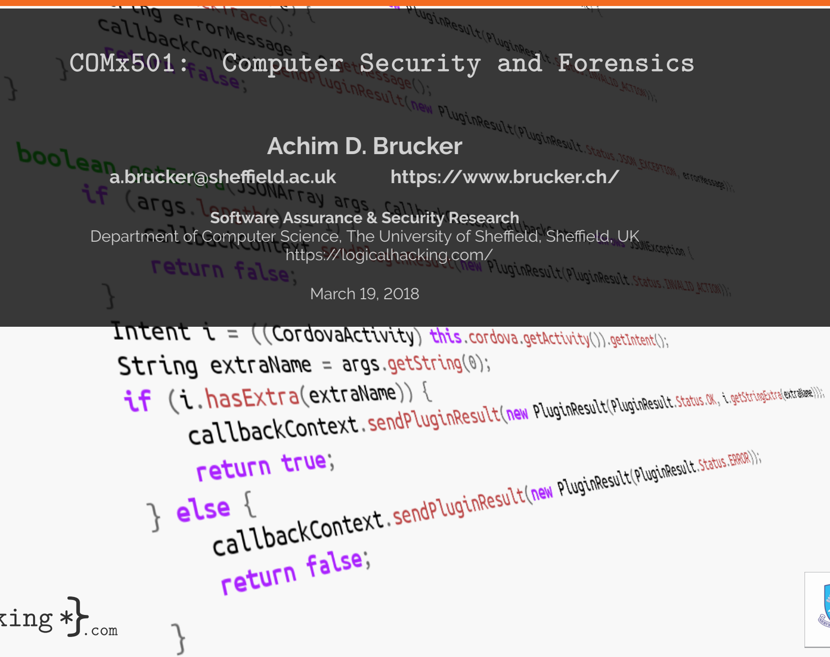
COMx501: Computer Security and Forensics

Achim D. Brucker

a.brucker@sheffield.ac.uk **https://www.brucker.ch/**

Software Assurance & Security Research
Department of Computer Science, The University of Sheffield, Sheffield, UK
https://logicalhacking.com/

March 19, 2018



```
Intent i = ((CordovaActivity) this.cordova.getActivity()).getIntent();
String extraName = args.getString(0);
if (i.hasExtra(extraName)) {
    callbackContext.sendPluginResult(new PluginResult(PluginResult.Status.OK, i.getStringExtra(extraName)));
    return true;
} else {
    callbackContext.sendPluginResult(new PluginResult(PluginResult.Status.ERROR));
    return false;
}
}
```

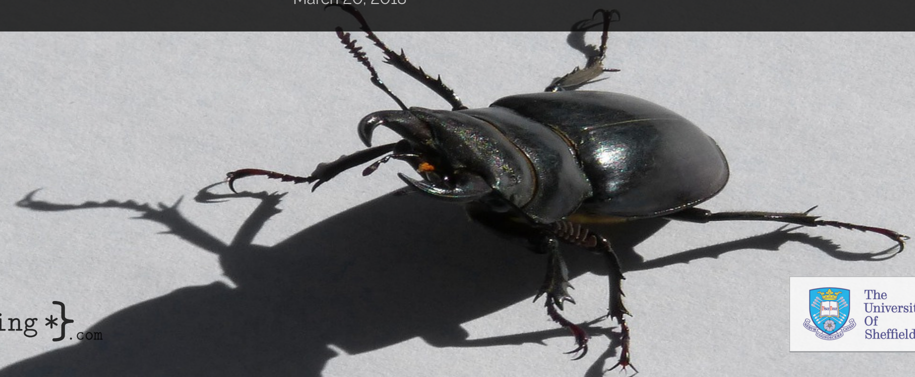
COMx501: Computer Security and Forensics
Part 10: Threat Modeling

Achim D. Brucker

a.brucker@sheffield.ac.uk **https://www.brucker.ch/**

Software Assurance & Security Research
Department of Computer Science, The University of Sheffield, Sheffield, UK
https://logicalhacking.com/

March 20, 2018



Outline

1 Introduction

2 Threat Modeling: Overview

3 Threat Modeling: Architectural Threats

4 Conclusion

5 Appendix

Motivation

Observation

Securing systems is expensive ⟷ Not all systems are equally rewarding for a ttackers

Let's consider you want to secure your bike:

- What do you want to protect
  - your old city bike
  - your new stylish bike
- Against whom
  - the casual attacker
  - targeted attack
- Available countermeasures
  - a cheap bike lock
  - an expensive lock
- Most vulnerable points
  - locking the front wheel only
  - locking the frame

Outline

1 Introduction

2 Threat Modeling: Overview

3 Threat Modeling: Architectural Threats

4 Conclusion

5 Appendix

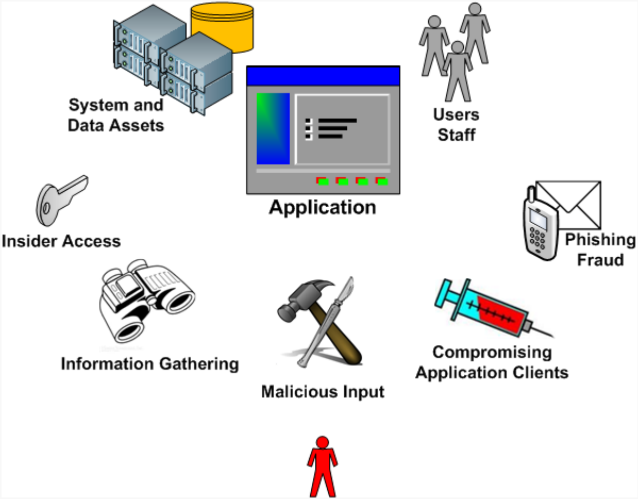
Threat Modeling as Part of a Secure Software Development Lifecycle

Threat modeling is a process, usually as part of the early steps of software development, by which potential threats are identified, enumerated, and prioritized.

Think like an attacker:

- Where are the high-value assets?
- Where am I most vulnerable to attack?
- What are the most relevant threats?
- Is there an attack vector that might go unnoticed?

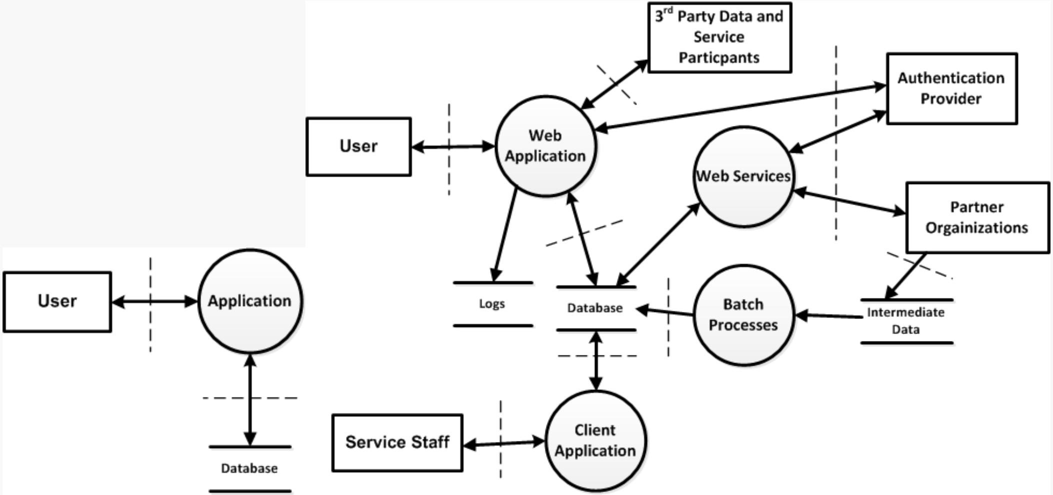
Understanding the Threats (and Risks)



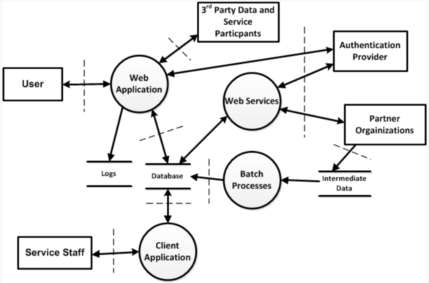
- High-Level attack vectors
  - Defeating a security mechanism
  - Abusing an application feature
  - Exploiting the insufficient security or poor implementation
- Remember, your application is part of a larger system

Understanding the Threats (and Risks)

A Simple Application Explodes Quickly Into Something Complex



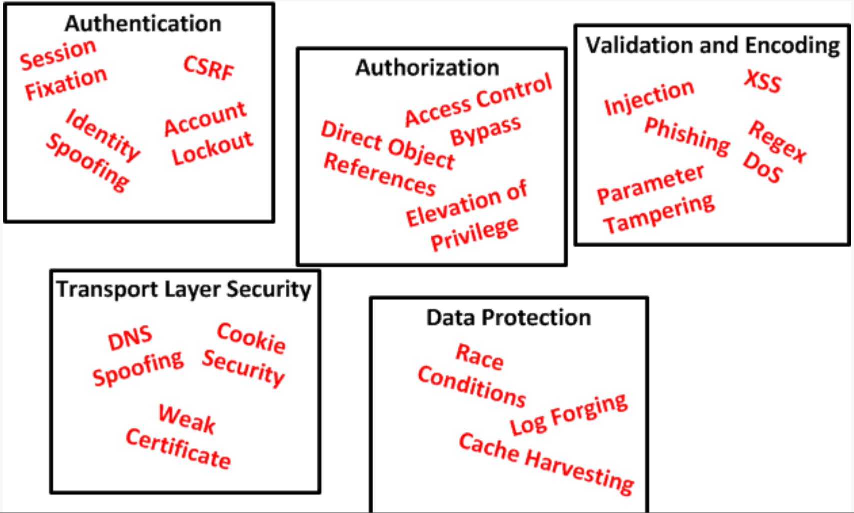
Understanding the Threats (and Risks)



- Try not to decide the scope of an architecture review or security assessment before thinking of the big picture
- The weakest point in a system may not be what you think
- With the right information on-hand, discovering vulnerabilities can be a simple matter of Q&A

Understanding the (Threats and) Risks

Poor Functional Security



Authentication

Session Fixation

CSRF

Identity Spoofing

Account Lockout

Authorization

Access Control Bypass

Direct Object References

Elevation of Privilege

Validation and Encoding

Injection

XSS

Phishing

Regex DoS

Parameter Tampering

Transport Layer Security

DNS Spoofing

Cookie Security

Weak Certificate

Data Protection

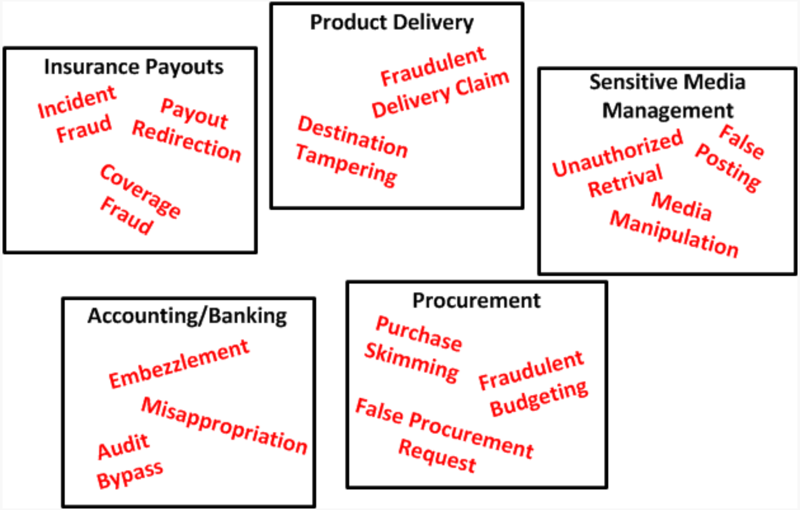
Race Conditions

Log Forging

Cache Harvesting

Understanding the (Threats and) Risks

Insecure Features



Insurance Payouts

- Incident Fraud
- Payout Redirection
- Coverage Fraud

Product Delivery

- Fraudulent Delivery Claim
- Destination Tampering

Sensitive Media Management

- Unauthorized Retrival
- False Posting
- Media Manipulation

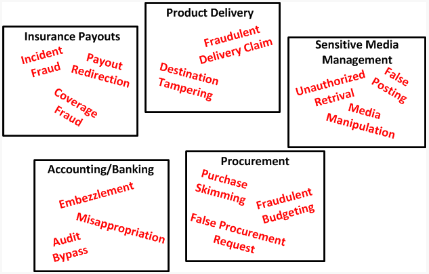
Accounting/Banking

- Embezzlement
- Misappropriation
- Audit Bypass

Procurement

- Purchase Skimming
- Fraudulent Budgeting
- False Procurement Request

Understanding the Threats (and Risks)



Insurance Payouts
Incident Fraud
Payout Redirection
Coverage Fraud

Product Delivery
Fraudulent Delivery Claim
Destination Tampering

Sensitive Media Management
Unauthorized Retrival
False Posting
Media Manipulation

Accounting/Banking
Embezzlement
Misappropriation
Audit Bypass

Procurement
Purchase Skimming
Fraudulent Budgeting
False Procurement Request

- Technology should not abstract business processes, but aid their efficient handling
- Application logic should not completely circumvent normal accountability
- You do not need to be proficient with a particular technology to evaluate a security solution
  - Is it adequate?
  - Do operational processes support it?
  - Is the solution an established, tested one or custom-made?

Threat Modeling: What we need

- Business: knowledge what the system should do, e.g., in terms of
  - scenarios, use cases
  - use cases
- Architectural: knowledge how information/data “flows” in the system, e.g., in terms of
  - block/component diagrams
  - data-flow diagrams
- Functional Security: how to defeat an attack, e.g., in terms of
  - planned security technologies/checks/processes
- Attackers Goals: Knowledge what an attacker might want to achieve, e.g., in terms of
  - Attack Trees
  - Threat Trees
- A team of experts, e.g.,
  - software architect
  - product owner
  - lead developer
  - security experts
  - domain experts
- A “structured” process to
  - ensure that no important aspects got forgotten
  - results are prioritized and documented

Outline

1 Introduction

2 Threat Modeling: Overview

3 Threat Modeling: Architectural Threats

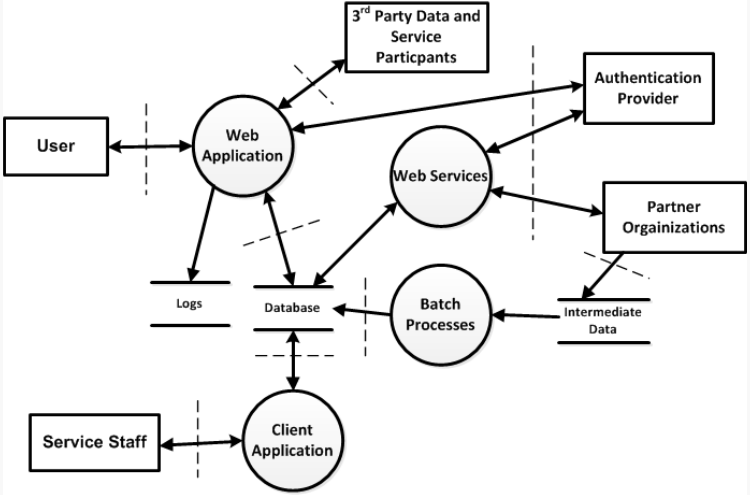
4 Conclusion

5 Appendix

Identifying Threats: Stride

- STRIDE is expansion of the common CIA threat types
  - Confidentiality
  - Integrity
  - Availability
- STRIDE:
  - **S**poofing Identity
  - **T**ampering with Data
  - **R**epudiation
  - **I**nformation Disclosure
  - **D**enial of Service
  - **E**levation of Privilege

Identifying Threats: An Example



Outline

1 Introduction

2 Threat Modeling: Overview

3 Threat Modeling: Architectural Threats

4 Conclusion

5 Appendix

Conclusion

- Threat modeling often a structured way of brain-storming
- Result should be document containing
  - the identified threats (with priorities!)
  - either acknowledging that a threat/risk is accepted
    ideally with justification why the risk is acceptable
  - or
  - the planned counter measures for an identified threat
    ideally with information how to test that the countermeasure is implemented correctly

Thank you for your attention!
Any questions or remarks?

Contact:

Dr. Achim D. Brucker
Department of Computer Science
University of Sheffield
Regent Court
211 Portobello St.
Sheffield S1 4DP, UK

a.brucker@sheffield.ac.uk
@adbrucker
https://de.linkedin.com/in/adbrucker/
https://www.brucker.ch/
https://logicalhacking.com/blog/

Bibliography I

Ross J. Anderson.

Security Engineering: A Guide to Building Dependable Distributed Systems.

John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 2001.

The complete book is available at: http://www.cl.cam.ac.uk/~rja14/book.html.

Document Classification and License Information

- This presentation is classified as *Student (COMx501 – 2017/18)*:
  Except where otherwise noted, this presentation is classified "*Student (COMx501 – 2017/18)*" and only available to students of the University of Sheffield that are registered to the module "COMx501: Computer Security and Forensics" in the academic year 2017/2018. Disclosure to third parties only after a confidentiality agreement has been signed.