Cigital
BUILDING SECURITY IN

Threat Modeling at Scale

SecAppDev 2017

Cigital

Jim DelGrosso

- Run Cigital's Architecture Analysis practice
- 30+ years in software development in many different domains
- 15+ years focusing on software security

@jimdelgrosso

- Executive Director of IEEE Computer Society CSD initiative

http://cybersecurity.ieee.org/center-for-secure-design/

Cigital

About Me

- Andrew Lee-Thorp
- alee-thorp@cigital.com
- @Cigital - Threat modelling, Android tool development, assessments, still code, source code reviews
- > 10 years cutting code
- Occasional speaker

Cigital

What Is Threat Modelling?

- Software design analysis capable of finding flaws
- A defect discovery technique that is part of your SSI
  - You do have an SSI, right?

Cigital

The Defect Universe – Bugs and Flaws

Cross Site Scripting
Buffer Overflow

Weak/Missing/Wrong
Security Control

(Implementation) BUGS

(Design) FLAWS

Code Review

Penetration Testing

Threat Modeling

Cigital

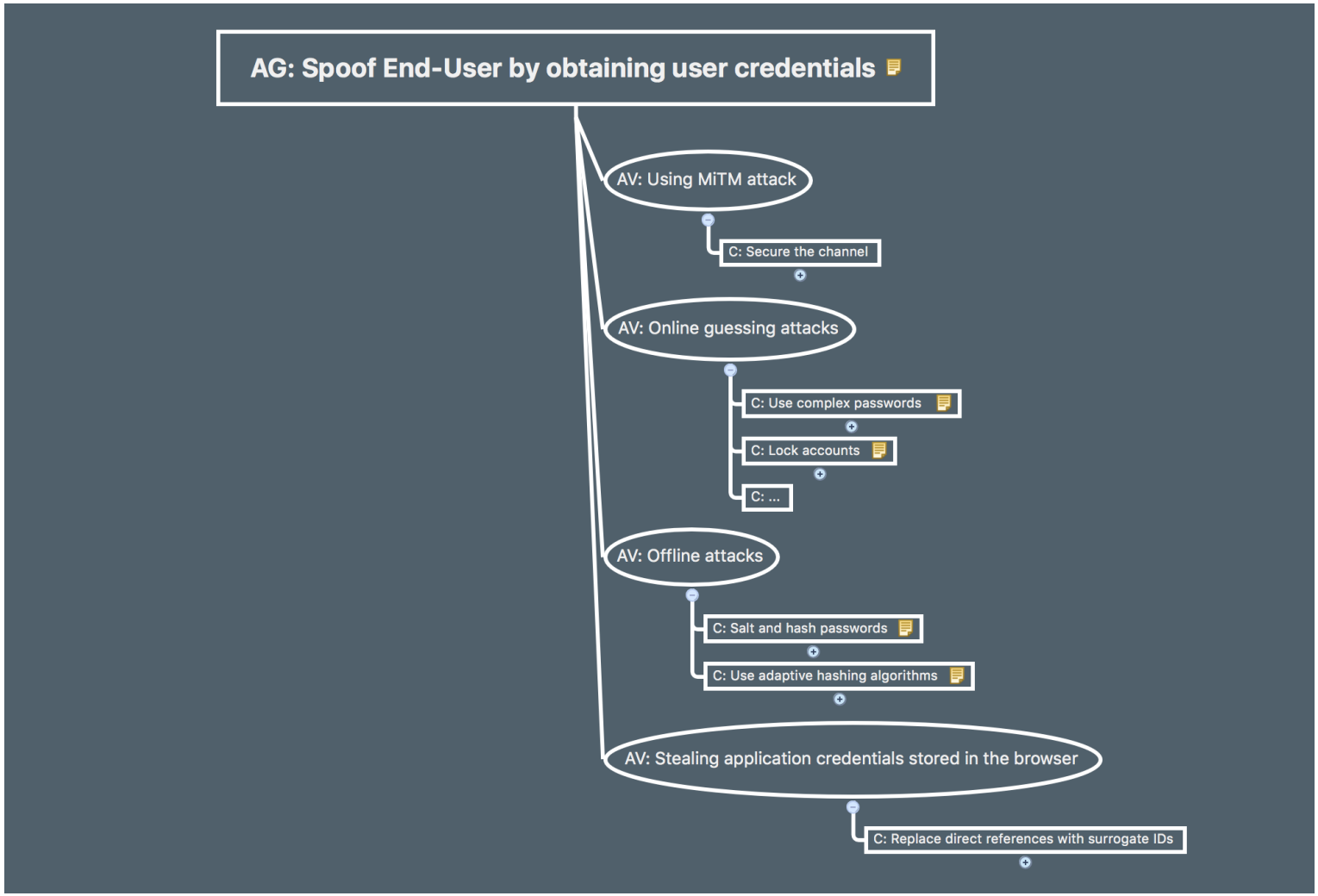
The Need For Threat Modelling

- When done early in the SDLC, it can avoid a lot of pain later in the SDLC
- It complements the other (19) capabilities of your SSI
- Although it can find defects other SSI capabilities find…
- It's the **ONLY** way to find certain defects

Some Threat Modelling Options

- Microsoft Threat Modelling
  - “Software-centric approach”
  - DFDs and STRIDE
- Attack trees
- Cigital Threat Modelling
  - Assets, Threat Agents, Controls modelled directly
  - Component Diagram
- Others
  - PASTA
  - Trike
  - ...

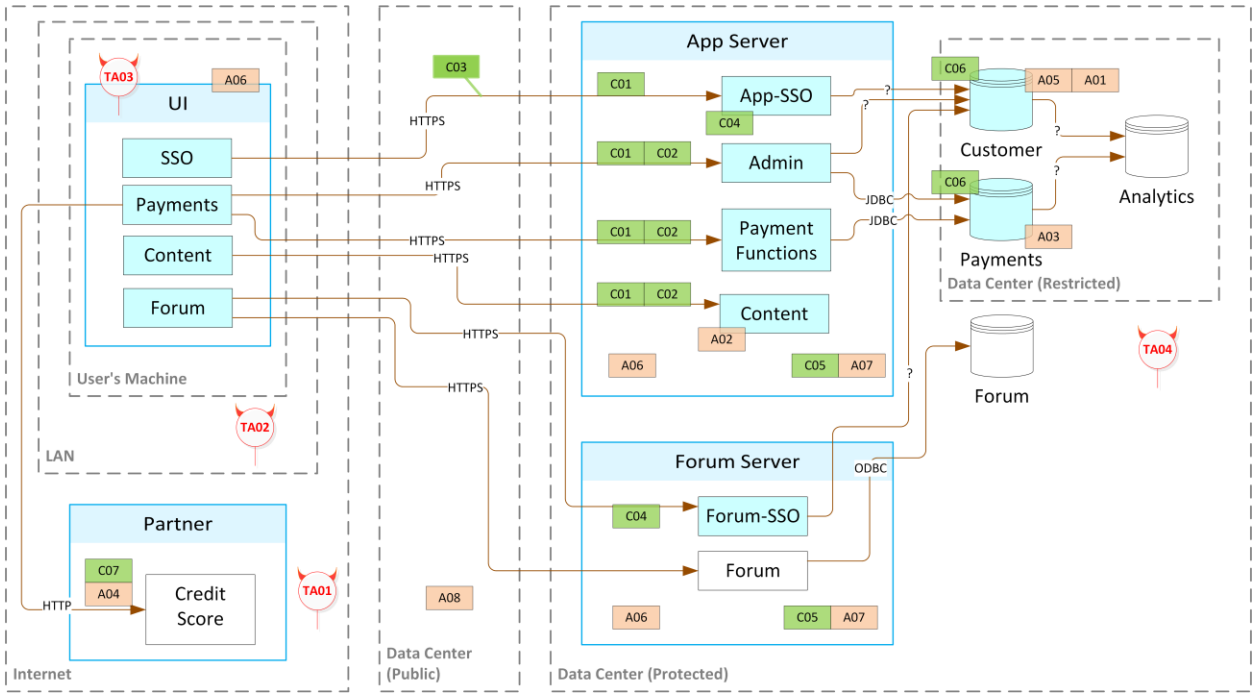
Attack Trees



Threat Model Example

Characteristics of the System Threat Model include:

- Holistic view of application's security posture
- Considers both application and infrastructure
- Builds roadmap for additional security activities



Cigital

Steps For Threat Modeling

- Define scope and depth of analysis
- Gain understanding of what is being threat modeled
  - **Interviews**
  - **Review Existing Models**
- Model the system
  - **Build software model**
  - **Trust Zones**
- Model the attack possibilities
  - **Assets, Controls, Threat Agents**
- Interpret the threat model
  - **Analysis**
- Keep track of your analysis
  - **Traceability Matrix**

Cigital

HOW CAN THIS SCALE?

Scaling In Theory

1. Horizontal scaling (or scale-out)
   - Increase the number of units doing the work
   - Since we're talking threat modelling … more threat modellers

2. Vertical Scaling (or scale-up)
   - Increase the capacity of whoever is doing the work
   - Since we're talking threat modelling … smarter people, process improvements

3. Parallelize

Cigital

Scaling In Theory

4. Divide and conquer
   - Solve more tractable sub-problems (distributed)
   - Re-assemble results
   - Repeat

5. Automate
   - Identify repetitive parts of the process
   - Automate, automate, automate
   - Integrate the automation stream into the manual stream
   - Tooling

The challenge

- https://www.bsimm.com/about/faq/
- … the software security group (SSG) median size is **5** people (smallest 1, largest 130, average 11.7)

Cigital

Scaling In Practice

"Increase the number of units doing the work"

- Hire more security people
  - Maybe, but might not be cost effective
  - Remember SSG is only ~2% of the size dev organisation
- Have more people "do" threat modeling

"Increase the capacity of whatever is doing the work"

- Work longer hours
  - No thank you
- Increase brain function and do things faster, remember more, be more creative
  - Probably not

Scaling In Practice

"Increase the number of units doing the work"

- Move workload out of software security team

- Use development org to **help** build the threat model

Or

- Use development org to **help** build the ~~threat~~ **system** model

Cigital

Scaling In Practice

"Increase the capacity of whoever is doing the work"

- By doing less (off-loading work to dev org) … the software security team does more (analysis) tied to software security

"Divide and conquer"

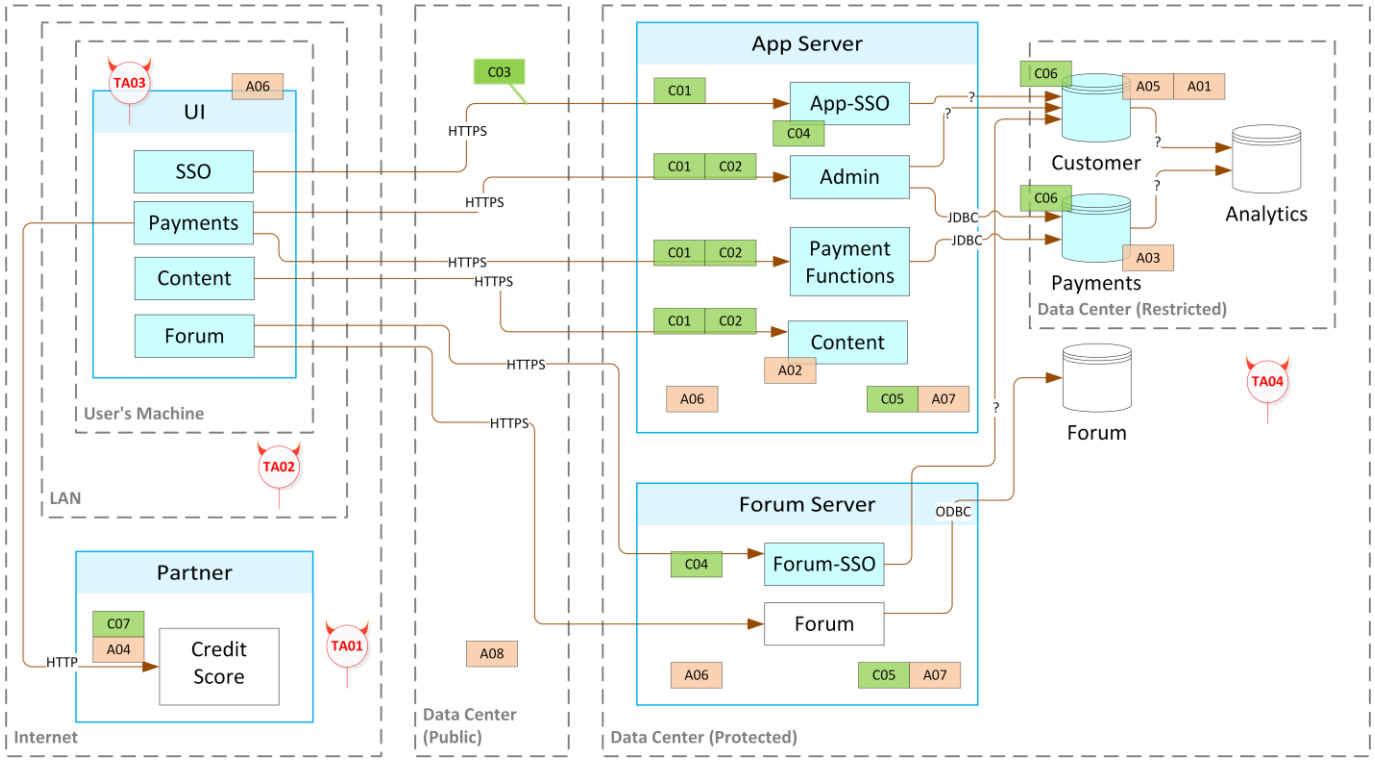
- Analyse design patterns or archetypes (later)

"Automate"

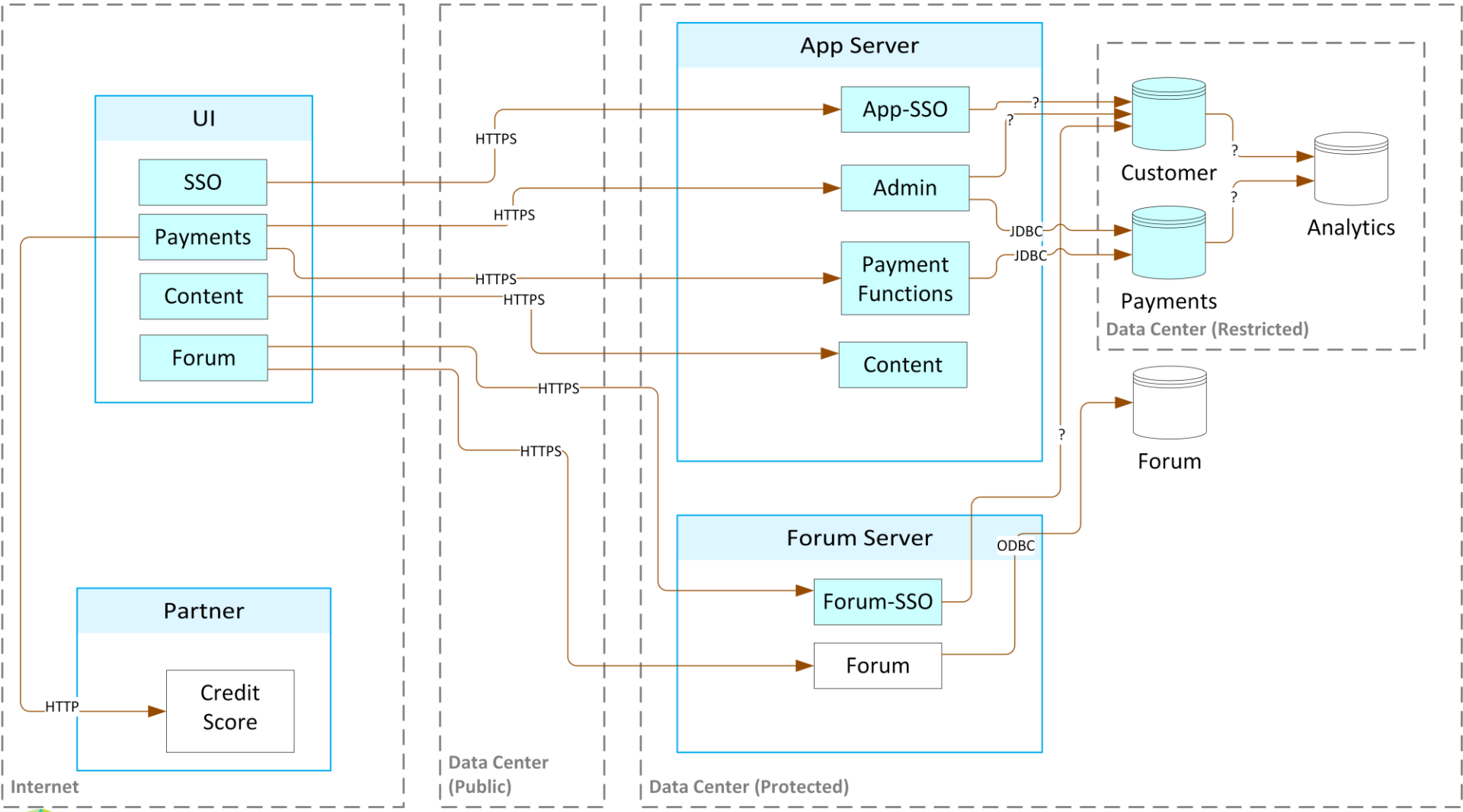
- Analysis of the low-hanging fruit … once again the software security team does more (analysis) tied to the hard software security problems

Back to Our Sample Threat Model

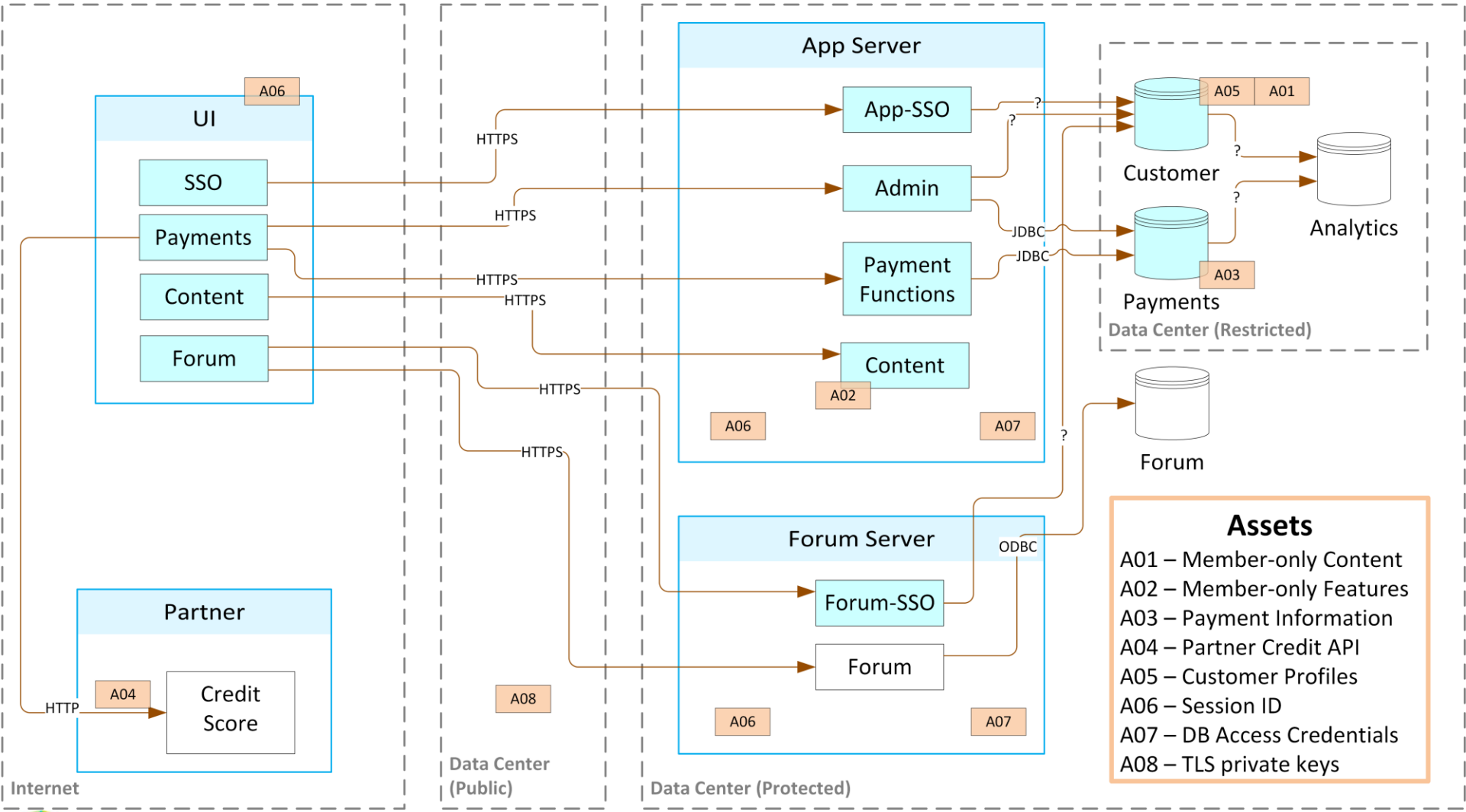
- The modelling end goal is something like the diagram below
- How can we get there efficiently?



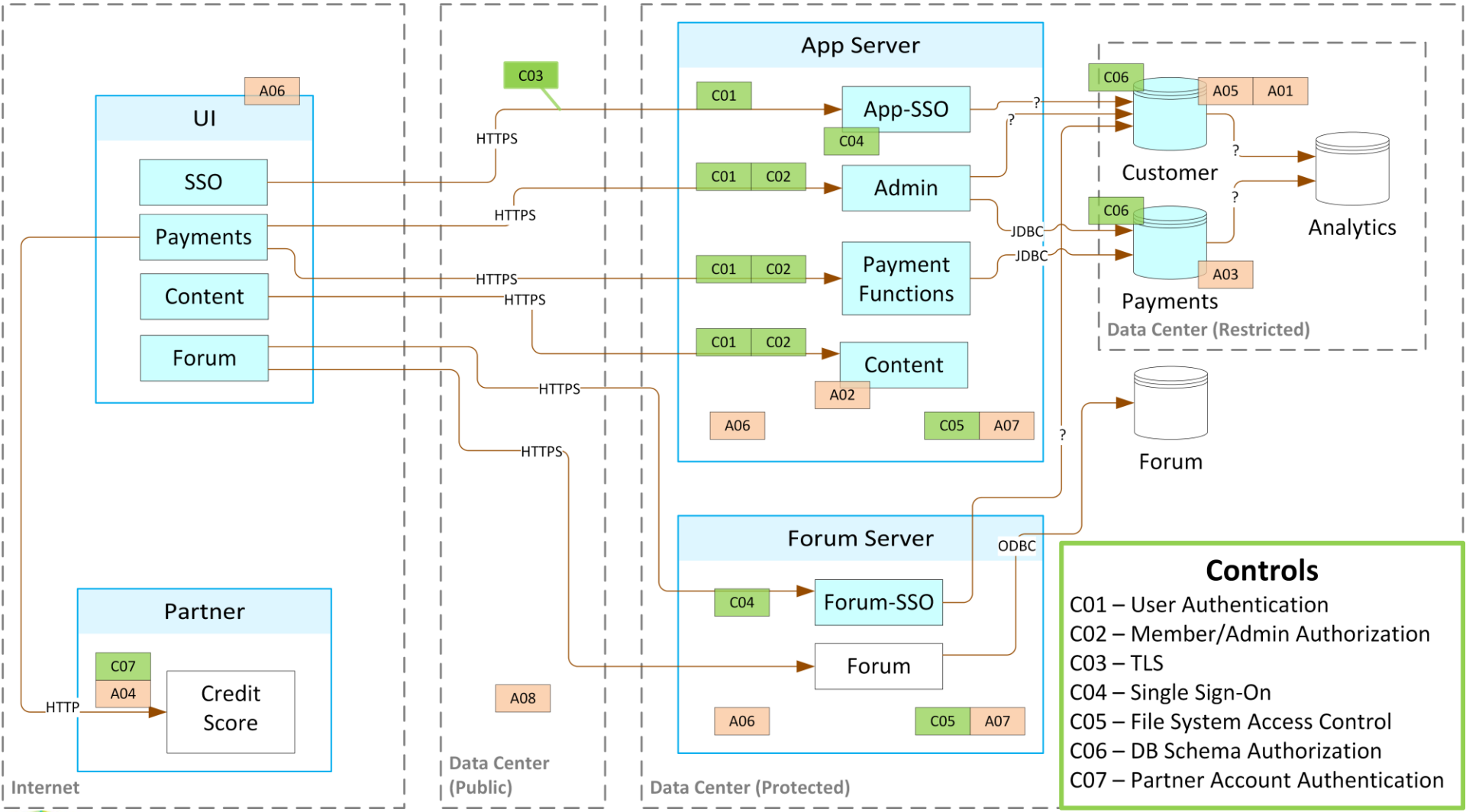
Who Knows About Components and Connections?



Who Knows About Assets?



Who Knows About Controls?

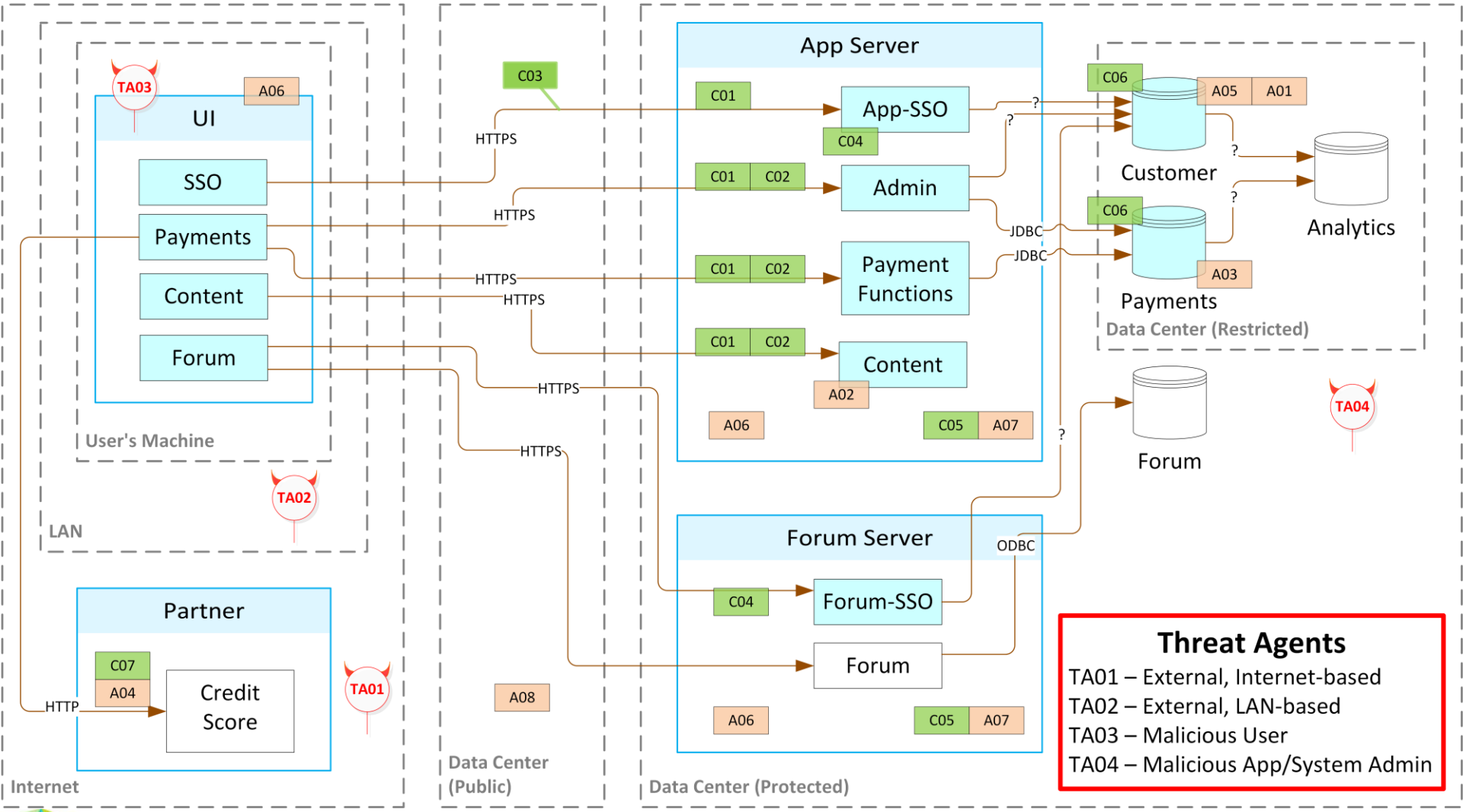


Controls

C01 – User Authentication
C02 – Member/Admin Authorization
C03 – TLS
C04 – Single Sign-On
C05 – File System Access Control
C06 – DB Schema Authorization
C07 – Partner Account Authentication

Cigital

Who Knows How Threat Agents Attack System?



Cigital

PAIN POINTS

There will be pain points … these need to be unblocked

Pain Points

- Developer: “my system is a framework … I can’t model a framework”
- “My control is distributed, I don’t know where to put it”
- Developer: “I don’t know how my system is deployed”
- Developer: “What’s in it for me?”
- Terminology confusion

Terminology

Attack Surface

Threat Model

Attack Vector

Control

Asset

Impact

Likelihood

Risk

Threat Agent (Attacker)

Attack Pattern

Mitigation

Attack

Exploit

Cigital

Terminology Confusion

- OWASP:

Authorization — is mediating access to resources …
Access control and Authorization mean the same thing

- ISO/IEC 10181-3:

Authorization is the a-priori provisioning of entitlements

Access control check is the access decision function …

- Signed data versus MAC-ed data

Cigital

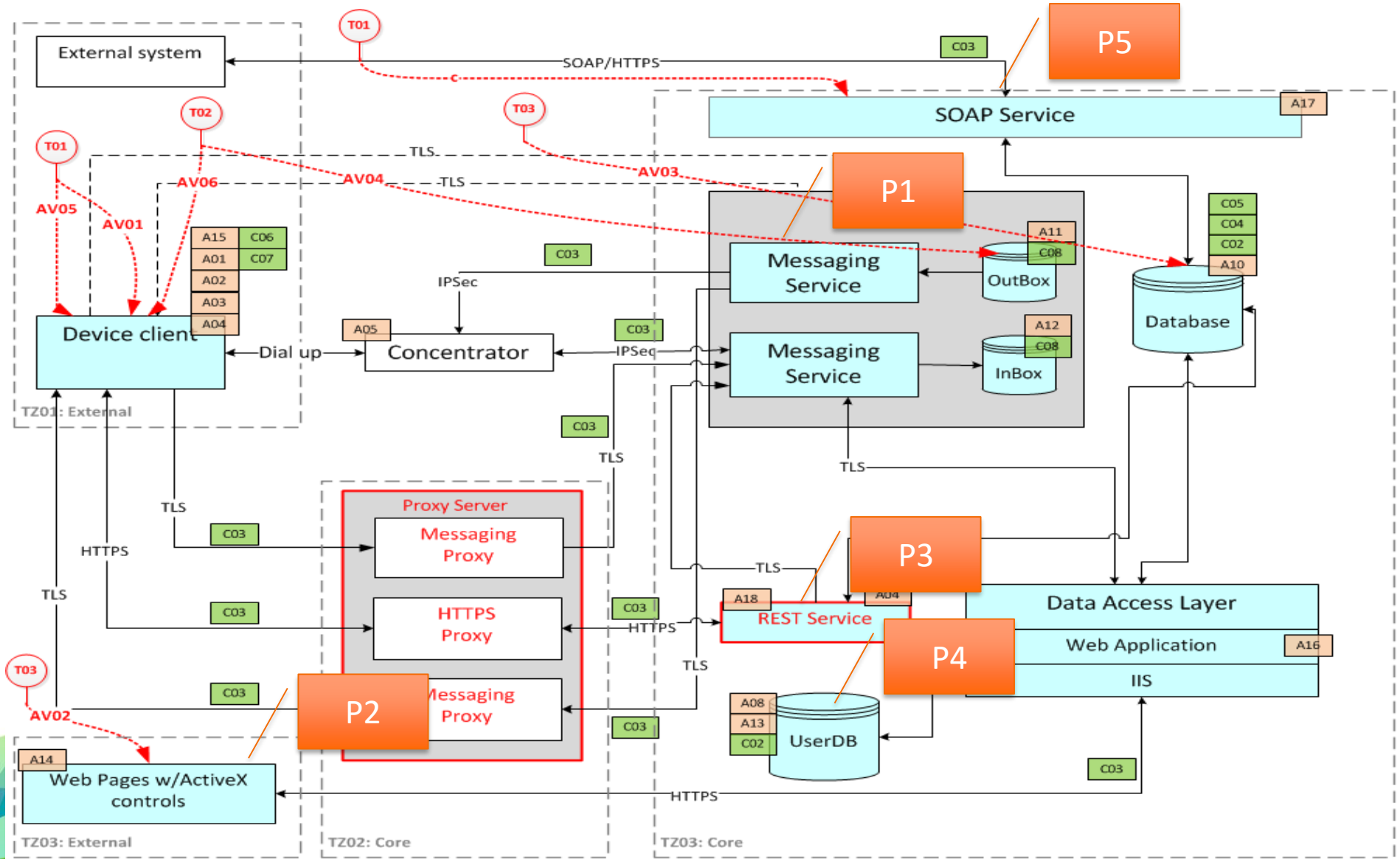
ARCHETYPES

Reusing design patterns

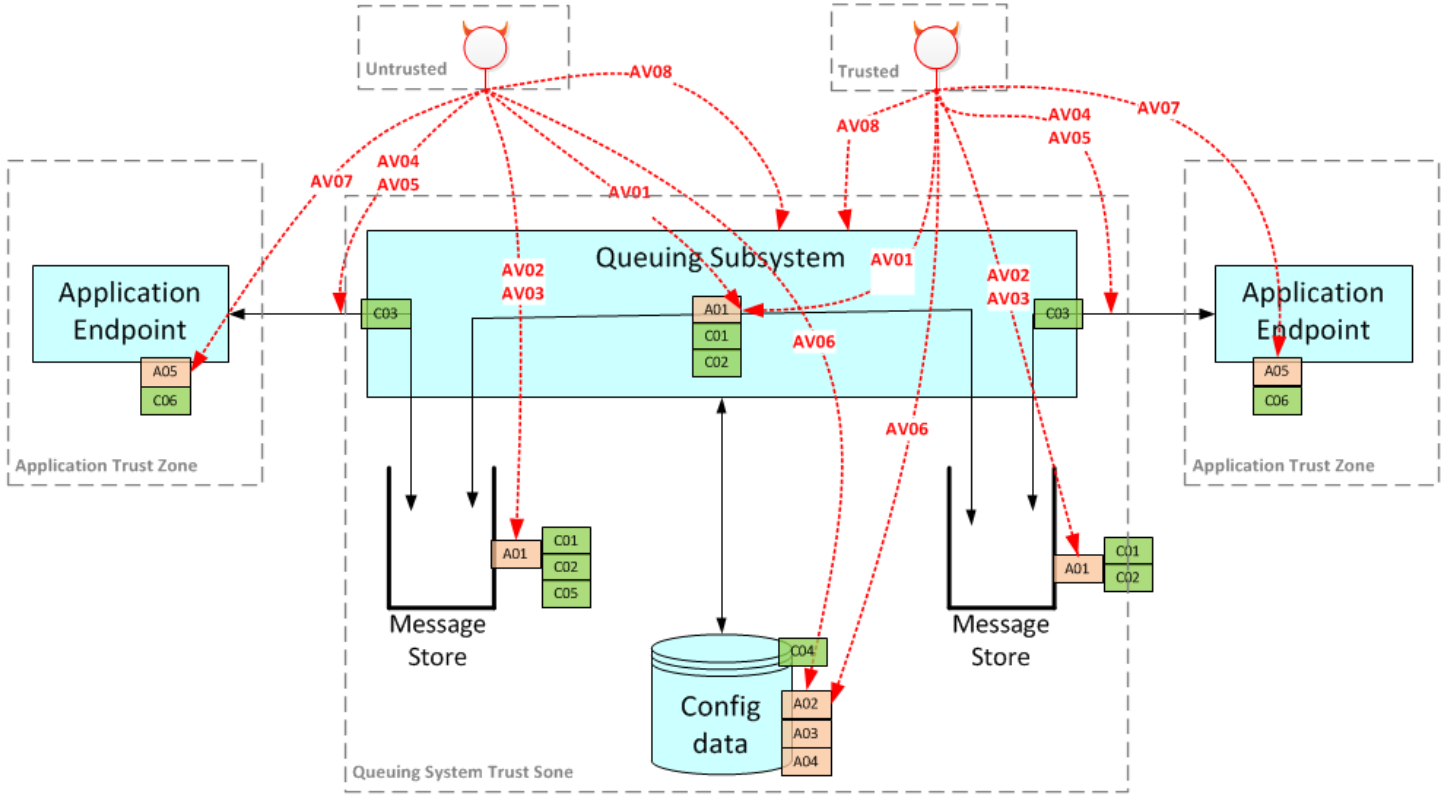
Patterns raise the abstraction level

- Humans think in terms of patterns!
- Threat modeling experts use pattern based approach (*implicitly*)
- When patterns are *implicitly* understood
  - Patterns are not comprehensible
  - Approach is not scalable
- Patterns need to be *explicitly* understood
  - Explicit patterns are comprehensible
  - Consistent
  - Efficient/Scalable

Archetypes Everywhere



Build a Library of Threat Models



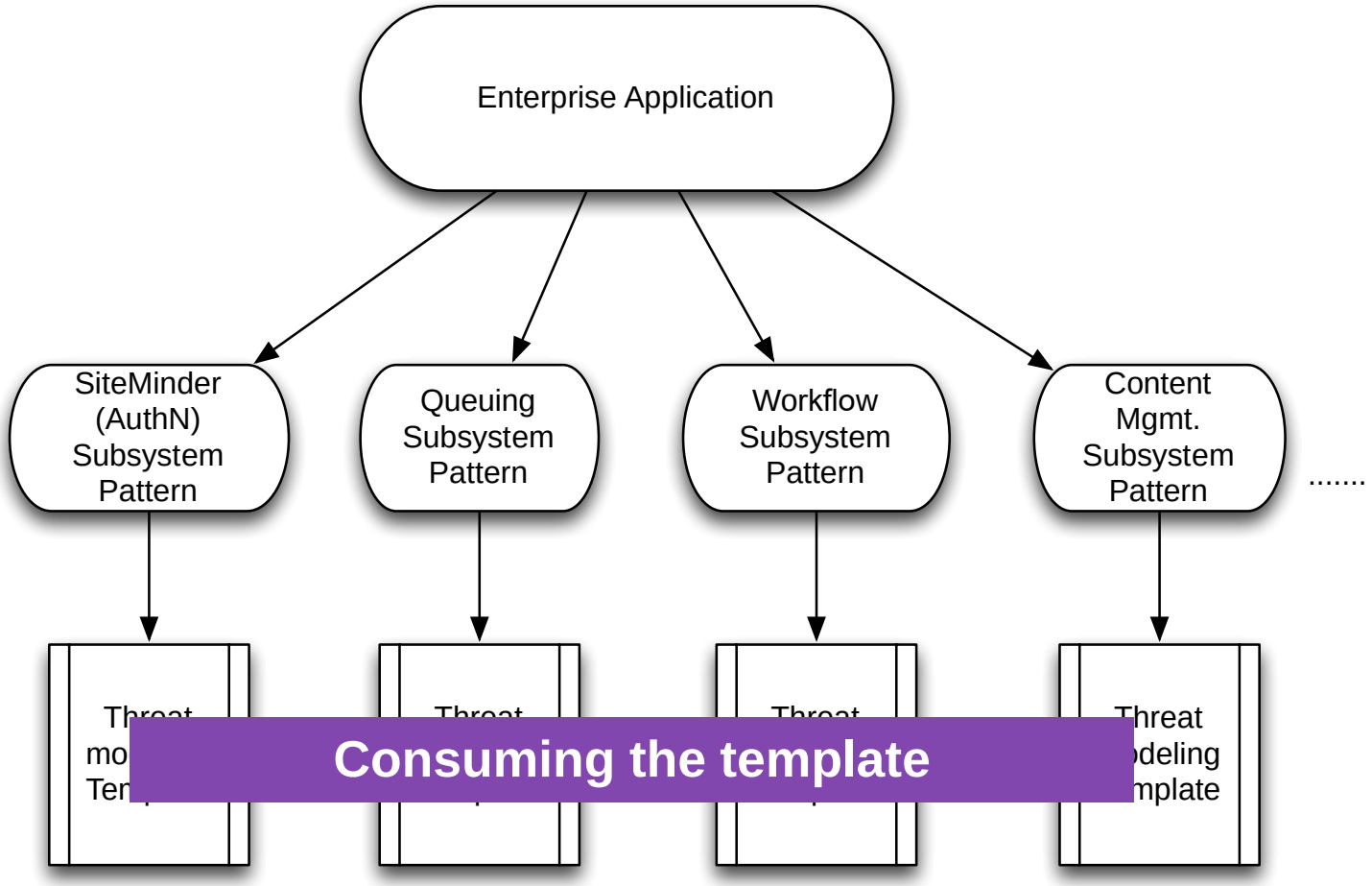
Assets
A01: Messages
A02: Queue definitions
A03: User profiles
A04: Policy Decision Data
A05: Application configuration data to access queuing system

Required Controls
C01: Encryption of messages
C02: Integrity control on messages
C03: Queuing subsystem authentication control
C04: Database access control
C05: Message store access control
C06: Configuration data access control

Attack Vectors
AV01: Read/modify with messages in transit
AV02: Read with messages In the message store
AV03: Modify message in the message store
AV04: Unauthorized users publish messages
AV05: Unauthorized users receive messages
AV06: Get unauthorized access to the admin interface to modify configuration data
AV07: Compromise queuing system authentication credentials
AV08: Denial of Service

Cigital

Using Archetypes



Consuming the template

- Checklist for Message Queue pattern
  - AV01: Read, modify, tamper messages in transit
    - Description: A man-in-the-middle attacker can read, modify … messages in transit
    - Expected control: An authenticated, confidential channel
    - When to apply: (1) Attacker has access to the message queue, (2) No channel protection applied, (3) …
  - AV02: Read messages from store persistence
  - AV03: Unauthorised users publish messages

  - Assets

Archetypes: advantages

- Each pattern is well understood from a security viewpoint
- Catalogue of patterns is accumulated over time
- Archetypes jump-start the analysis
  - Common assets, controls, threat agents, expected trust boundaries
- Covers the low-hanging fruit
- **Using archetypes does not require high-level software security expertise**

Archetypes disadvantages

- “Cross-pattern” interactions, can’t consider in isolation, can’t offload deeper analysis and second attacks.
- Tempting to force a pattern to fit your system

WORKSHOPS

Multi-disciplinary brainstorming

What is a workshop?

- Threat **identification** exercise facilitated by a security expert – can be the satellite
- Development, architects, deployment, QA, product management, support/ops, all in one place

Why run a workshop?

- Having a single analyst can be false economy, e.g. multiple question-answer round trips
- New threats and perspectives on an application when everyone contributes with their view and knowledge

CONCLUSION

Cigital

Major Benefits of Threat Modeling at Scale

You're threat modeling more applications!!

- Finding defects that cannot be found any other way
- Avoiding headaches later in the SDLC process
- Raising awareness
- Gaining insight about **YOUR** frequent design flaws

Reduce defect density

- Guidance
- Training
- Design patterns and/or checklists
- Libraries
- Etc.

Cigital
BUILDING SECURITY IN

Thank You