reat Modeling Password Stora

Enumerating & Understanding Threats

-jOHN
Internal CTO,
@m1spla

Problem Definition

History /etc/password

```
etc/password

:0:0:EC90xWpTKCo
kman:100:100:KMEzyulaQQ2
dthwa:101:101:Po2gweIEPZ2
ven:102:500:EC90xWpTKCo
l:103:500:NTB4S.iQhwk
aj:104:500:a2N/98VTt2c
```

- Circa 1973
- ‘one-way’ password encryp
- chmod a+r /etc/passwd
- DES took 1 sec per passw

…bringing us to 2012

```
0fac2ec84586f9f5221a05c0e9acc3d2e670
022c7caab3ac515777b611af73afc3d2ee50
6f052152cfed79e3b96f51e52b82c3d2ee8e
0dc7cc04ea056cc8162a4cbd65aec3d2f0eb
0a2c4f4b579fc778e4910518a48ec3d2f111
4eaec4585720ca23b338e58449e4c3d2f628
b9e37ace89b77401fa2bfe456144c3d2f708
b1edf4f84a85d79d04d75fd8f8a1c3d2fbde
0e56fae33ab04c81e727bf24bedbc3d2fc5a
058918701830b2cca174758f7af4c3d30432
02e09ee4e5a8fcdae7e3082c9d8ec3d304a5
cbe8d2a38a1575d3feed73d3f033c3d304d8
0273b52ee943ab763d2bb3d83f5dc3d30904
```

What do you see here?

How do we know what it is?

How could we figure this ou

In the news

LinkedIn

IEEE

Yahoo

…

n Rules

Don’t be on the front
ge of InfoWeek

Have a great story when
u’re on the front page of
oWeek

Your passwor
WILL be
extracted fro
your system

What is a
Threat Mod

What is a Threat?

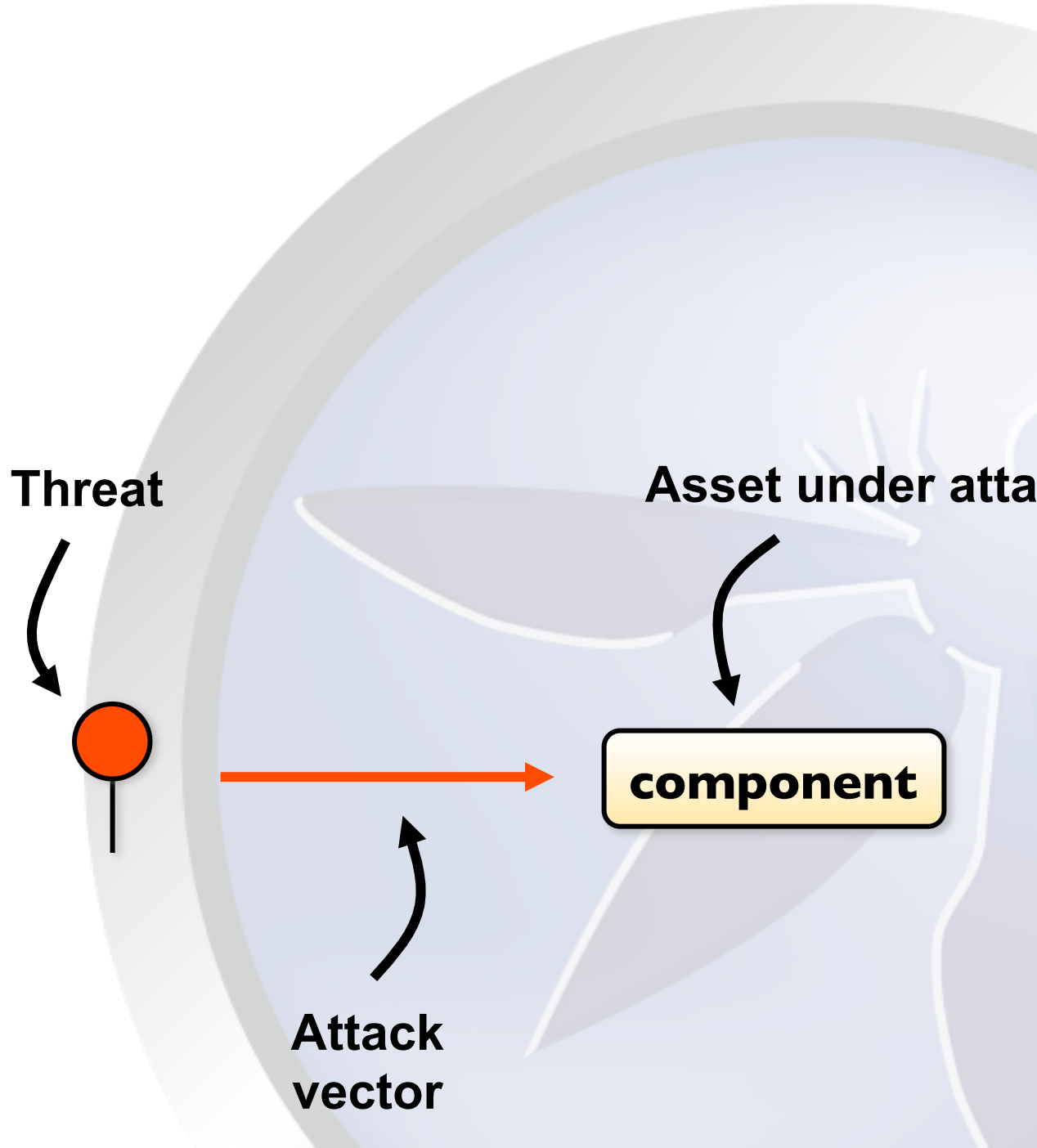
n agent who attacks you?

n attack?

n attack's consequence?

risk?

n agent



Confusion Over "Threat"

Literature equates "threat" to "event with unwelcome consequence"

Devolves modeling to a checklist of events

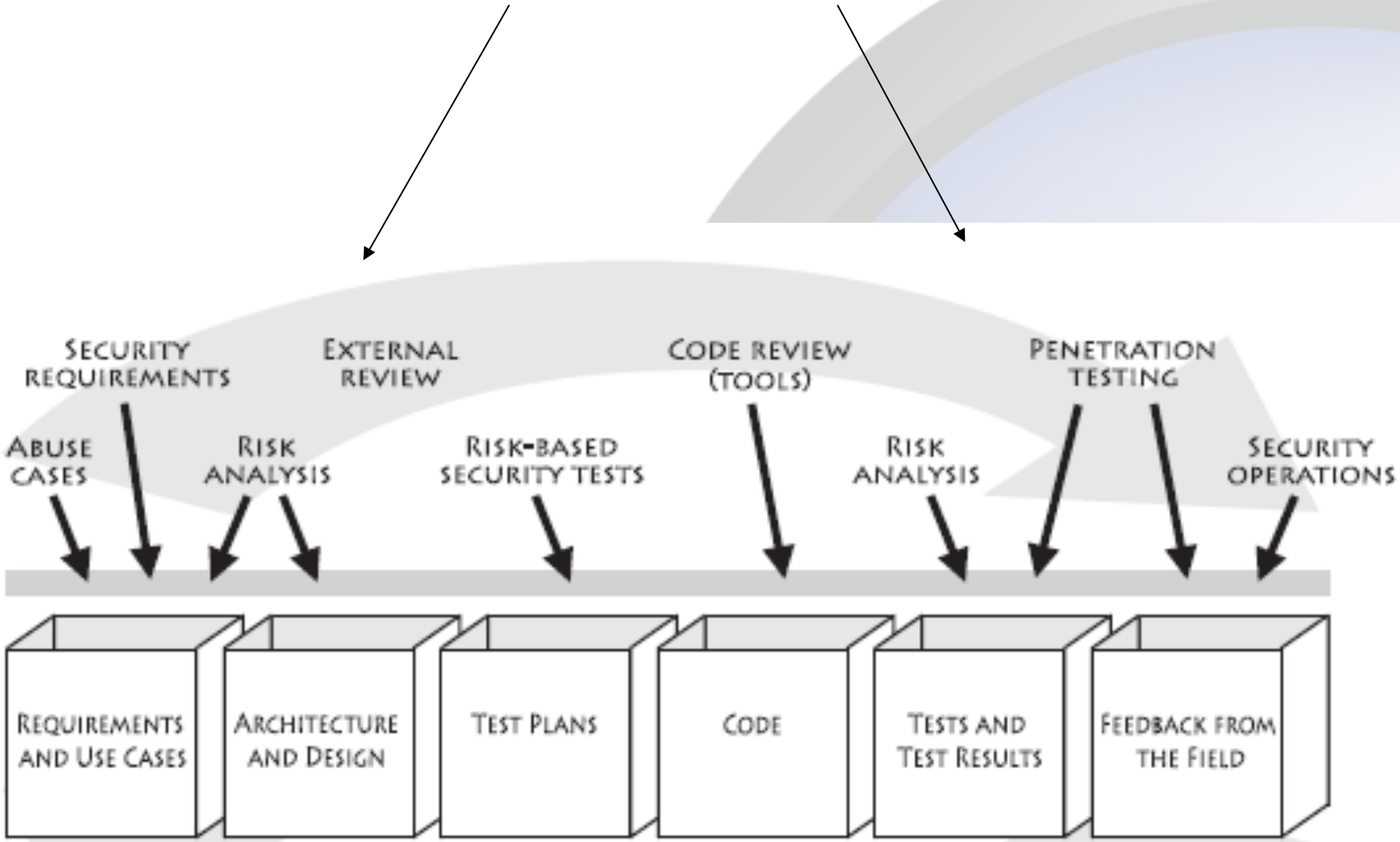
Should expand thinking about possible abuse

- Threats help
  - Encourage thorough thought about how intentions for misuse
  - Determine "out of bounds" scenarios

We refer to "threat" as a person or agent

You Are Here

Architectural Risk Analysis



What is a reat Model?

ction of:

e system's *attack surface*

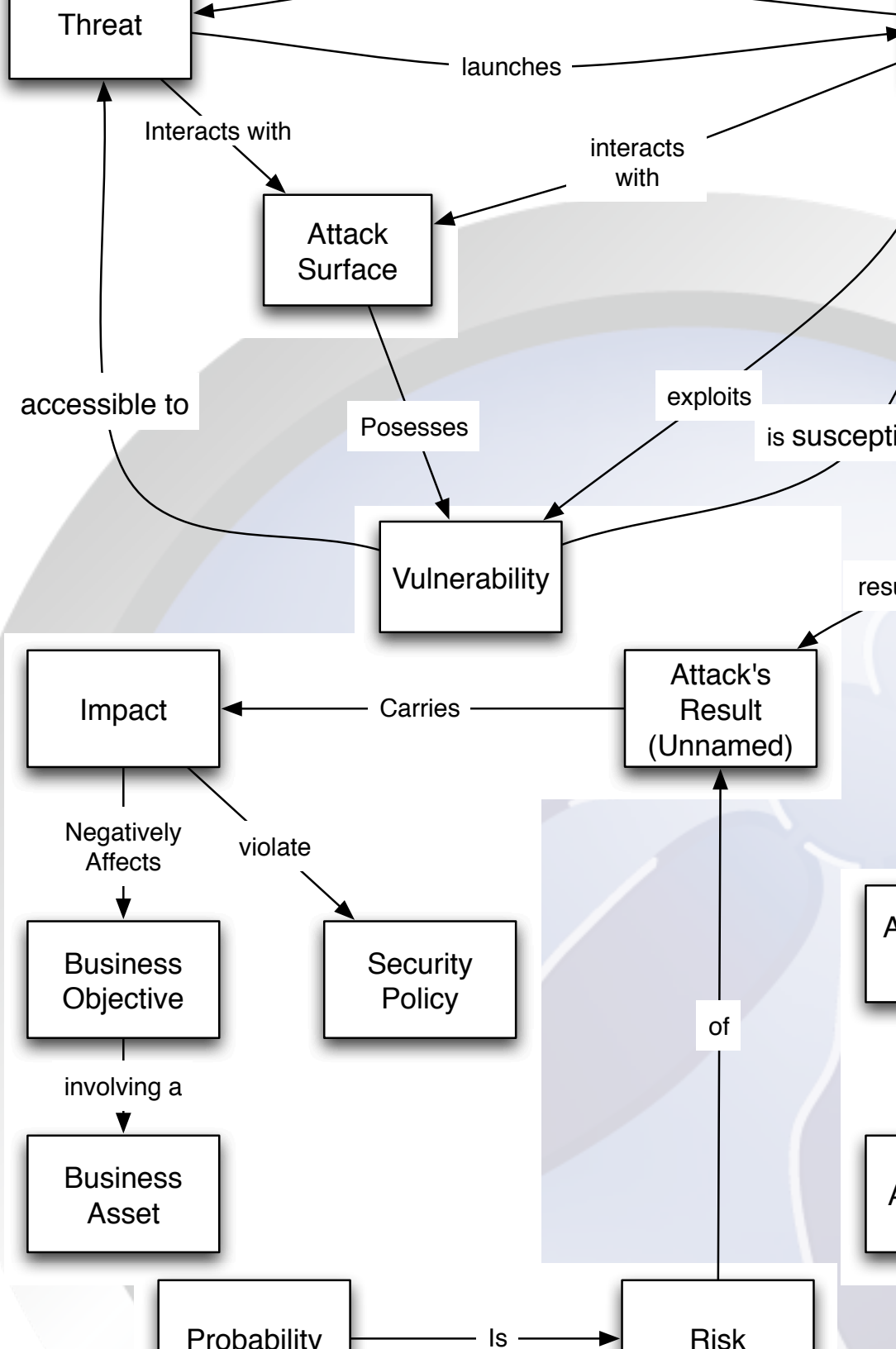
reats who can attack the system

sets threats may compromise

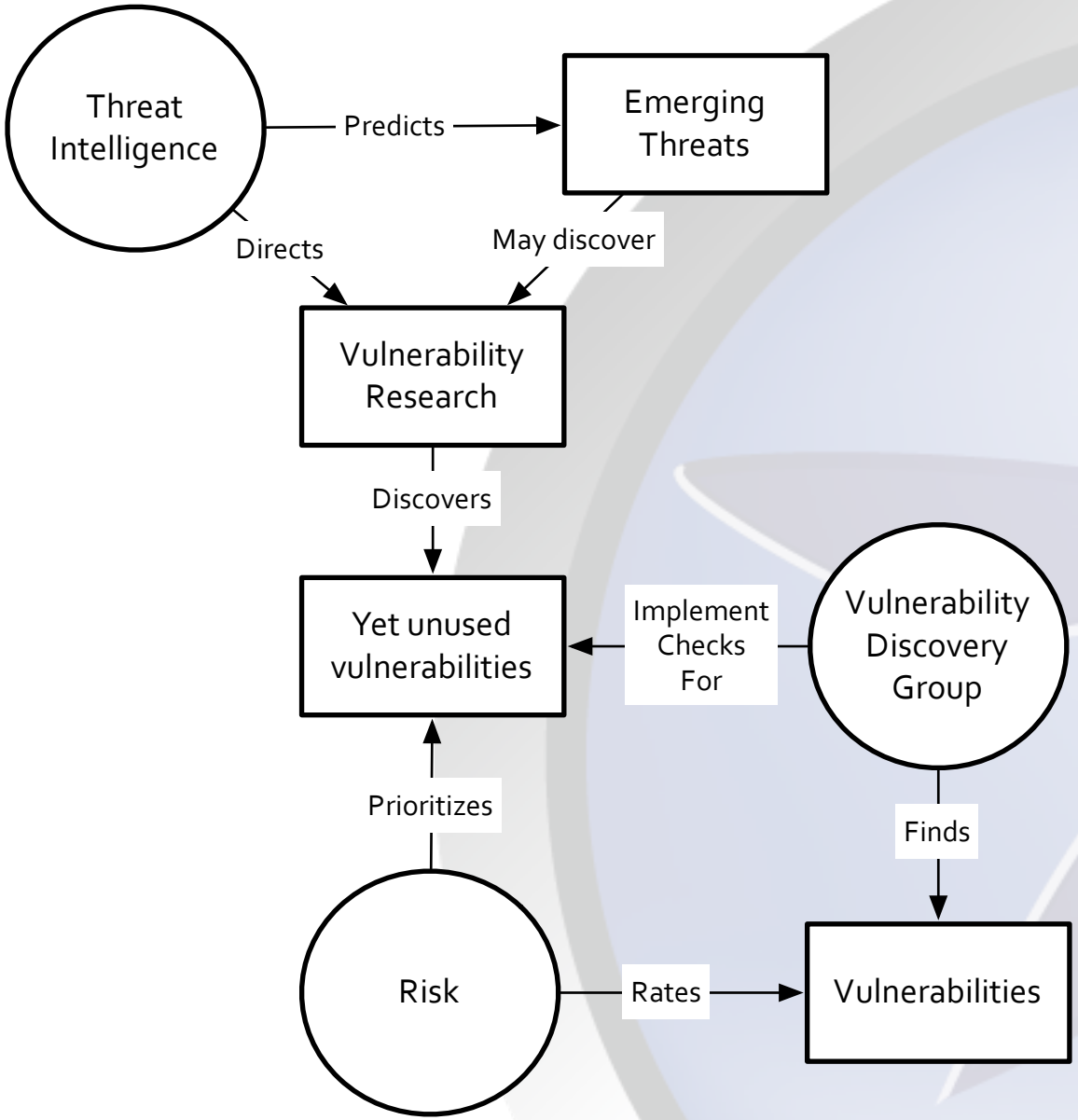
e leverage risk management tices

imate *probability* of attack

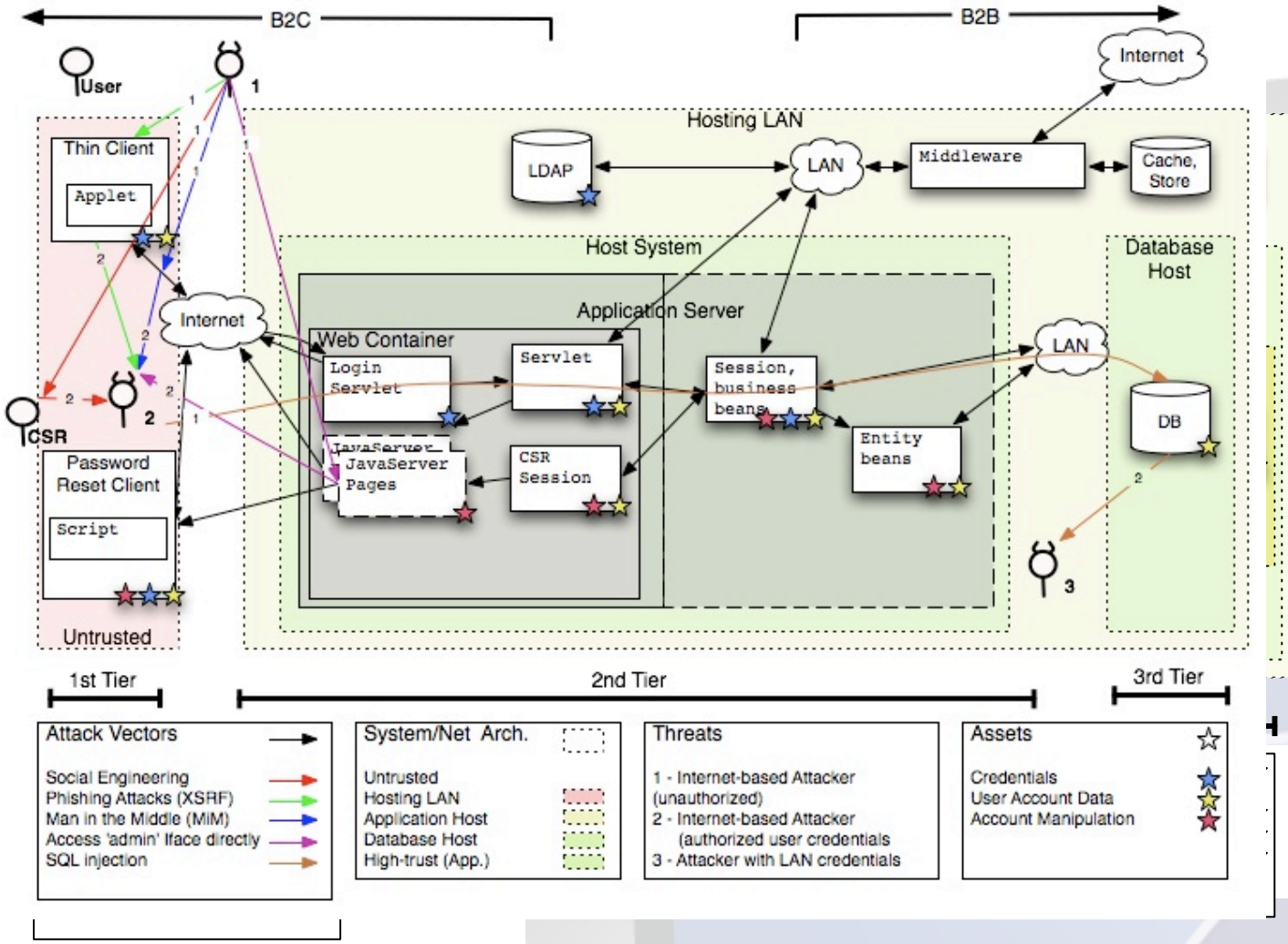
ight *impact* of successful



Who are the participants



hreat Model's Diagrammat



- Structural view
- Behavioral Views
- Assets
- Attack Vectors

Threat Traceability Matrix

	What	How	Impact	Mitigation

Threat Modeling as Process

eat Modeling – High-level proc

Diagram structure

Identify assets

Identify Threats

Enumerate doomsday scenarios

Document misuse/abuse

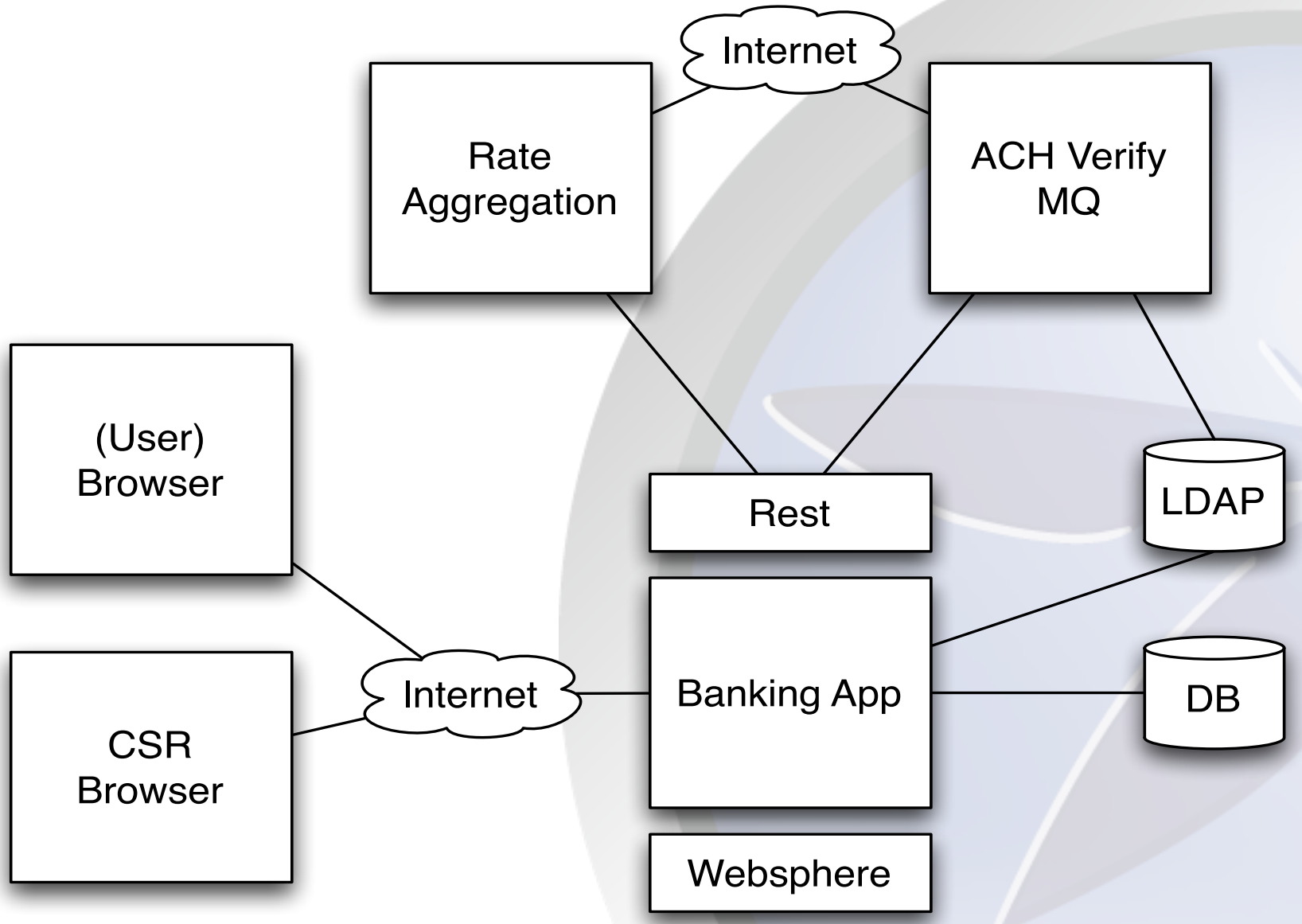
Architectural Risk Analysis

Iterate

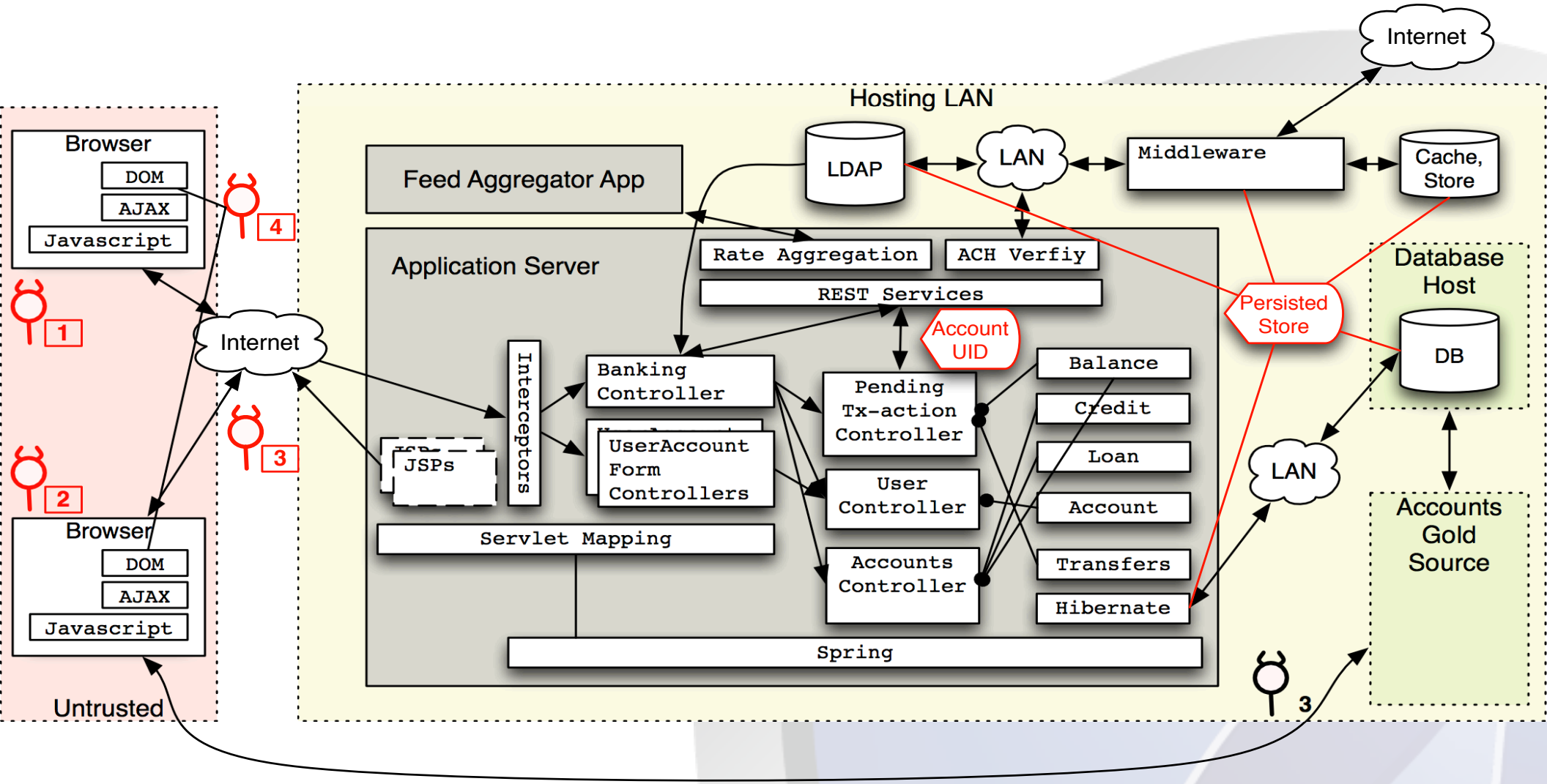
1. Identify threats
2. Set particular goals
3. Partition by capability
4. Enumerate attack vectors
5. Explore state of practice c attacks

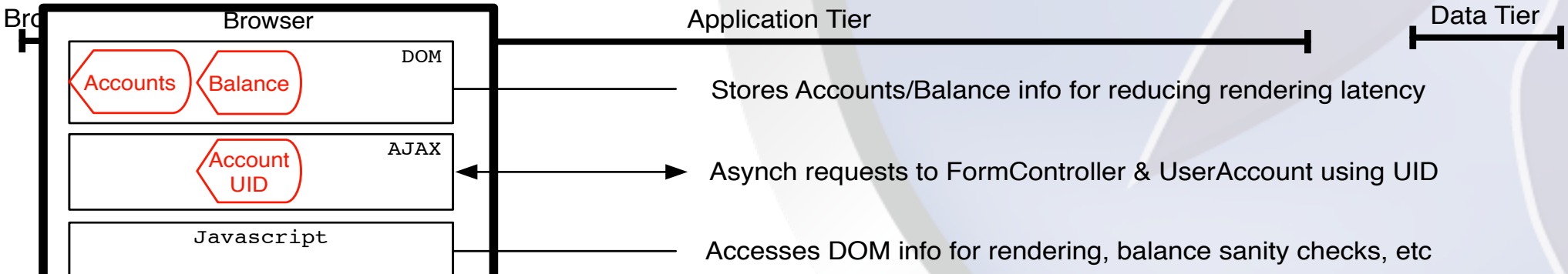
Software
Structure
Identify Atta
Surfaces

Given


Internet
Rate Aggregation
ACH Verify MQ
(User) Browser
Rest
LDAP
CSR Browser
Internet
Banking App
DB
Websphere

More Useful


Internet
Hosting LAN
Browser
DOM
AJAX
Javascript
Feed Aggregator App
LDAP
LAN
Middleware
Cache, Store
Application Server
Rate Aggregation
ACH Verfiy
REST Services
Persisted Store
Database Host
DB
Account UID
Internet
Interceptors
Banking Controller
Pending Tx-action Controller
Balance
Credit
Loan
Account
Transfers
Hibernate
JSPs
UserAccount Form Controllers
User Controller
Accounts Controller
LAN
Accounts Gold Source
Servlet Mapping
Spring
Browser
DOM
AJAX
Javascript
Untrusted
1
2
3
4
3


Browser
Application Tier
Data Tier
DOM
Accounts
Balance
Stores Accounts/Balance info for reducing rendering latency
AJAX
Account UID
Asynch requests to FormController & UserAccount using UID
Javascript
Accesses DOM info for rendering, balance sanity checks, etc

1 – Identify Application Attack Surface

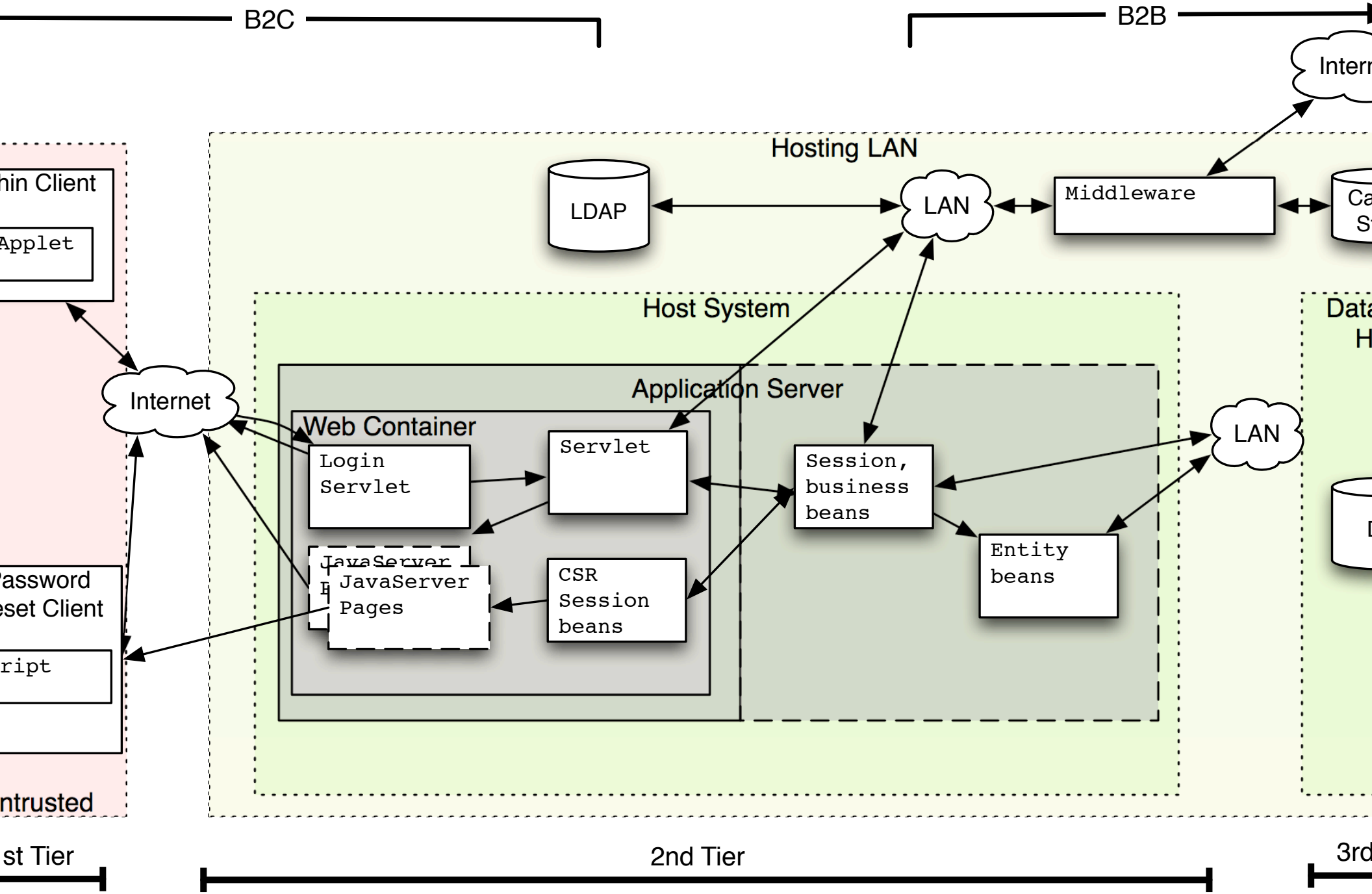
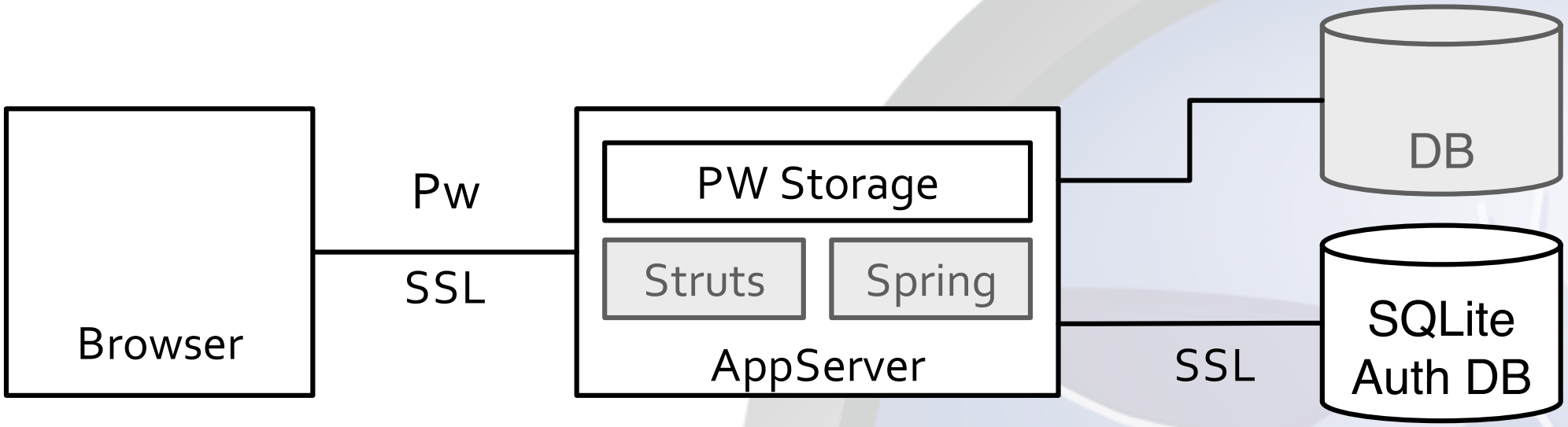


Diagram System/Softwar structure

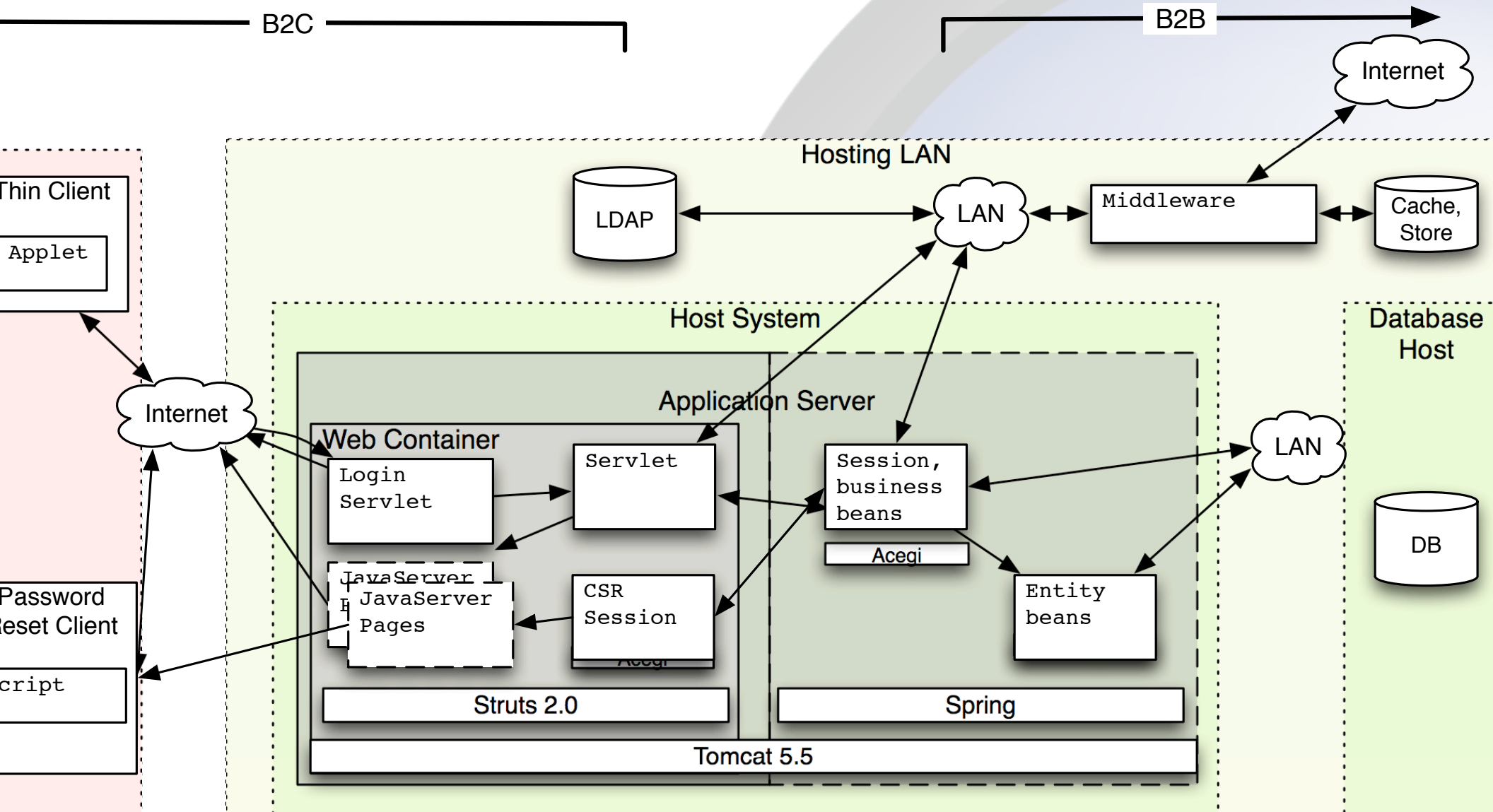


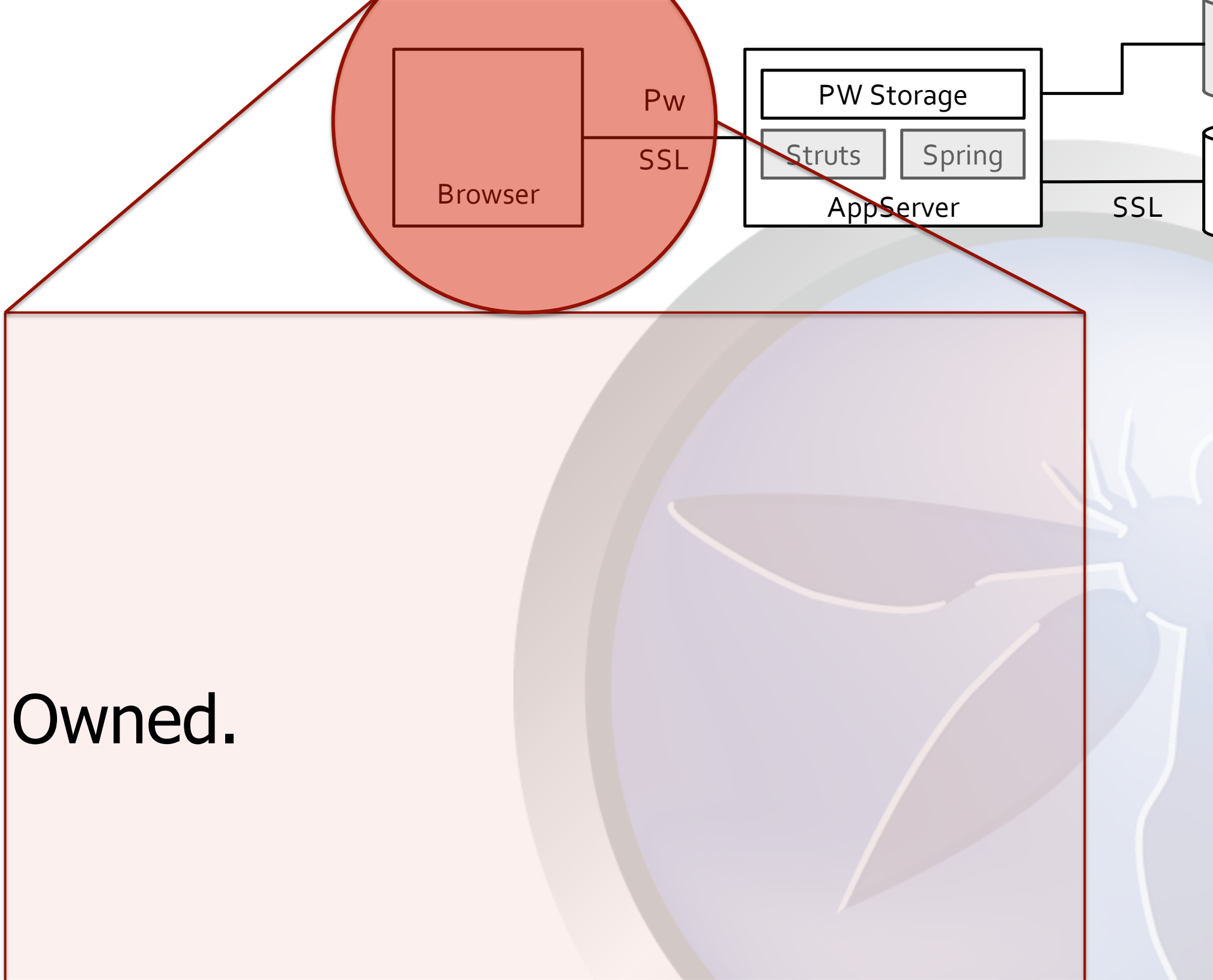
) Acquiring PW DB

) Reversing PWs from stolen booty

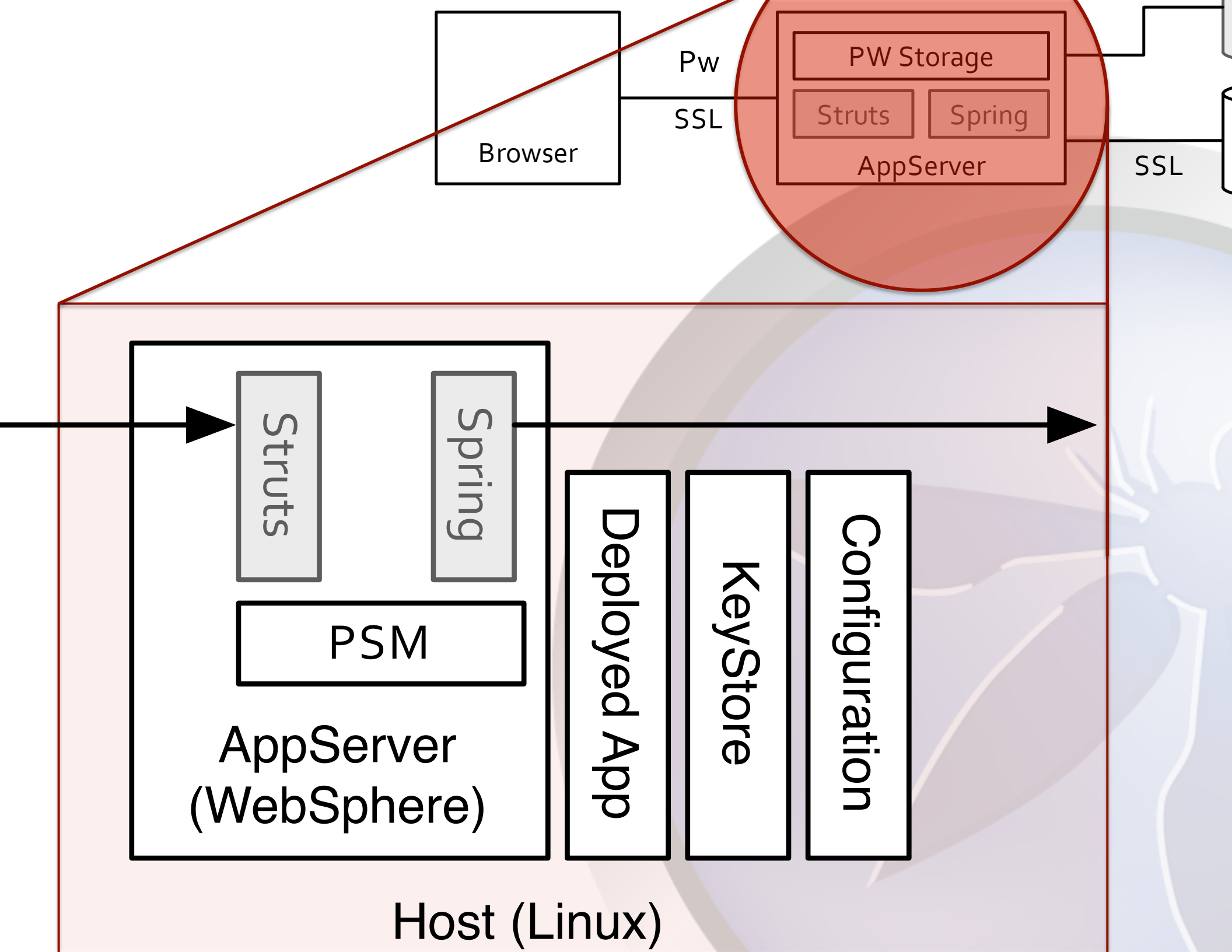
Identify Frameworks

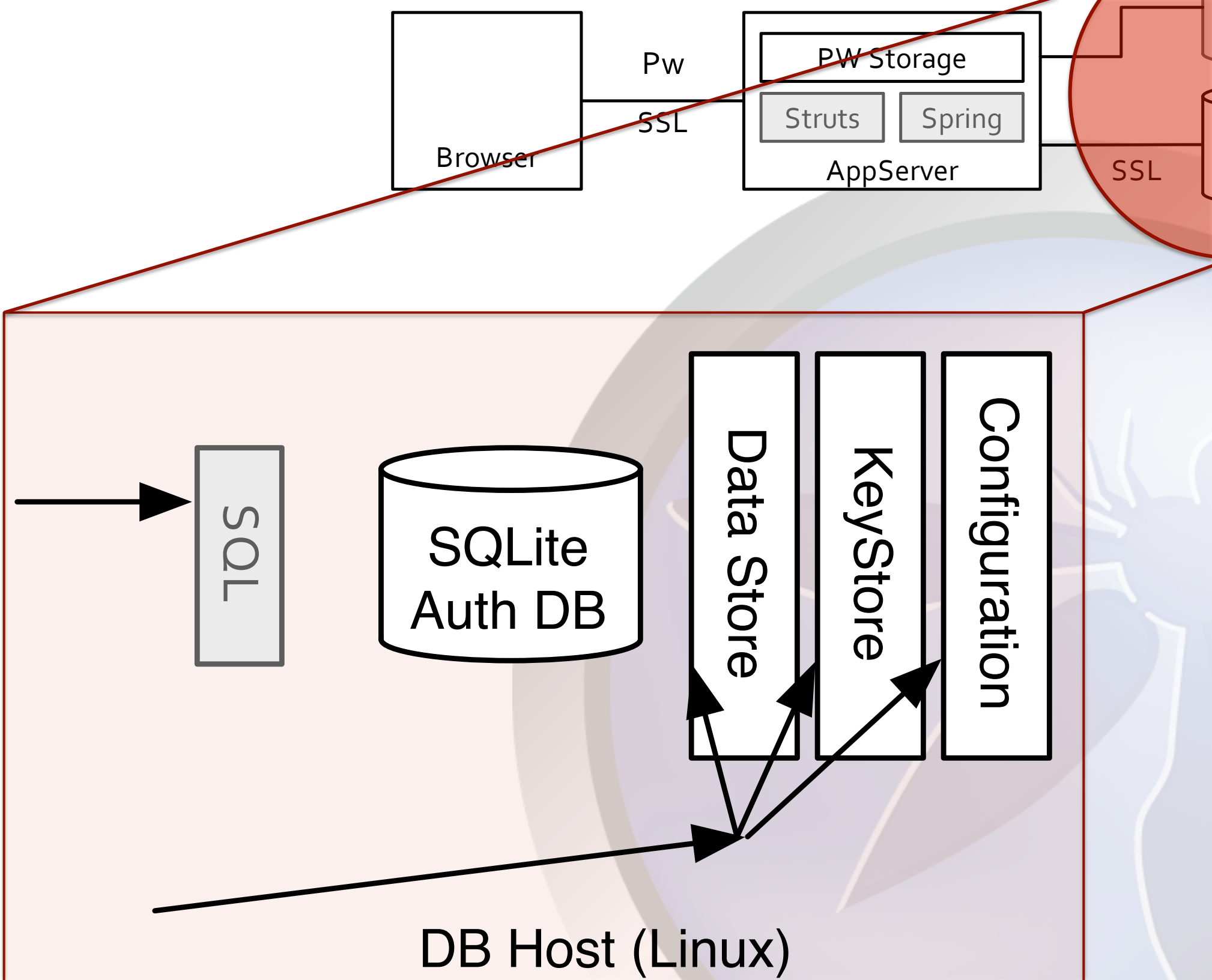
wing frameworks indicates where important service contracts exist
nd 'down'





Owned.


Browser
Pw
SSL
PW Storage
Struts
Spring
AppServer
SSL
Struts
Spring
PSM
AppServer
(WebSphere)
Deployed App
KeyStore
Configuration
Host (Linux)


Browser
Pw
SSL
PW Storage
Struts
Spring
AppServer
SSL
SQL
SQLite
Auth DB
Data Store
KeyStore
Configuration
DB Host (Linux)

Identify Threat Agents

Threat

- Access to the system
- Able to reverse engineer binaries
- Able to sniff the network

ill Level

- Experienced hacker
- Script kiddie
- Insiders

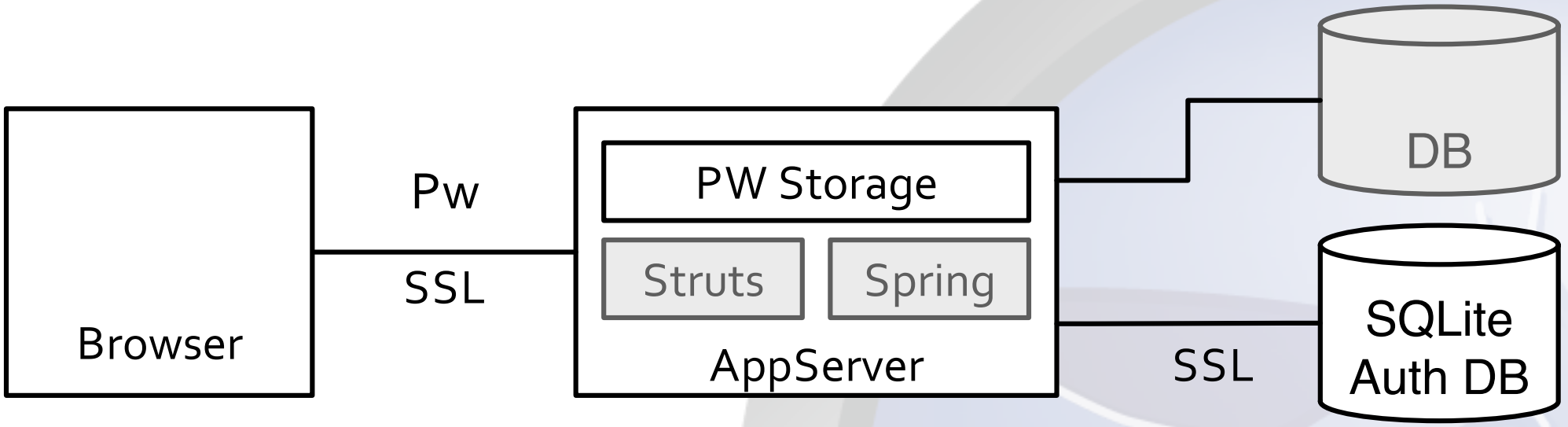
sources and Tools

- Simple manual execution
- Distributed bot army
- Well-funded organization
- Access to private information

reats help

- Encourage thorough thought about how intentions for misuse

Diagram System/Software structure

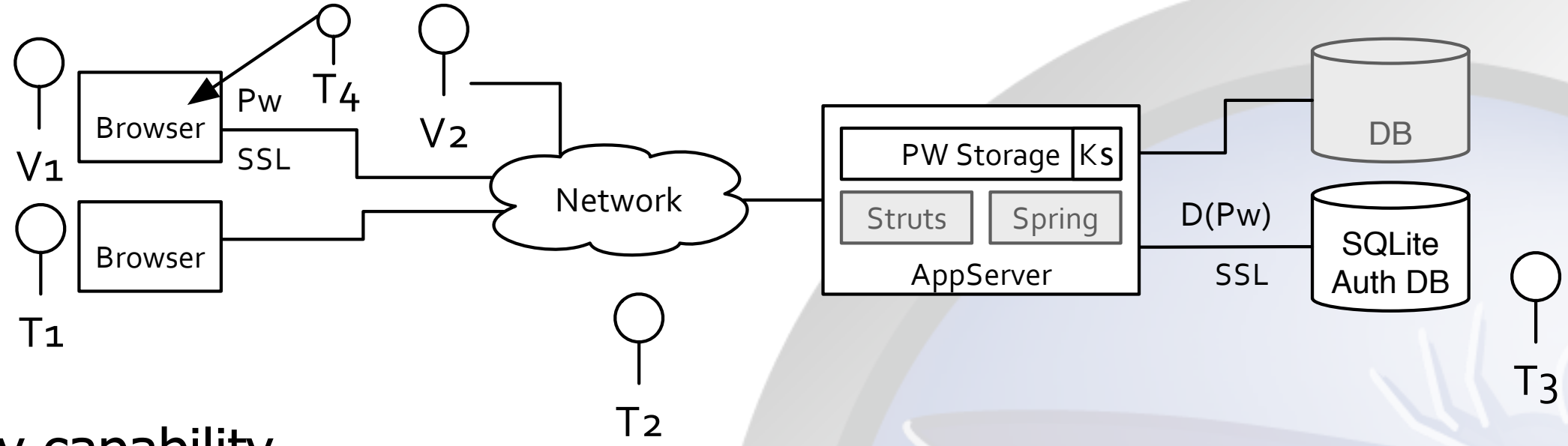


) Acquiring PW DB

) Reversing PWs from stolen booty

Threat Model

1) Acquiring PW DB
2) Reversing PWs from stolen boo



y capability

- Script-kiddie
- **AppSec Professional**
- **Well-equipped Attacker**
- Nation-state

Threat Actors

Threat Actor	Attack Vector
] External Hacker	AV0 - Observe client operations
	AV1 - Inject DB, bulk credentials lift
	AV2 - Brute force PW w/ AuthN API
	AV3 - AppSec attack (XSS, CSRF)
	AV4 - Register 2 users, compare
] MiM	AV1 - Interposition, Proxy
	AV2 - Interposition, Proxy, SSL
	AV3 - Timing attacks
] Internal/Admin	AV1 - Bulk credential export
	AV2 - [T1] style attack

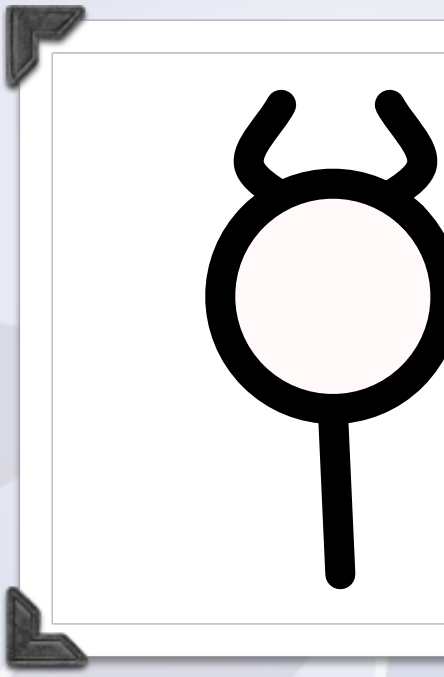
ttacks Specific to PW Storag

1. Dictionary attack
2. Brute-force attack
3. Rainbow Table attack

Well-equipped

4. Length-extension attack
5. Padding Oracle attack
6. Chosen plaintext attack
7. Crypt-analytic attack
8. Side-channel attack

Nation State



Identify Domain-specific Attacks

Attacks and Capabilities

"Top – N" Lists

- SQLi
- Dictionary Attacks

Best Practices

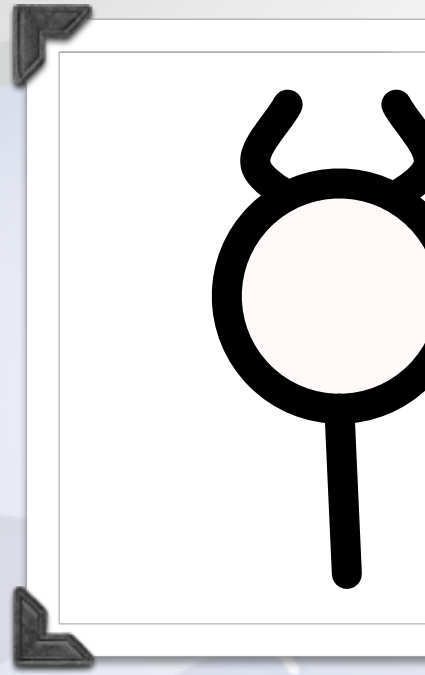
Threat Intelligence

Data feeds

n We Successfully Attack a Has

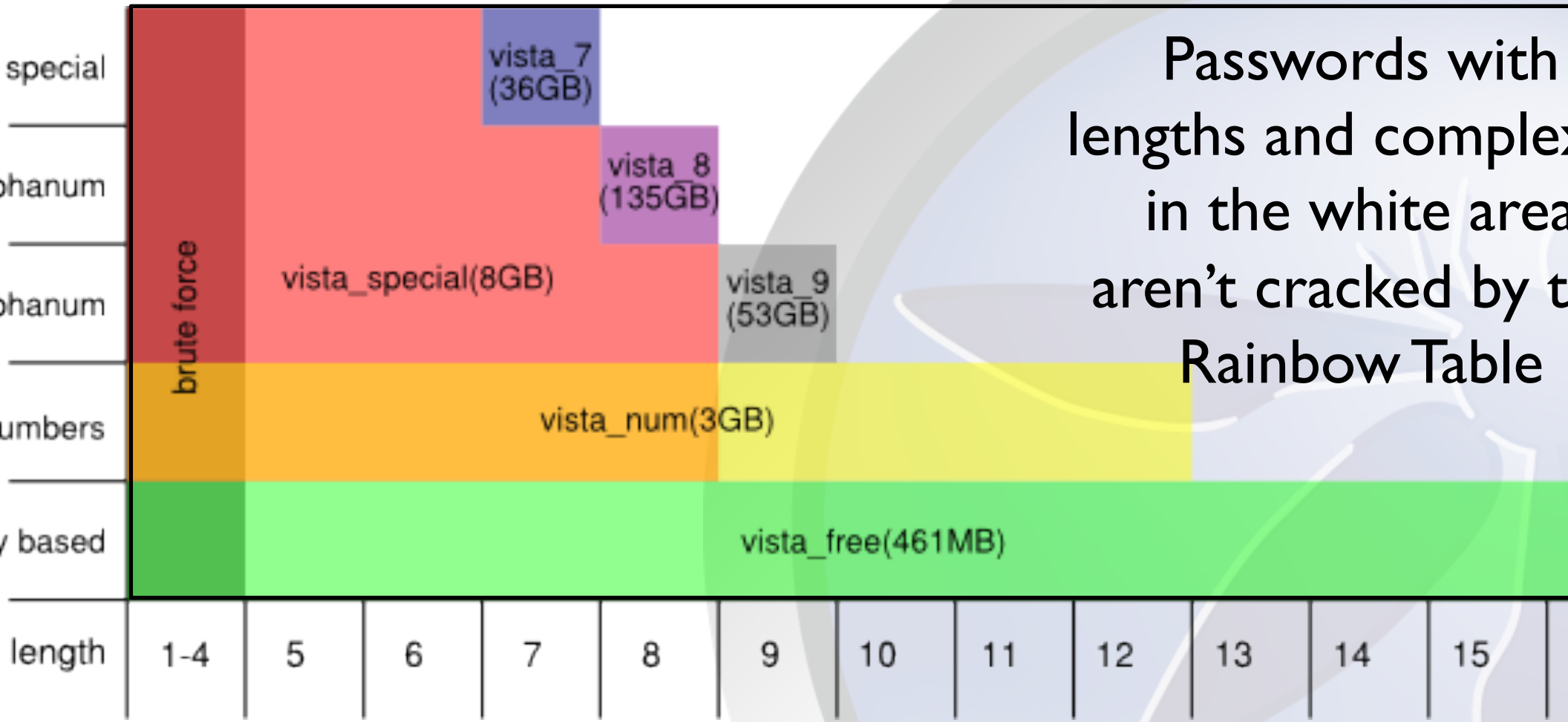
epends on the threat-actor...

- Script-kiddie
- AppSec Professional
- Well-equipped Attacker
- Nation-state



the algorithm supported by a
ool?

nbow Tables: Fast but Inher Limitations



Source: ophcrack

es are crafted for specific complexity and le

Table Sizes

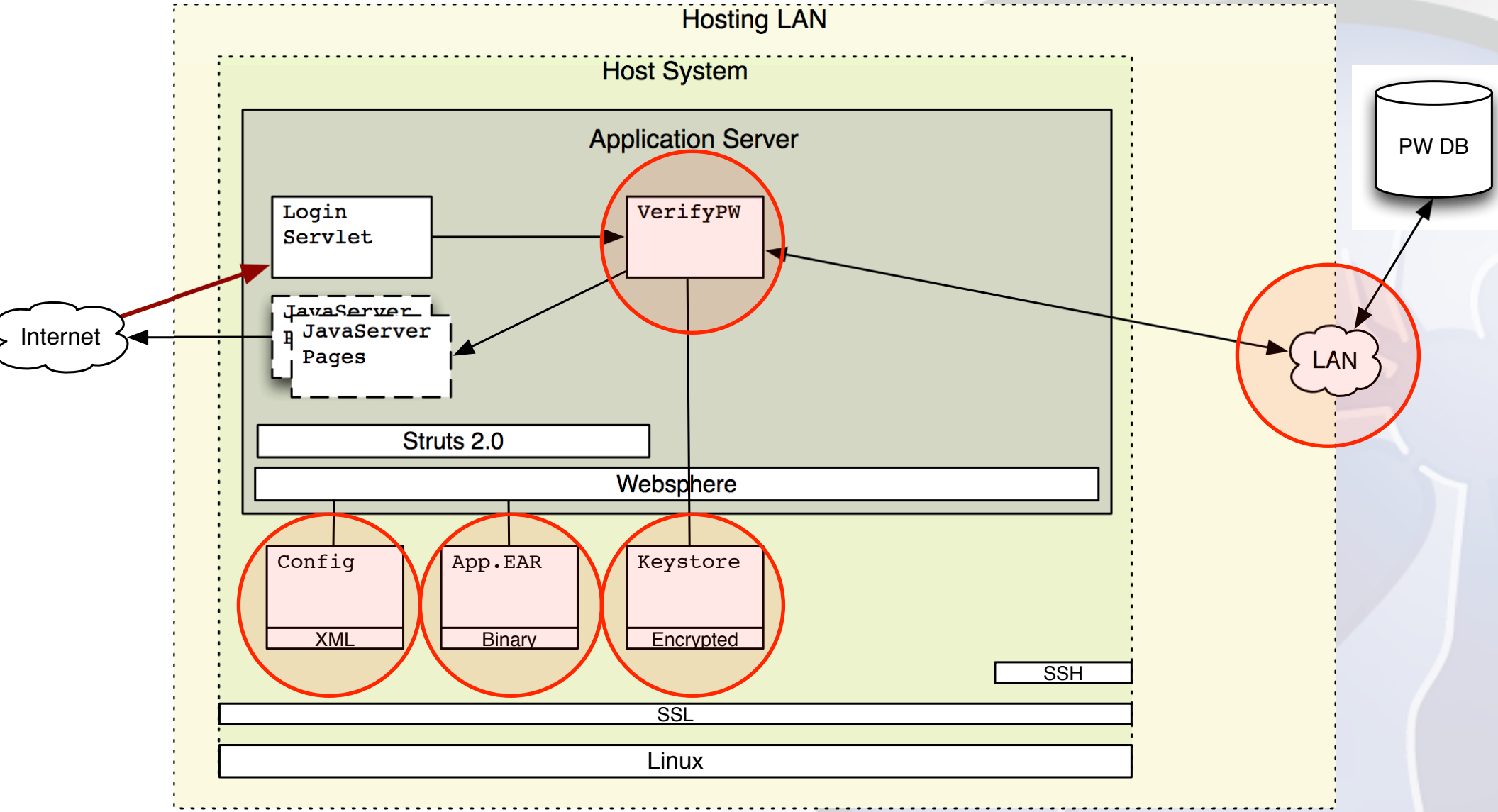
Search Space	Lookup Table (Brute Force)	Rainbow Tab (NTLM hashe
307,000 word dictionary	16 MB	461
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^4$	338 MB	8.0
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^5$	21 GB	8.0
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^6$	1.3 TB	8.0
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^7$	87 TB	8.0
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^8$	5,560 TB	134.
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^9$	357,000 TB	No t
$(a\text{-}z \mid A\text{-}Z \mid 0\text{-}9)^{10}$	22,900,149 TB	No t

Per User Table Building

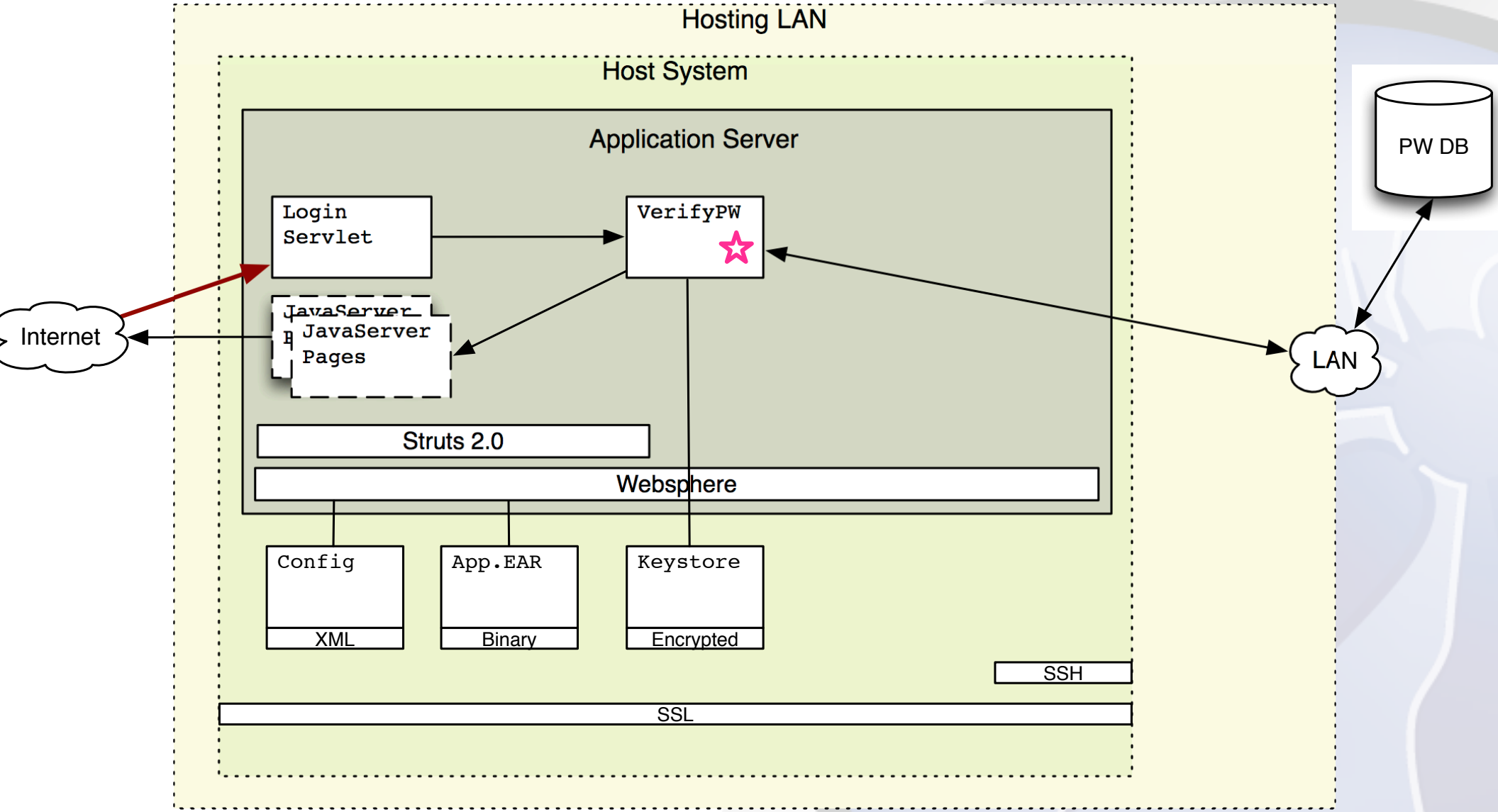
Brute Force Time for SHA-1 hashed, mixed-case-a alphanumeric password

		8 Characters	9 Character
cking a single n (32 M/sec)	NVS 4200M GPU (Dell Laptop)	80 days	13 years
cking a single n (85 M/sec)	$169 Nvidia GTS 250	30 days	5 years
cking a single n (2.3 B/sec)	$325 ATI Radeon HD 5970	1 day	68 days

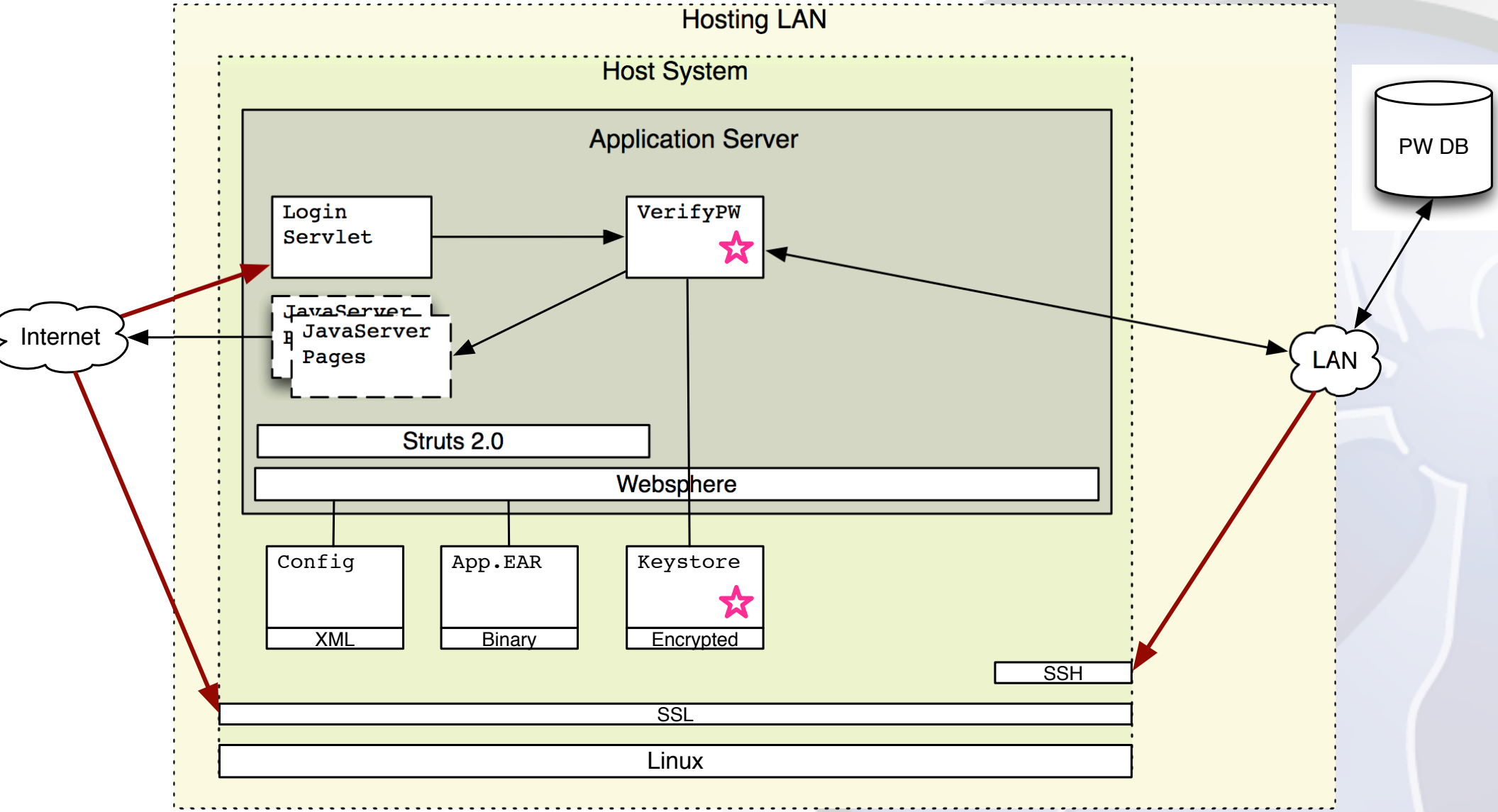
Find Ancillary Targets



Key Theft (technology)



Alternative to Key Theft



Matrix

ATK-1.1 : Resist "chosen plaintext" attacks - Attackers possessing system access and a valid account [T1] (*See [T1.AVA01], [T5.AVA11]*) should ***not*** be able to:

Discern password protection scheme

Attack another user [V2] in $O(V1_{pw})$ time

Choose and use set(s) of credentials and discern scheme cryptographic secrets

ATK-1.2 : Resist "brute-force" attacks - Attackers possessing access to PW DB and knowledge of protection scheme (*See [T1.AVA02]*) should not be able to:

Discern individual account credentials in reasonable time

- Difficulty >> $O(V1_{pw})$
- Calendar time >= 1 yr

Discern all account credentials in reasonable time

- Difficulty >> $O(V_{pw})$ * Population(V)
- Calendar time >= 1 yrs

ATK-1.3 : Resist D.o.S. as a result of entropy/randomness exhaustion (*See [T5.AVR08]*);

ATK-1.5 : Resist identifying identical credentials by observing <protected>(PW) (*See [T1.AVA00], [T3.AVR03, T5.AVA12, T5.AVR04]*);

ATK-1.6 : Prevent attackers from generating valid forms <protected>(PW) without knowing credentials and possessing any/all secrets;

ATK-1.7 : Prevent attackers from exfiltrating any ancillary secrets associated with <protected>(PW), such as MAC or encryption keys (*See [T3.AVA05-T3.AVA09]*

ATK-1.8 : Prevent attacks from gaining information about plain/digest-text through side-channel or timing attack: for instance, gauging how long equality check between two digests takes (*See [T5.AVR05]*); [*TA]#

Matrix (Subtle)

SCC-1.1 : Prevent attackers from gleaning information about server secrets or [V1] plaintext through multiple chosen plaintexts (such as (PW, PW') and (PW', PW'') : PW' = digest(PW)); [RG]#

SCC-1.2 : Prevent attackers from gleaning information due to use of a common key between cipher and mac constructs, such as when CBC-MAC used; [HA]#

SCC-1.3 : Prevent leakage of information (such as password, key material, initialization vectors, etc.) when using cryptographic ciphers, hashes, or MACs.

SCC-1.4 : Assert that input to cryptographic primitives possesses the appropriate level of randomness without imposing such undue requirements on the system so as to easily exhaust its entropy thus denying service;

SCC-1.5 : Bound input to those primitives which fall prey to length-extension attacks;

SCC-1.6 : Take care to avoid padding oracle attacks where applicable;

SCC-1.7 : Take specific steps to prevent primitives from leaking information about plaintext or keys when attackers have access to plaintext/ciphertext pairs.

Thank you for your time

Who owns the table?

	What	How	Impact	Mitigation
ed, ser	Directly request and gain access to another user's info	• Forceful browsing • Failure to demand auth • Session Fixation	PR Incident *Non-compliance* Increase QSA assessment cost	• FD:3.2: session mgmt • SR:2.3.4: URL, forms data • FD: 3.4: Controller design • SD: 1.3: WebSeal integrat • SP:1.3: Demanding Auth.
artner, user	Upload malicious content as part of normal workflow	• Upload exceptional large file • Use file as injection vector • Upload dual-type file (such as GIFAR)	SLA violation *Data loss/ corruption* *Wholesale system breach*	• SP: 9.3: Virus scanning up • FD: 6.1: Upload quota • SP: 2.2: Filtering input • SD: 6.3: Re-encoding files • SR: 6.5: Spec for valid file

usiness Analyst

Business Analyst

(Security) Architect

(Security) Architec

Don't worry about "left to right"

	What	How	Impact	Mitigation
ORIZED user	Directly request and gain access to another user's info	• Forceful browsing • Failure to demand auth • Session Fixation • ***CSRF***	PR Incident *Non-compliance* Increase QSA assessment cost ***Fraud***	• FD:3.2: session mgmt • SR:2.3.4: URL, forms da • FD: 3.4: Controller desi • SD: 1.3: WebSeal integ • SP:1.3: Demanding Aut
artner, ***ed user***	Upload malicious content as part of normal workflow	• Upload exceptional large file • Use file as injection vector • Upload dual-type file (such as GIFAR)	SLA violation *Data loss/ corruption* *Wholesale system breach*	• SP: 9.3: Virus scanning • FD: 6.1: Upload quota • SP: 2.2: Filtering input • SD: 6.3: Re-encoding fi • SR: 6.5: Spec for valid f

hen testing finds an attack:

- First, decide if its *impact* warrants further exploration
- Are additional impacts possible?
- Consider *what* conceptual goals the attack supports
- Then consider *who* could launch the attack against the application

ter analysis converges, iterate secure design

How much is enough?

Incrementally improve *from wherever you are*

Think about organization's 'arch-types'

- B2C, n-tier*
- Mobile
- B2B, Legacy
- ATMs
- RIA**

Within each step, resist urge to do other steps

Start with step for *corresponding SDL activity*

Threat model what's new and different

Alternative Methods

Security Goals

CIA

onfidentiality

limiting access and disclosure to "the right people“; preventing access by or disclosure to "the wrong people”.

tegrity

the trustworthiness of information resources

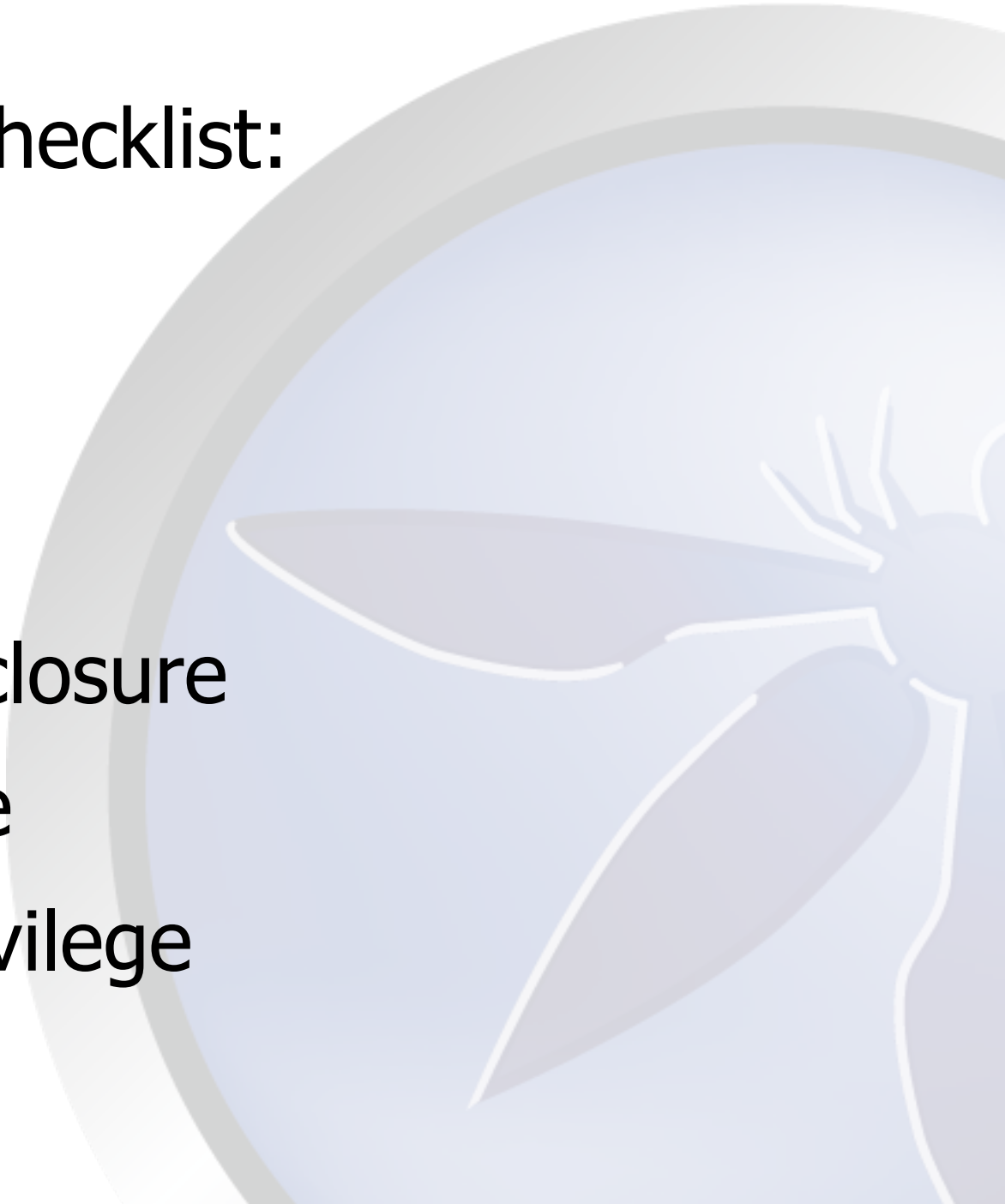
vailability

information systems provide access to authorized users

A Few Words on STRIDE

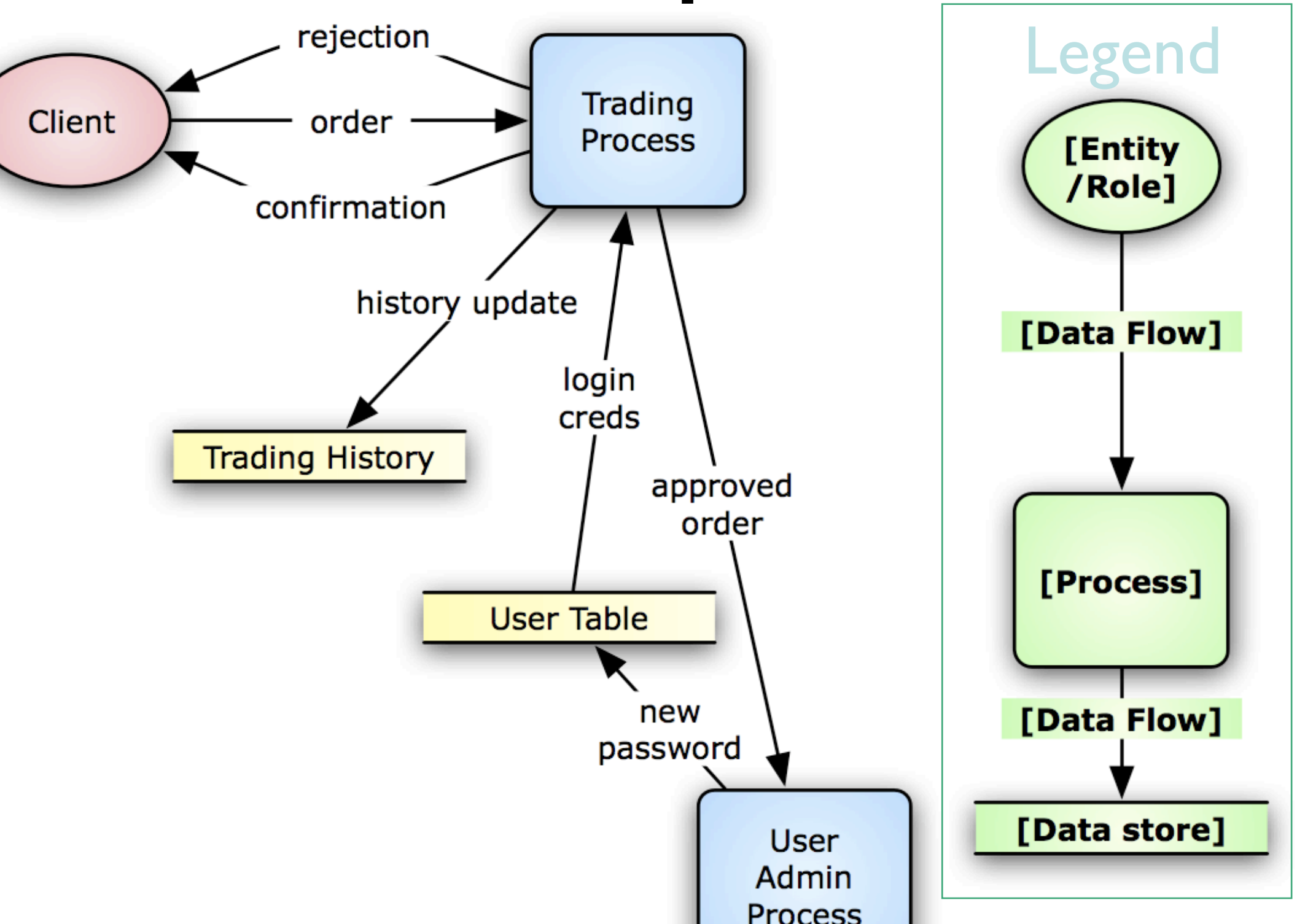
A conceptual attack checklist:

- **S**poofing
- **T**ampering
- **R**epudiation
- **I**nformation Disclosure
- **D**enial of Service
- **E**scalation of Privilege



- Backed by DFDs

An Example DFD



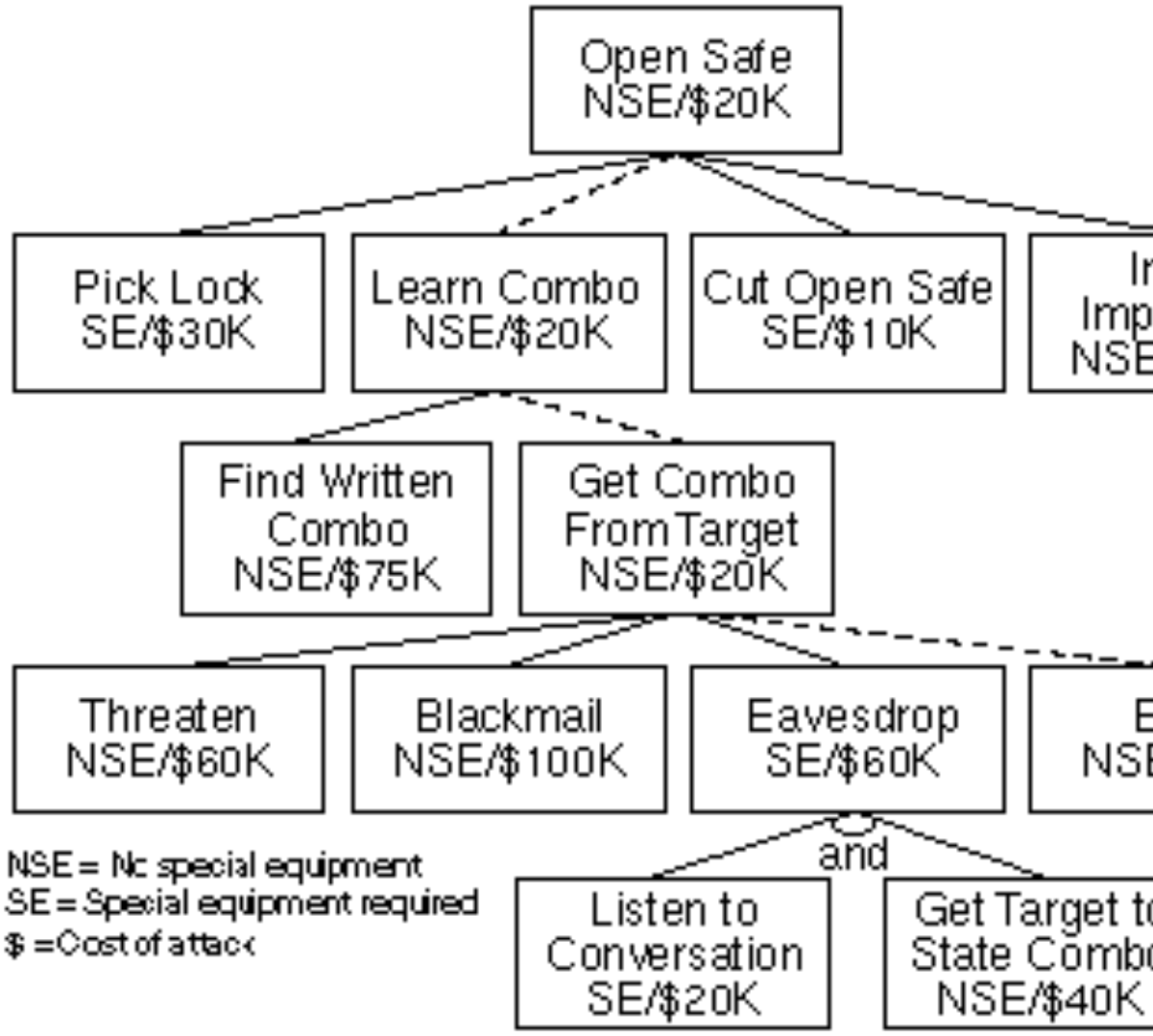
Attack Trees

ggregate attack possibilities

se OR,AND

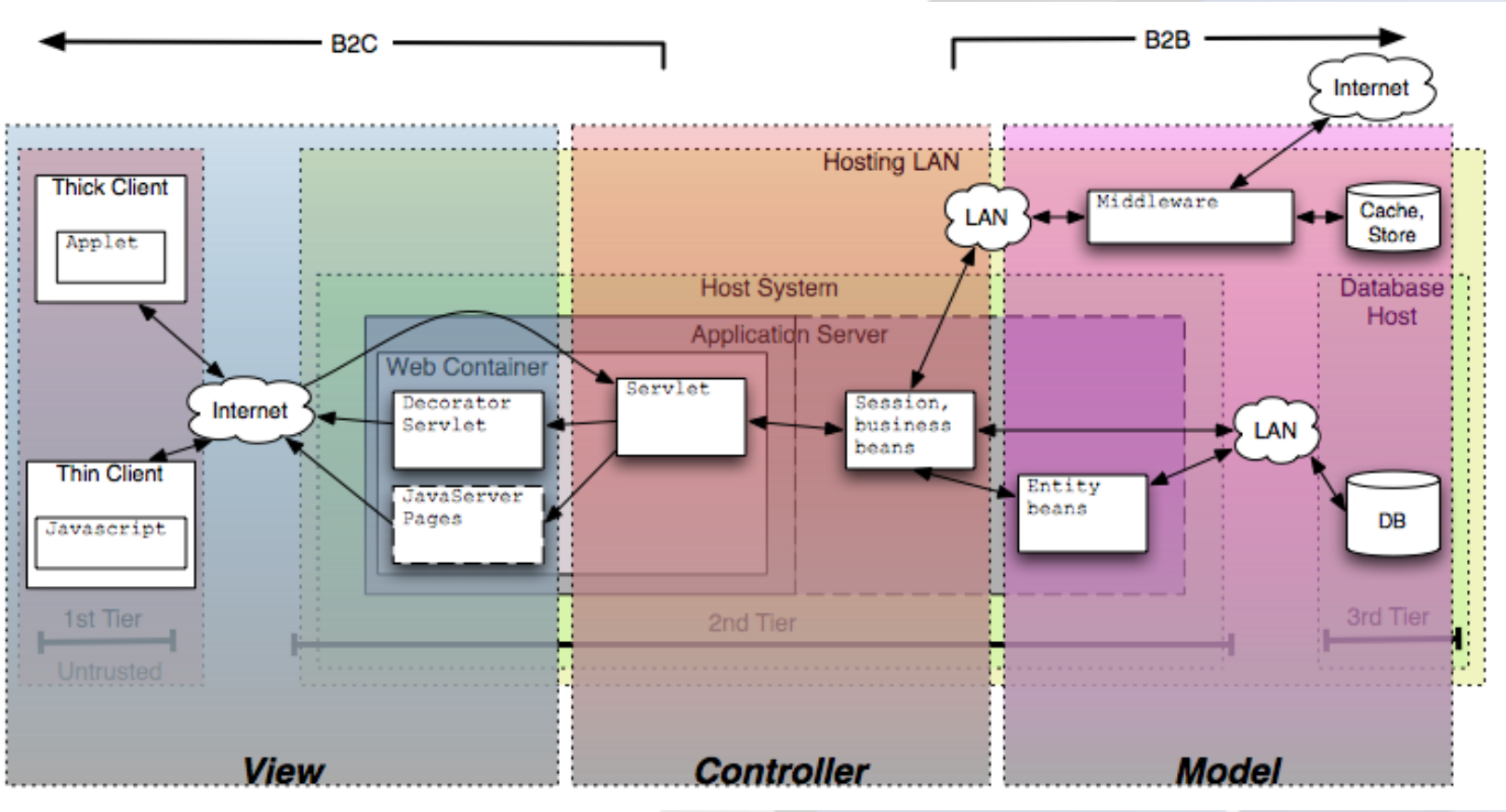
llow for decoration

- Probability
- Cost
- Skills required, etc



From Bruce Schneier's Blo

nnotate with design patter



esign Patterns, isn't that a bit Hifalutin

posed to find exploits

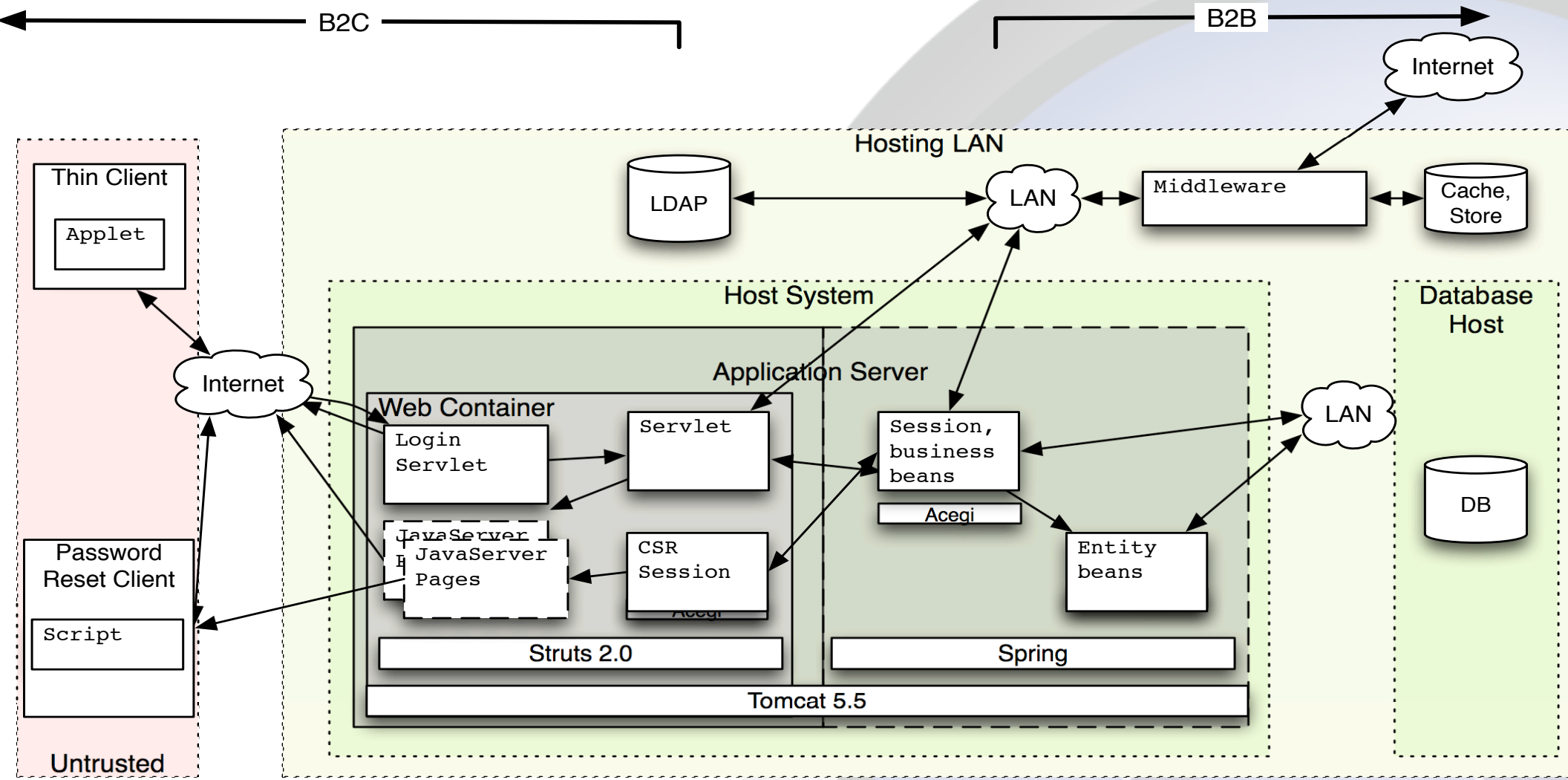
s, I don't have good design docs

Consider Patterns' .. ...

C Element	View		Controller		Model
nponent	Client-side Script	Decorator Servlet	Controller Servlet	Action Servlet	Persistent Store
ponsibility	• Aspects of User experience	• Consuming and hiding error conditions • Filtering output in a target-specific fashion	• Authenticating requests • Filtering / validating input • Limiting user access rights to appropriate workflows • Dispatching actions	• Processing requests • Generating content • Redirecting sessions to different views • Coarse-grain transaction boundary	• ACID transaction properties • Hold data

cument specific standards for implementing
each responsibility

Input Validation – Where does responsibility lie?



xplicit Responsibilities Mean Better Advic

t Side

er Interface

sponsive, instant

ly validation

- Perhaps imperfect
- Perhaps quickly

e the user *good* advice

- Be as specific as possible
- Help the user

Server side

Business logic

Decode

Canonicalize

Apply

- Known-good
- White-list
- Black list

Respond to attack

- Defend self
- Retain intelligence
- Monitor

w thy enemy & how they attack you (REDUX)

	What	How	Impact	Mitigation
ed, ser	Directly request and gain access to another user's info	• Forceful browsing • Failure to demand auth • Session Fixation	PR Incident *Non-compliance* Increase QSA assessment cost	• FD:3.2: session mgmt • SR:2.3.4: URL, forms data • FD: 3.4: Controller design • SD: 1.3: WebSeal integrat • SP:1.3: Demanding Auth.
artner, user	Upload malicious content as part of normal workflow	• Upload exceptional large file • Use file as injection vector • Upload dual-type file (such as GIFAR)	SLA violation *Data loss/ corruption* *Wholesale system breach*	• SP: 9.3: Virus scanning up • FD: 6.1: Upload quota • SP: 2.2: Filtering input • SD: 6.3: Re-encoding files • SR: 6.5: Spec for valid file

Who: Skill, Motivation, Access

What: Technology-agnostic conceptual

ow: The specific tactics that might make attack successful

mpact: the cost of successful attack

itigation: traceability into elements designed to resist, identify, o