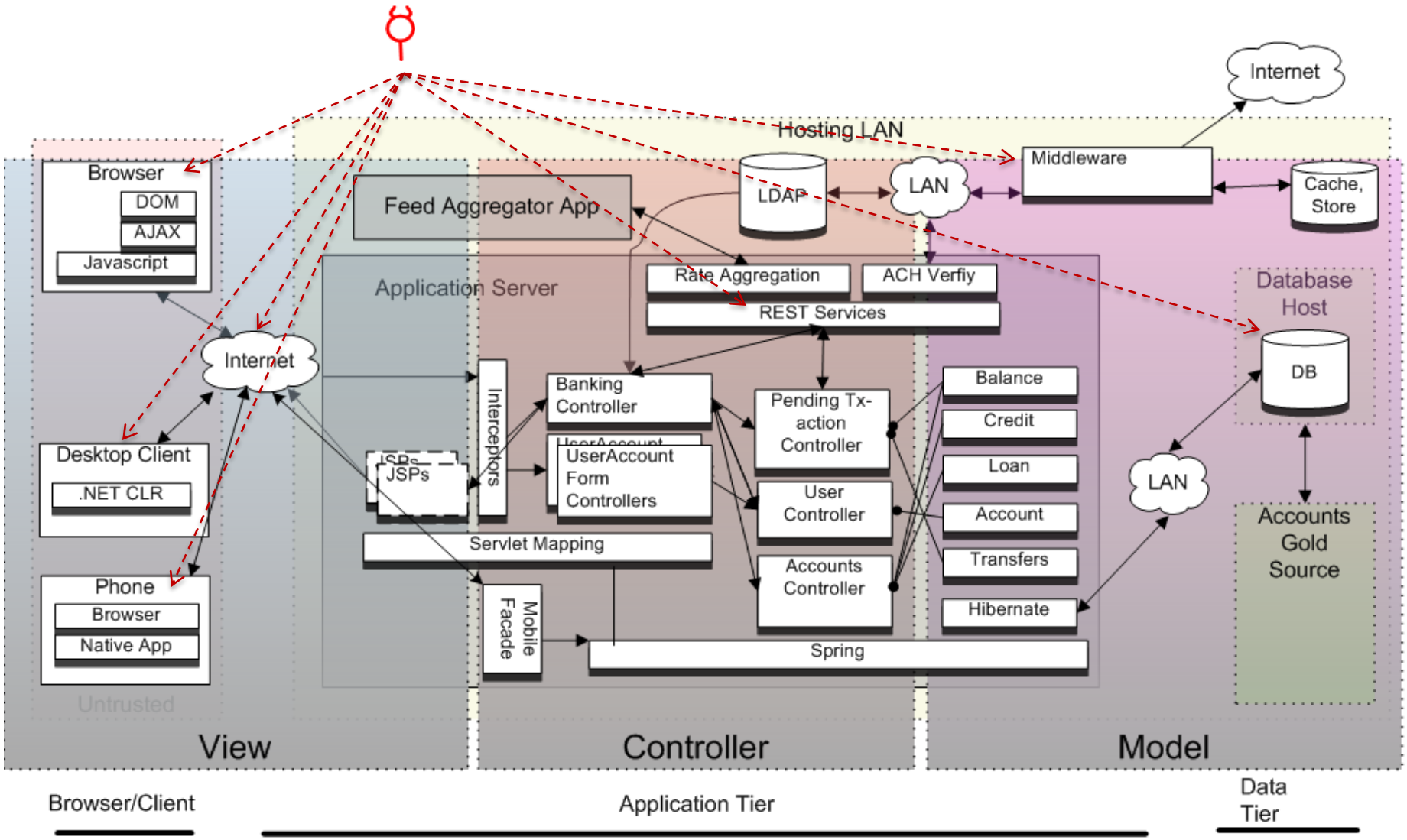
1. Diagram Software Architecture

Attack Surface



2. Enumerate Attack Surface(s)

Attack Surface



Attack Surface View

- Gluers
- Builders
- Breakers
- Defenders

Viewpoints

Characteristics

- Interfaces enabling interaction
  - Web, services, middleware, data tier, etc.
- Interaction model
  - Synch, async, transactional
  - Stateful, stateless
- Technology enabling interaction
- Authentication/authorization

SDLC

Design

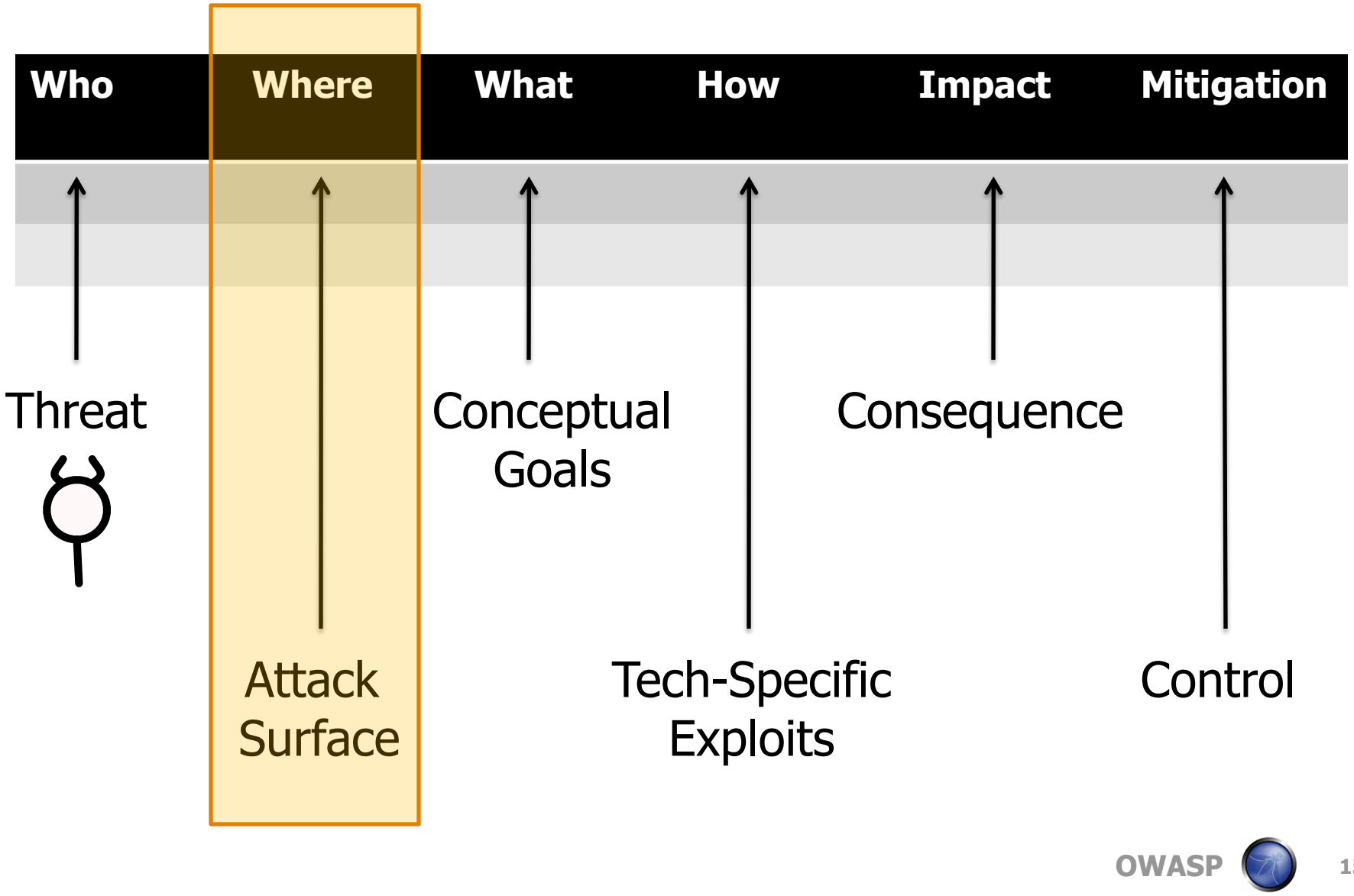
- High level architecture
- Low level design

Inputs/Usage

- Design/architecture changes
- Integration with:
  - Frameworks, toolkits, 3rd party libraries
  - Partners, service providers
  - Other enterprise systems
- Discovery, mapping, and other tool usage

- 'WHERE' traceability matrix column

Threat Traceability Matrix



3. Each User Class Becomes a Threat

Abuse/Misuse

User	Threat	Malicious Intent	Non-Malicious Behavior
Account Holder	Malicious Customer	Fraud, steal money, sabotage accounts	Inadvertent account lockout
Customer Support Representative (CSR)	Malicious CSR	Sell sensitive customer information	Backup customer data
Phone User	Malicious Device User	Install malware, reverse engineer app, jailbreak phone	Lose phone

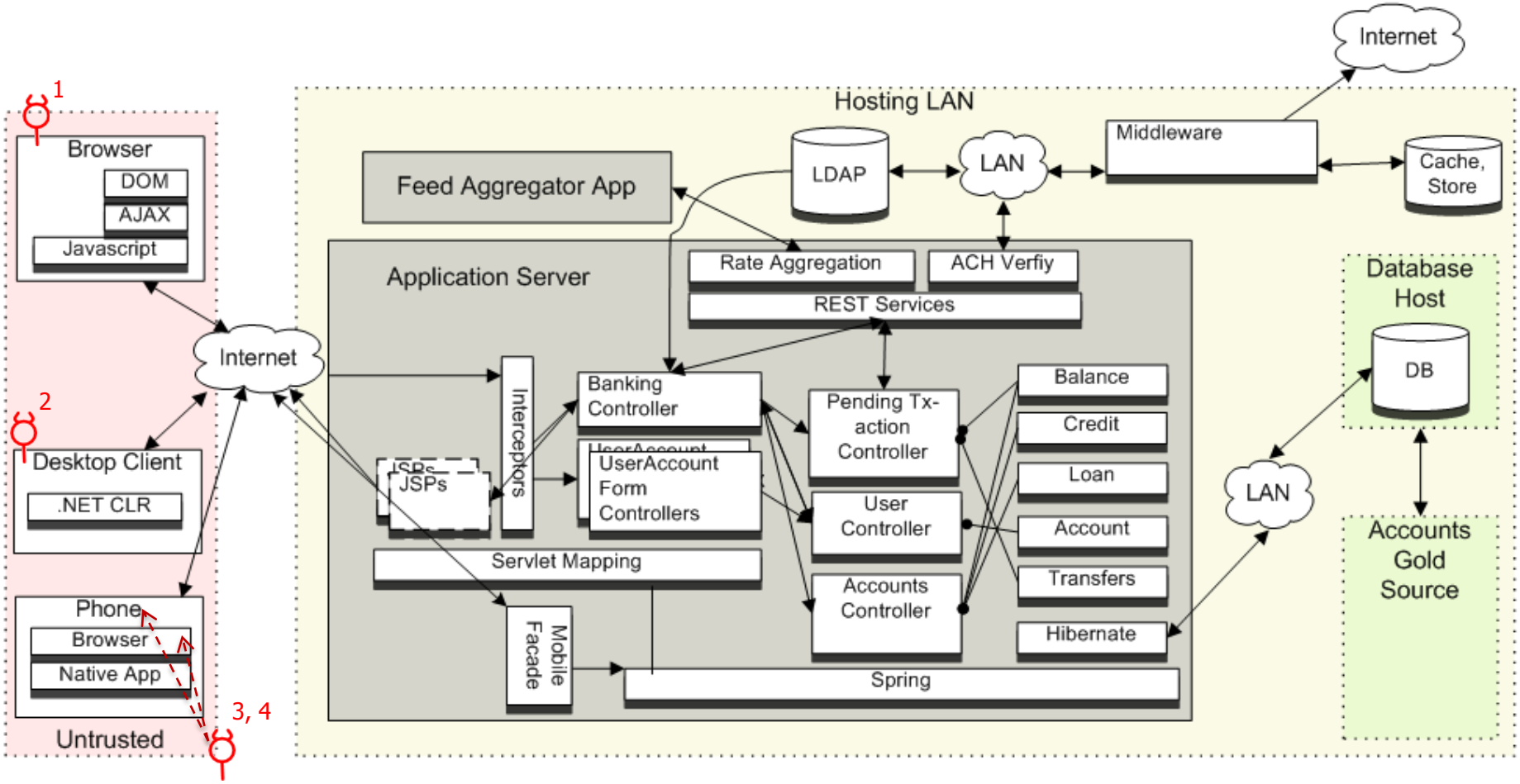
Malicious Intent Creates New Threat

Abuse/Misuse

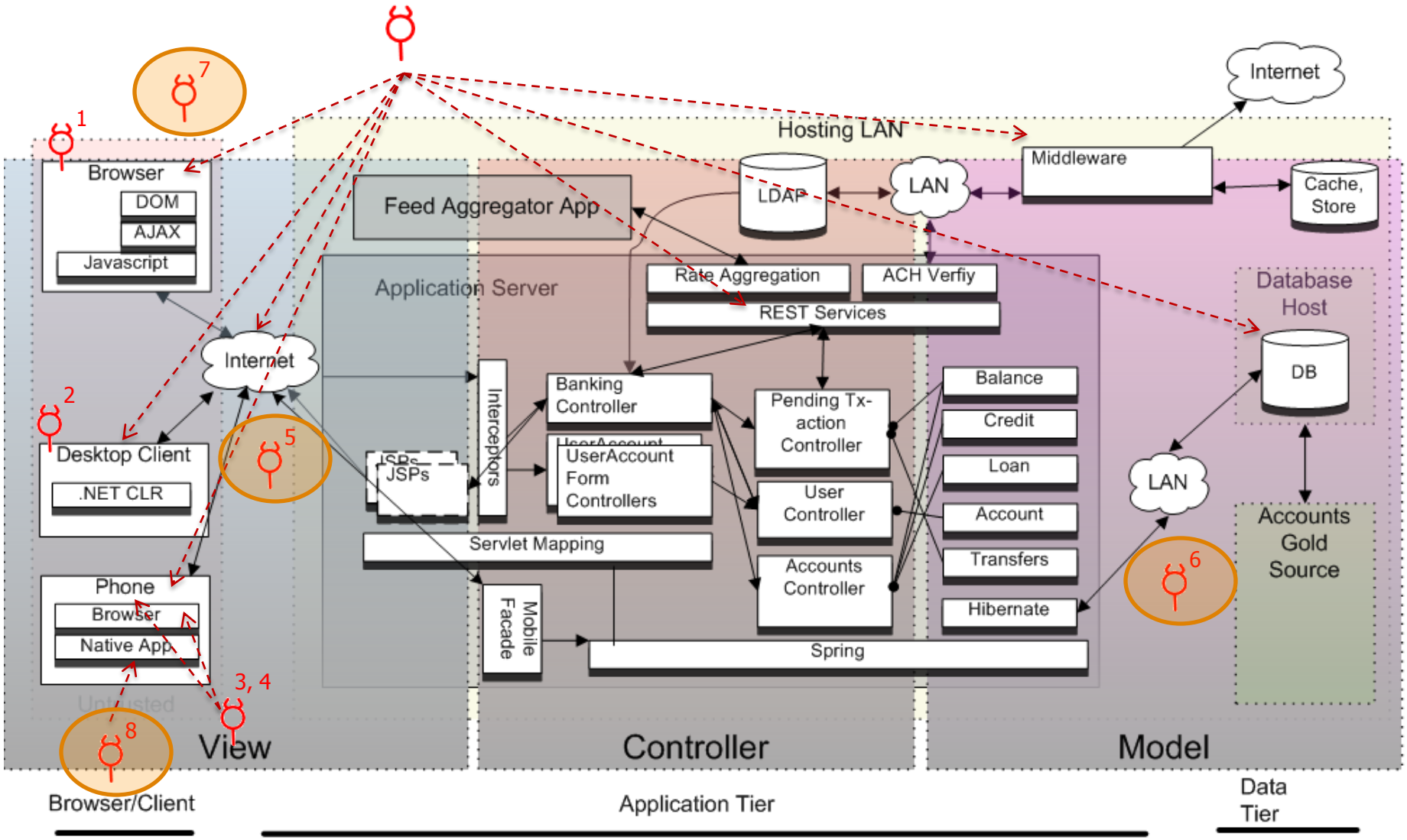
User	Threat	Malicious Intent	Non-Malicious Behavior
Account Holder	Malicious Customer	Fraud, steal money, sabotage accounts	Inadvertent account lockout
Customer Support Representative (CSR)	Malicious CSR	Sell sensitive customer information	Backup customer data
Phone User	Malicious Device User	Install malware, reverse engineer app, jailbreak phone	Lose phone
	Malicious Device		

Visualize Normal Users as Threats

Abuse/Misuse



Re-consider Attack Surface(s)



Abuse/Misuse Case View

- Owners
  - Business
  - Product
  - Requirements
- Breakers

Viewpoints

Characteristics

- Abuser/misuser (actor)
- System interface to actor (attack surface)
- Preconditions
- Inputs
- Actor's actions
- Expected outcomes

SDLC

- Use cases, user story elicitation
- High level requirements definition
- List of threat actor profiles
  - Skills
  - Access
  - Resources

Inputs/Usage

Requirements

- Functional
- Non-functional

- Link abuse/misuse to 'WHERE'
- 'WHO', 'WHAT', 'HOW'

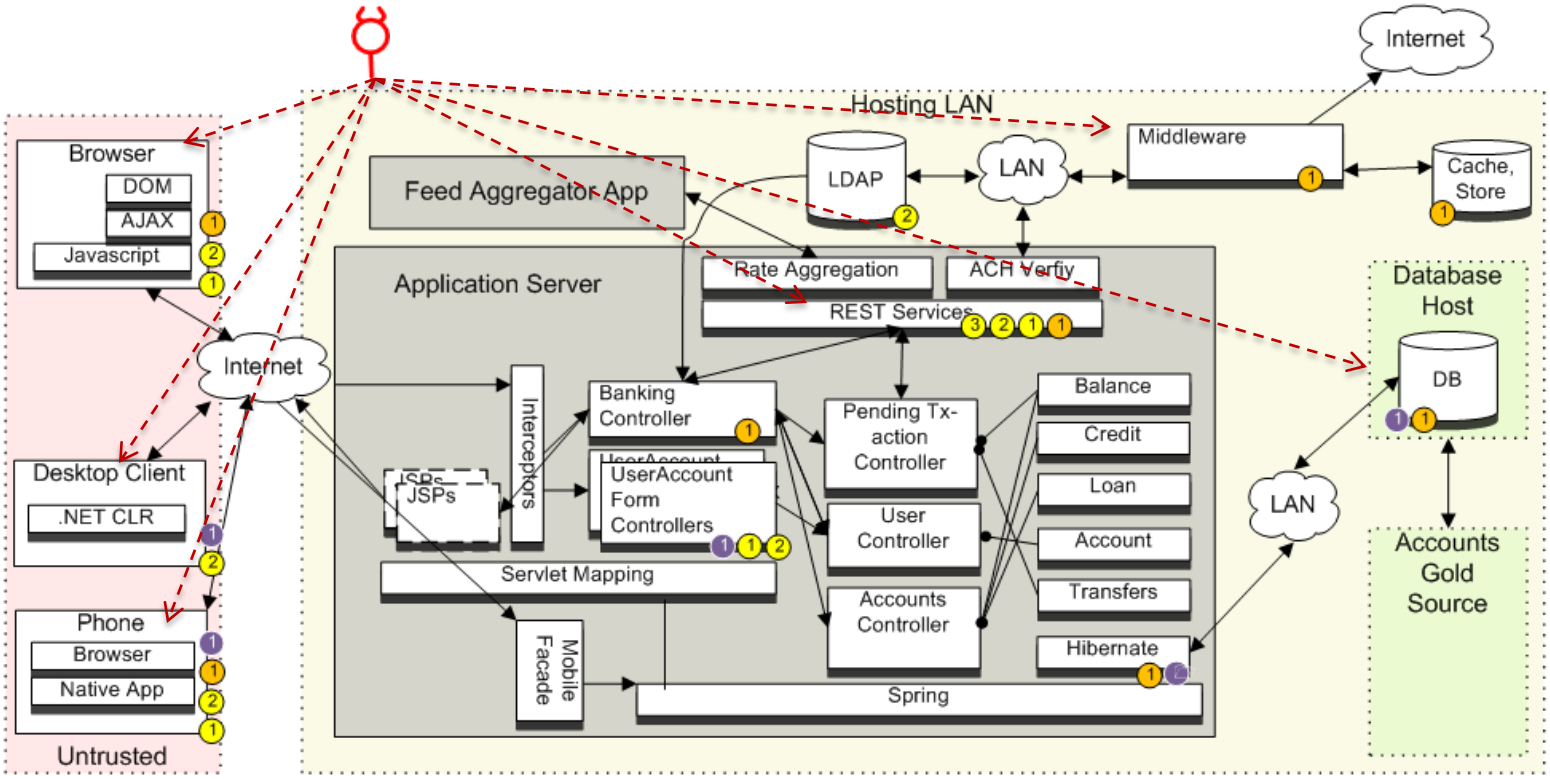
Capture 'Who', 'Where', and 'What'

Traceability Matrix

Who	Where	What	How	Impact	Mitigation
1. Malicious Account Holder	User's Browser	• Execute fraudulent transactions			
2. Malicious CSR	Desktop Client	• Steal customer PII			
4. Malicious Mobile Device	Phone OS, SDK	• Capture and transfer application data			
7. Malicious Third Party	User's Browser	• Steal user credentials			

4. Illuminate Assets

Asset Flow



1. Session Identifier
2. Credentials
3. Principal
1. PII
1. Account Info: balance, IDs, withdrawal, deposit, transfer

Asset Flow View

- Owners
  - Risk (IRM)
- Gluers
- Builders
- Breakers

Viewpoints

Characteristics

- Data and functionality
- Threat agent(s) level of access
- Exposure to attack surface(s)
- Asset classification
- Protection mechanisms
  - Rest, process, transit
  - Egress, ingress
- Qualifying technologies

SDLC

Requirements **Design**

- Information architecture
- High level architecture diagram

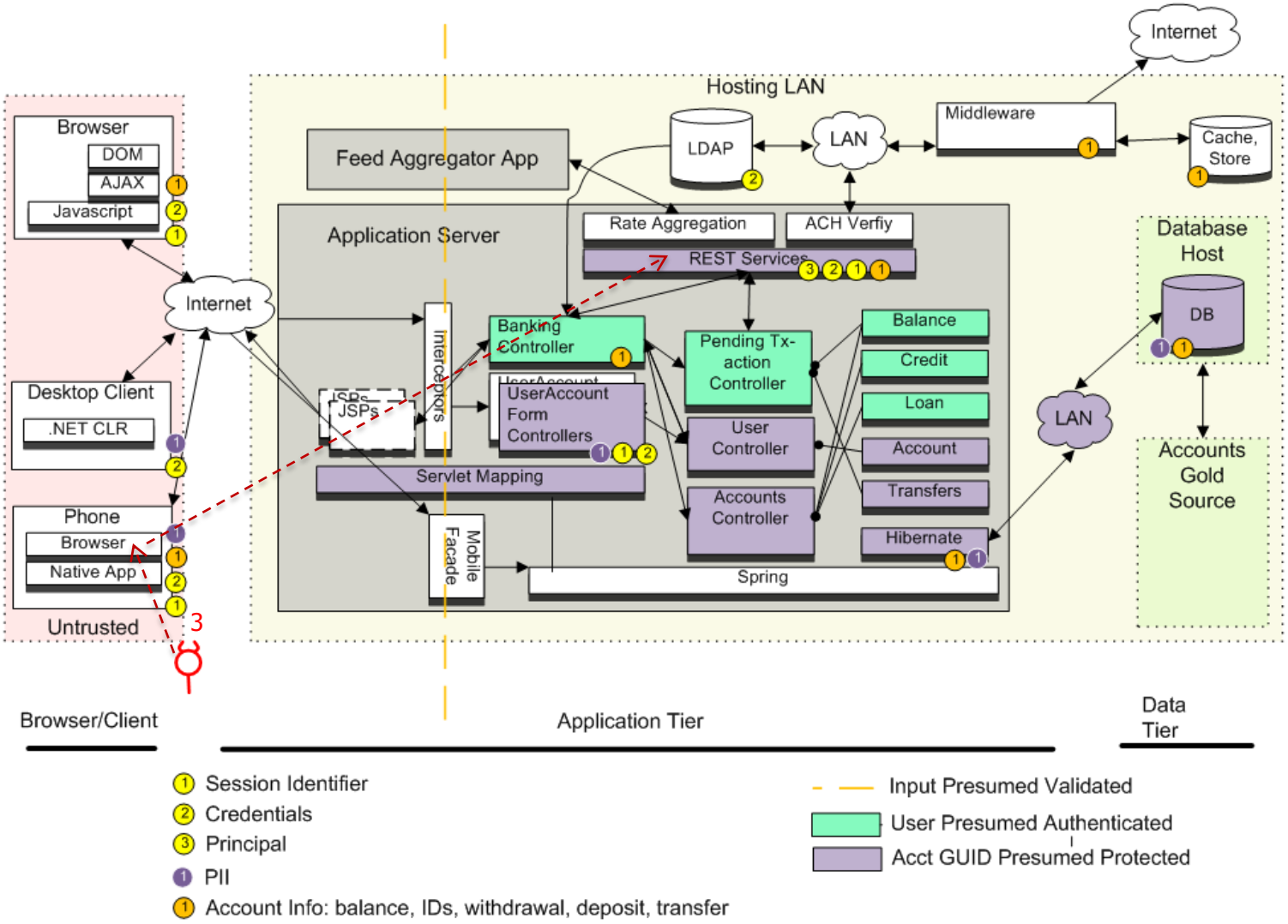
Inputs/Usage

- Data View + CRUD
- Schemas, config, DTDs
- SCR, VA assessment results

- Enhance 'WHAT', 'HOW' with contextual information
- Evaluate 'IMPACT' of abuse/misuse

5. Illuminate Trust Boundaries

Trust Boundaries



6. Postulate Attacks Against Assets

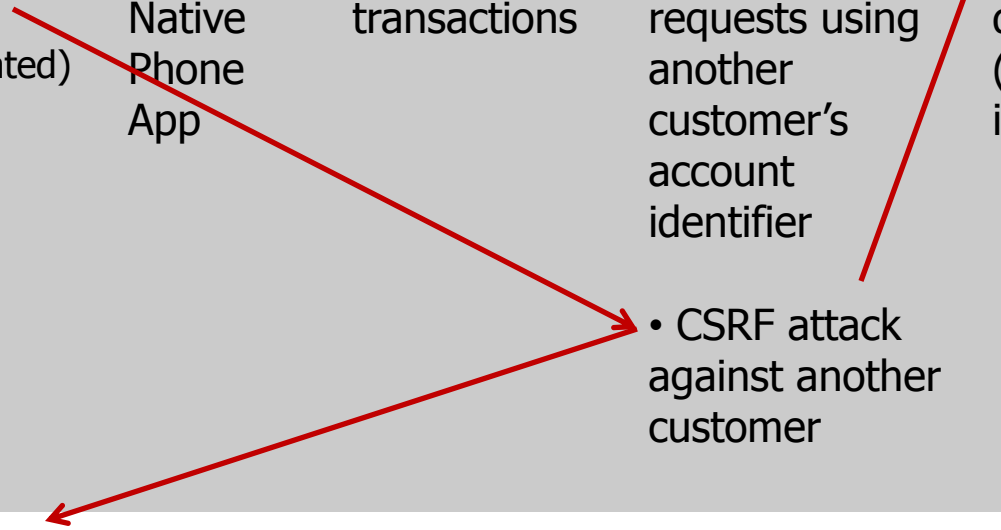
Traceability Matrix

Who	Where	What	How	Impact	Mitigation
3. Malicious Mobile Device User (unauthenticated)	User's Browser, Native Phone App	Execute fraudulent transactions	• Directly make REST API requests using another customer's account identifier • CSRF attack against another customer		

7. Evaluate Impact

Traceability Matrix

Who	Where	What	How	Impact	Mitigation
3. Malicious Mobile Device User (unauthenticated)	User's Browser, Native Phone App	Execute fraudulent transactions	• Directly make REST API requests using another customer's account identifier • CSRF attack against another customer	• Fines • Brand damage (PR incident)	
4. Authenticated Malicious User	User's Browser, Native Phone App	Modify user account information		• Account recovery costs • Lose customer(s)	



8. Mitigate

Traceability Matrix

Who	Where	What	How	Impact	Mitigation
3. Malicious Mobile Device User (unauthenticated)	User's Browser, Native Phone App	Execute fraudulent transactions	• Directly make REST API requests using another customer's account identifier	• Fines • Brand damage • Account recovery costs	R.1.a: Authenticate REST API requests (user level) R.1.b: Authorize all REST API calls (message level)
			• CSRF attack against another customer		S.1.a: Implement request tokens for all state changing servlets

Trust Boundaries View

- Boundaries defined by set of security properties
  - AuthN/AuthZ
  - I/O Controls
  - Privileged functionality/data
  - Connections & protocols
  - Object marshaling and remoting
  - Queues, channels
  - …

Characteristics

- Gluers
- Breakers
- Defenders

Viewpoints

SDLC

Design

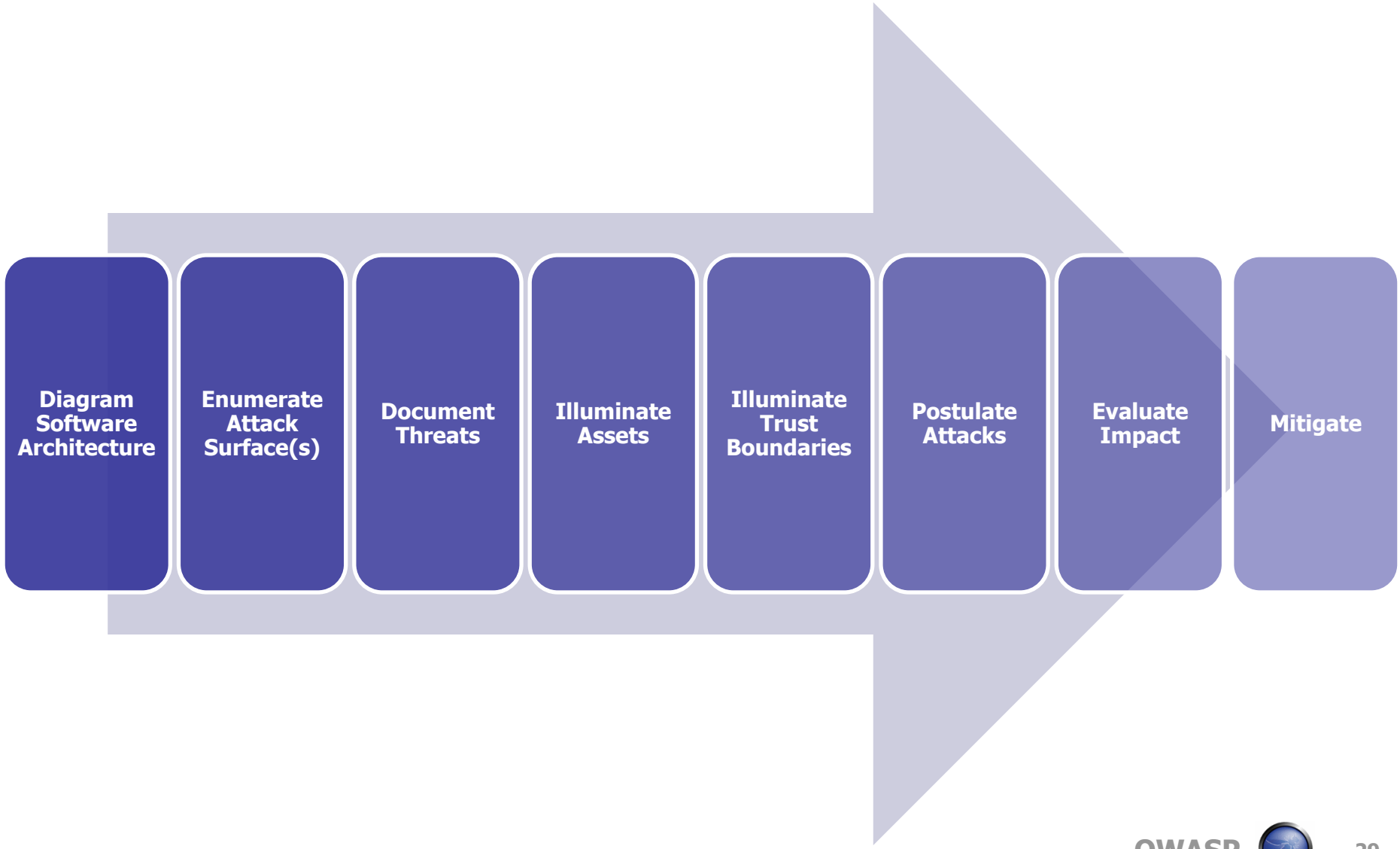
- High level architecture
- Low level design

Inputs/Usage

- 'Attack Surface View'
- 'Asset Flow View'

- Postulate 'HOWs' by speculating about weaknesses in trust boundary implementations

7+1 Threat Modeling Steps



Acting on Threat Modeling Results

Contact

- Mike Ware
- Sr. Security Consultant, Cigital
- mware at cigital dot com

Software Confidence. Achieved.