A Non-Trivial Task of Introducing Architecture Risk Analysis into Software Development Process

Denis Pilipchuk
Global Product Security, Oracle

E-mail: denis.pilipchuk@oracle.com

AppSec EU 2014

OWASP

The Open Web Application Security Project

What is the Goal?

OWASP
The Open Web Application Security Project

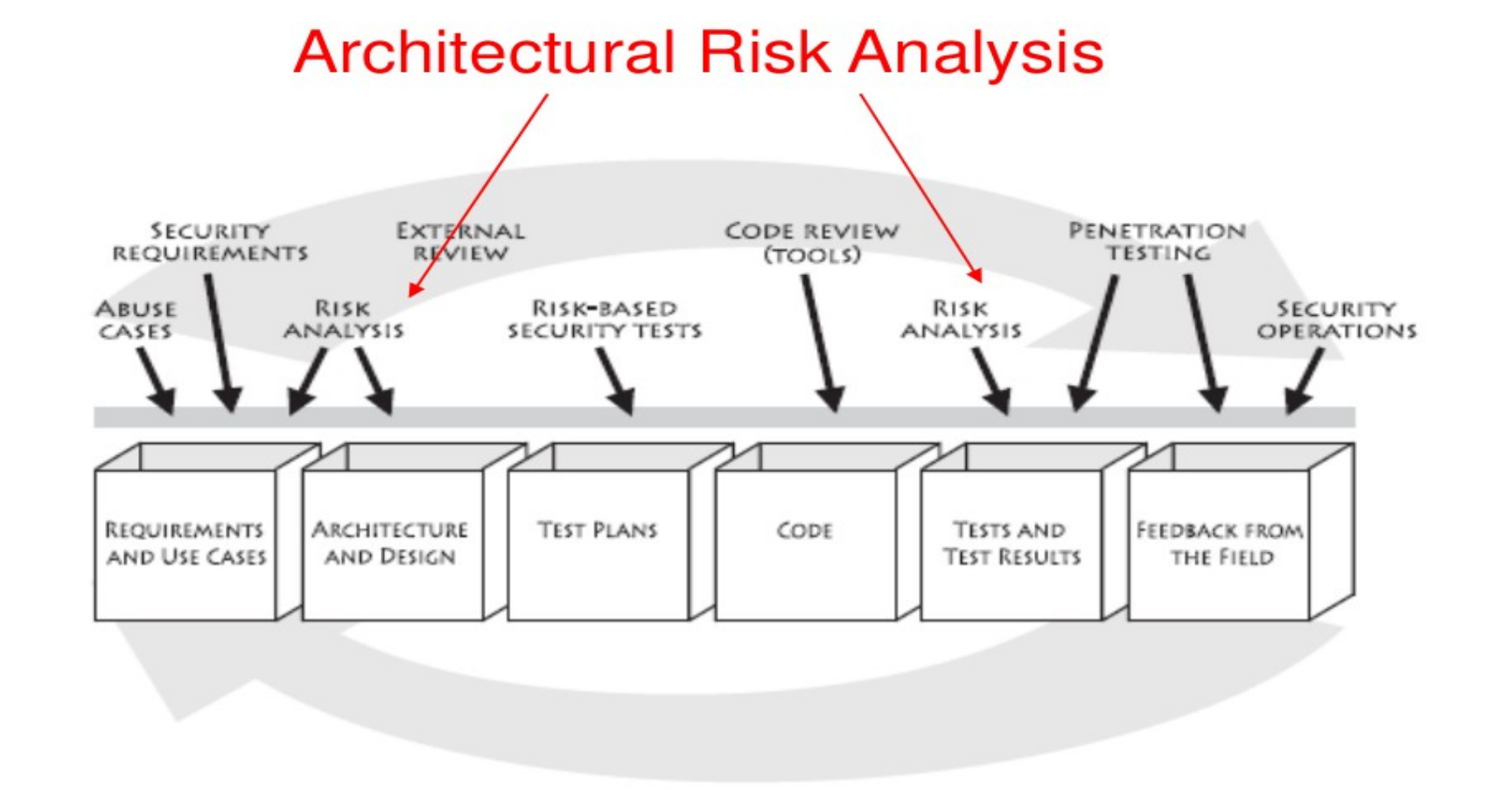
ARA should be practiced by development teams as an integral part of SDLC

"*The process is defined rigorously enough that people outside the SSG can be taught to carry it out.*"

BSIMM-V SSDL Touchpoint AA2.1

What is the Goal?

OWASP
The Open Web Application Security Project



Source: Cigital, "Software Security Touchpoint: Architectural Risk Analysis"

Reality Check

OWASP
The Open Web Application Security Project

- „Software Security Top 10 Surprises“, 2008 BSIMM analysis results

  "*Architecture analysis is just as hard as we thought, and maybe harder.*"

  "*Even well-known approaches to the architecture analysis problem, such as Microsoft's STRIDE model, turn out to be hard to turn into widespread practices that don't rely on specialists.*"

- Specialists = Software Security Group (SSG) or consultants

Reality Check

OWASP
The Open Web Application Security Project

BSIMM-V October 2013

- SSDL Touchpoints: Architecture Analysis (AA)

  AA3.1 "Have software architects lead review efforts." ~16%

- Intelligence: Attack Models (AM)

  AM2.1 "Build attack patterns and abuse cases tied to potential attackers." ~10%

  AM2.2 "Create technology-specific attack patterns." ~16%

SSDL Touchpoints	
Activity	**Observed**
[AA1.1]	56
[AA1.2]	35
[AA1.3]	24
[AA1.4]	42
[AA2.1]	10
[AA2.2]	8
[AA2.3]	20
[AA3.1]	11
[AA3.2]	4

Intelligence	
Activity	**Observed**
[AM1.1]	21
[AM1.2]	43
[AM1.3]	30
[AM1.4]	12
[AM1.5]	42
[AM1.6]	16
[AM2.1]	7
[AM2.2]	11
[AM3.1]	4
[AM3.2]	6

Who am I?

OWASP
The Open Web Application Security Project

- Developer, Architect
  - Security software – Netegrity, BEA
- Security Program Manager

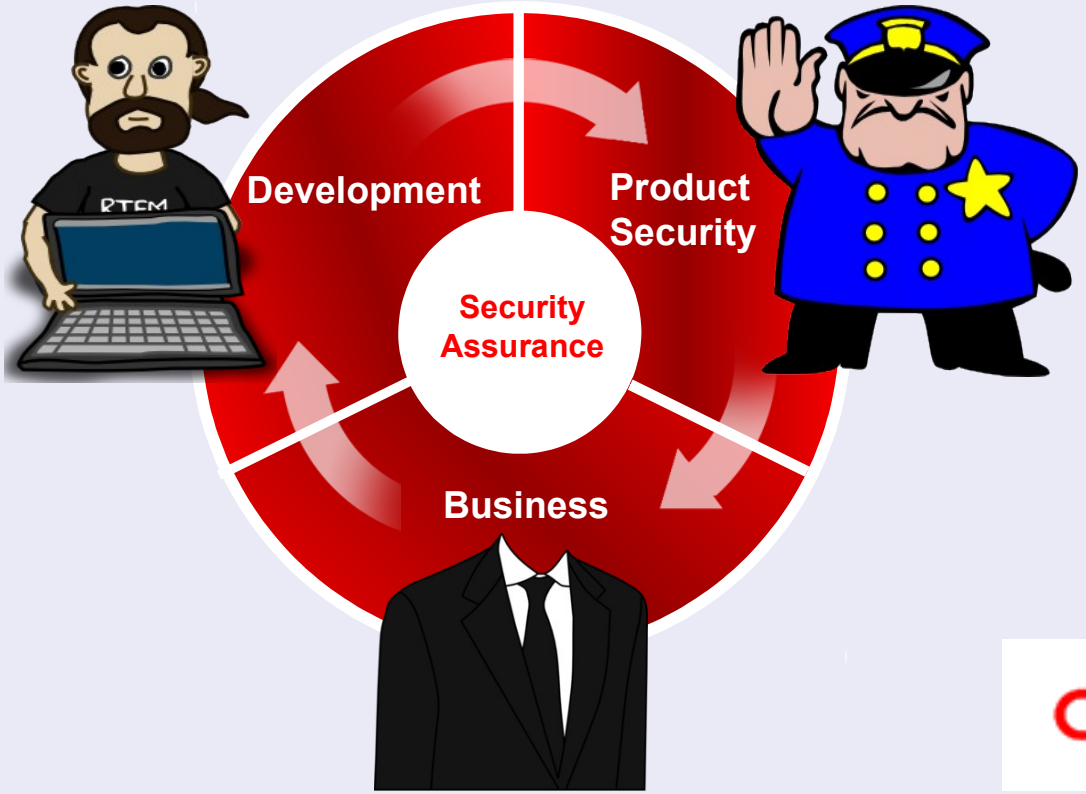
  Global Product Security team at Oracle
  - Security tools, threat modeling, risk analysis
  - Interact with senior management on security initiatives

ORACLE

Who am I?

OWASP
The Open Web Application Security Project

Or, representing it visually...



ORACLE

Agenda

OWASP
The Open Web Application Security Project

- Management viewpoint
- Development viewpoint
- SSG viewpoint
  - Analysis of the ARA landscape
- Where to go from here?

OWASP
The Open Web Application Security Project

The Management View...

ARA – Management View

OWASP
The Open Web Application Security Project

- Reactive security is „easier“
  - SWAT team approach is more visible

    „testing security in“ mentality

- Reported vulnerabilities have highest priorities
  - „Red Teams“ tend to dominate the discussion

ARA – Management View

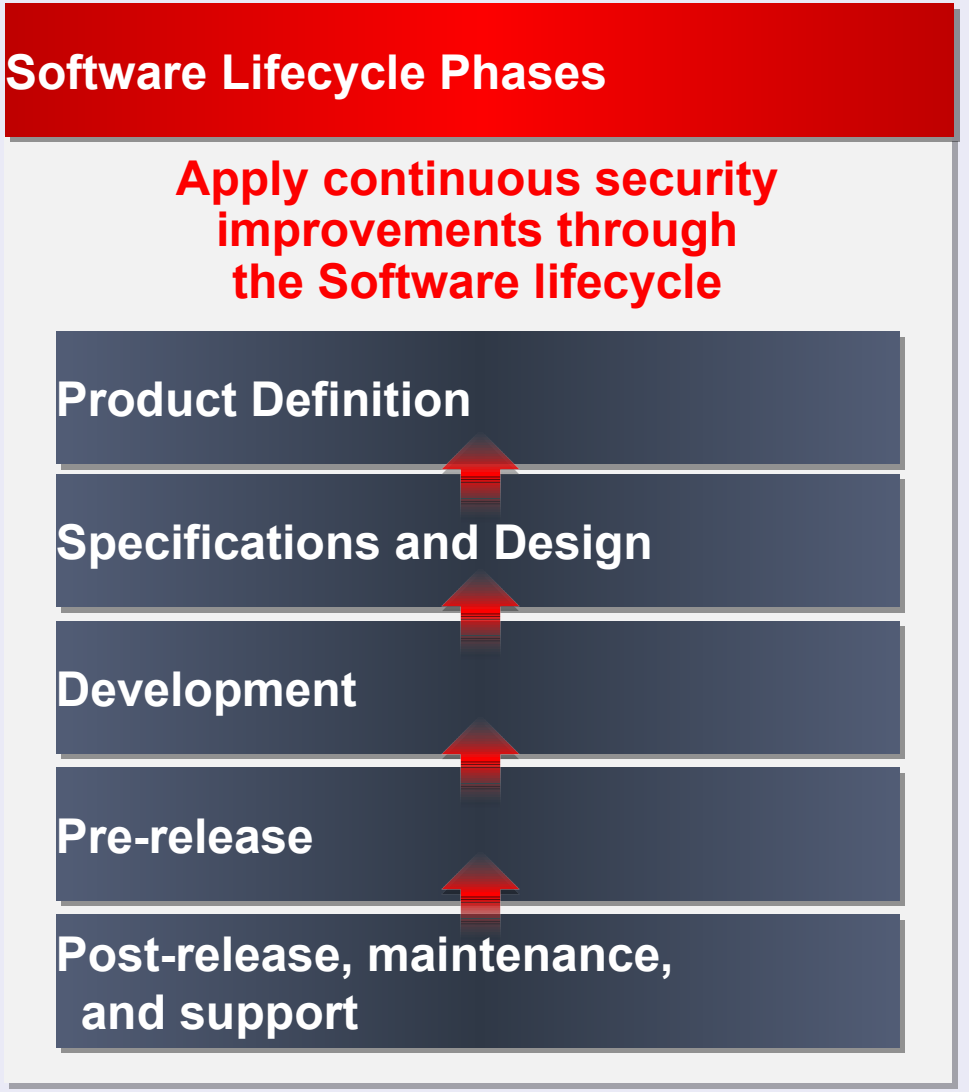
OWASP
The Open Web Application Security Project

- ARA ROI calculation is difficult (if at all possible!)
  - Costs:
    training, tooling, ongoing analysis costs
  - Returns: ***???***
- Possible short-term savings from outsourcing security analysis
  - Can outsource internally (SSG) or externally (consulting)

ARA – Management View

OWASP
The Open Web Application Security Project

- Mature SDLC is a must!
- ARA does not fit naturally into Agile processes



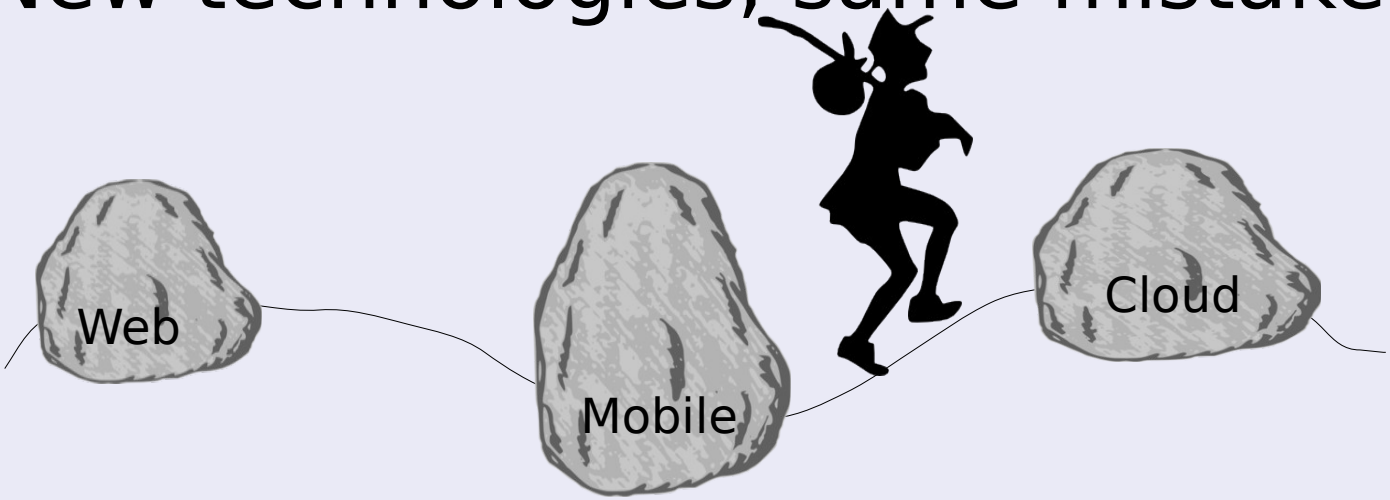
OWASP
The Open Web Application Security Project

The Development View...

ARA – Development View

OWASP
The Open Web Application Security Project

- Developers are interested in security, but lack specialized skills
  - Security considerations are not part of basic developers education
- New technologies, same mistakes



ARA – Development View

OWASP
The Open Web Application Security Project

„Attacker mentality“ goes against trained instincts...

VS

Build & verify

Attack & destroy

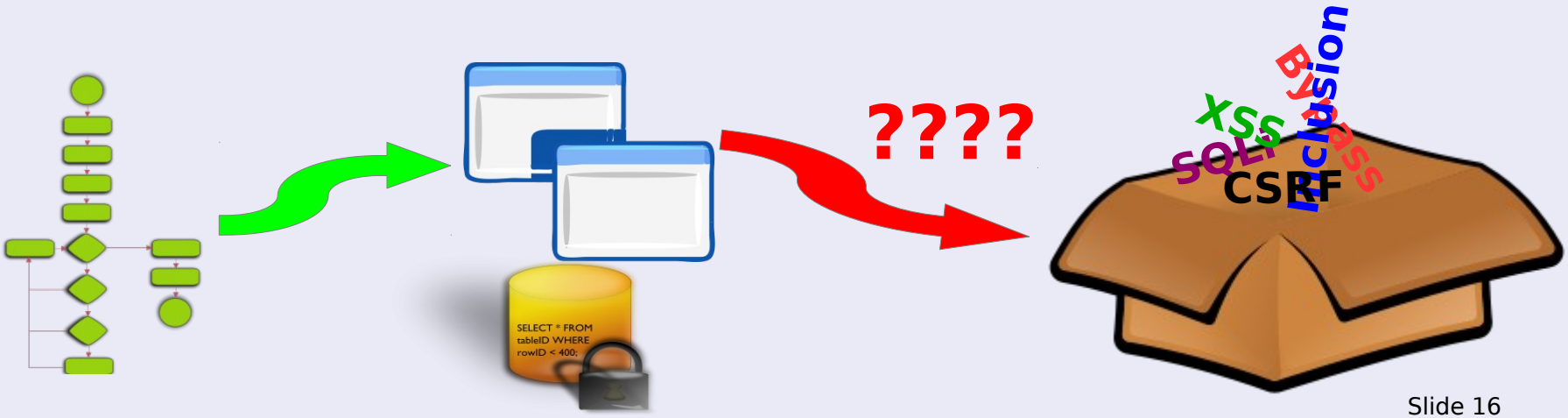
ARA – Development View

OWASP
The Open Web Application Security Project

- Terminology disconnects
  - Not everyday developers jargone:
    spoofing, repudiation, injection, …
- Logical disconnects

  Draw components, connections – OK

  Determine threats, attacks - NO



OWASP
The Open Web Application Security Project

The SSG View...

ARA – SSG View

OWASP
The Open Web Application Security Project

- ARA != Threat Modeling
  - Terminology confusion
- Risk measure is the key differentiator
  - Requires context… and lots of it

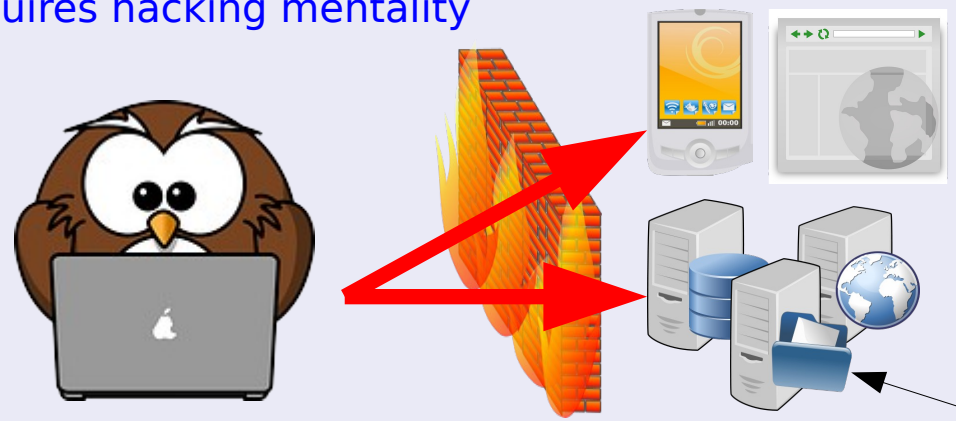
  Development teams can only measure technical risks!
- What can it discover?
  - Heartbleed, maybe … or maybe not?

OWASP
The Open Web Application Security Project

Challenges with methodologies...

Software-centric view
- Can measure only technical risks
- Typical for ISVs

Attacker-centric view
Requires hacking mentality



Asset-centric view
- Relies on deployment context
- Typical for IS/IT assessments

ARA – SSG View

OWASP
The Open Web Application Security Project

- Attack modeling is a crucial component of ARA process
  - Time-consuming, requires specialized skills
  - Need to know users, motivations, goals, etc
- Alternatives - tooling, attack knowledge bases

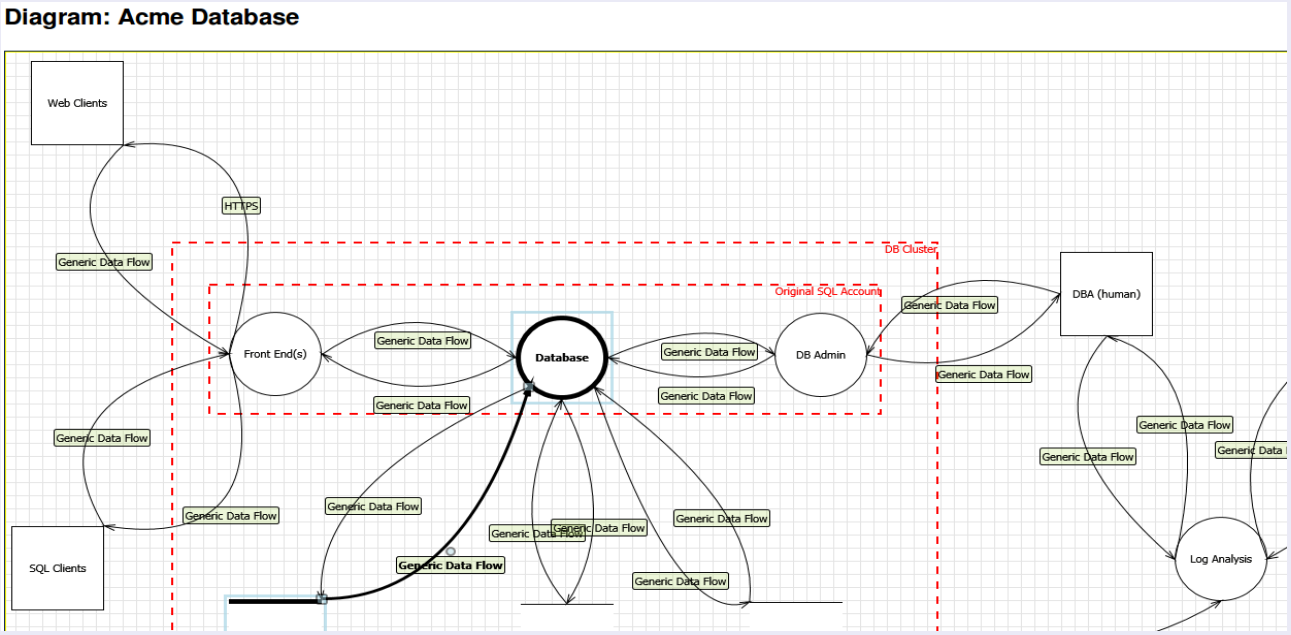
Goal: Read a message encrypted with PGP. (OR)
1. Decrypt the message itself. (OR)
 1.1. Break asymmetric encryption. (OR)
  1.1.1. Brute-force break asymmetric encryption. (OR)
  1.1.2. Mathematically break asymmetric encryption. (OR)
   1.1.2.1 Break RSA. (OR)
   1.1.2.2 Factor RSA modulus/calculate ElGamal discrete log.
  1.1.3. Cryptanalyze asymmetric encryption.
   1.1.3.1. General cryptanalysis of RSA/ElGamal. (OR)
   1.1.3.2. Exploiting weaknesses in RSA/ElGamal. (OR)
   1.1.3.3. Timing attacks on RSA/ElGamal.
 1.2. Break symmetric-key encryption.
  1.2.1. Brute-force break symmetric-key encryption. (OR)
  1.2.2. Cryptanalysis of symmetric-key encryption.
2. Determine symmetric key used to encrypt the message via other means.
 2.1. Fool sender into encrypting message using public key whose private key is known. (OR)
  2.1.1. Convince sender that a fake key (with known private key) is the key of the intended recipient.
  2.1.2. Convince sender to encrypt using more than one key—the real key of the recipient, and a key whose private key is known.
  2.1.3. Have the message encrypted with a different public key in the background, unbeknownst to the sender.
 2.2. Have the recipient sign the encrypted symmetric key. (OR)
 2.3. Monitor sender's computer memory. (OR)
 2.4. Monitor receiver's computer memory. (OR)
 2.5. Determine key from pseudorandom number generator. (OR)
  2.5.1. Determine state of randseed.bin when message was encrypted. (OR)
  2.5.2. Implant software (virus) that deterministically alters the state of randseed.bin. (OR)
  2.5.3. Implant software that directly affects the choice of symmetric key.
 2.6. Implant virus that exposes the symmetric key.
3. Get recipient to (help) decrypt message. (OR)
 3.1. Chosen ciphertext attack on symmetric key. (OR)
 3.2. Chosen ciphertext attack on public key. (OR)
 3.3. Send the original message to the recipient. (OR)
 3.4. Monitor outgoing mail of recipient. (OR)
 3.5. Spoof Reply-to: or From: field of original message. (OR)
 3.6. Read message after it has been decrypted by recipient.
  3.6.1. Copy message off user's hard drive or virtual memory. (OR)
  3.6.2. Copy message off backup tapes. (OR)
  3.6.3. Monitor network traffic. (OR)
  3.6.4. Use electromagnetic snooping techniques to read message as it is displayed on the screen. (OR)
  3.6.5. Recover message from printout.
4. Obtain private key of recipient.
 4.1. Factor RSA modulus/calculate ElGamal discrete log. (OR)
 4.2. Get private key from recipient's key ring. (OR)
  4.2.1. Obtain encrypted private key ring. (AND)
   4.2.1.1. Copy it from user's hard drive. (OR)
   4.2.1.2. Copy it from disk backups. (OR)
   4.2.1.3. Monitor network traffic. (OR)
   4.2.1.4. Implant virus/worm to expose copy of the encrypted private key.
  4.2.2. Decrypt private key.
   4.2.2.1. Break IDEA encryption. (OR)
    4.2.2.2.1.1. Brute-force break IDEA. (OR)
    4.2.2.2.1.2. Cryptanalysis of IDEA.
   4.2.2.2. Learn passphrase.
    4.2.2.2.1. Monitor keyboard when user types passphrase. (OR)
    4.2.2.2.2. Convince user to reveal passphrase. (OR)
    4.2.2.2.3. Use keyboard-logging software to record passphrase when typed by user. (OR)
    4.2.2.2.4. Guess passphrase.
 4.3. Monitor recipient's memory. (OR)
 4.4. Implant virus to expose private key.
 4.5. Generate insecure public/private key pair for recipient.

Source: https://www.schneier.com/paper-attacktrees-fig7.html

ARA – SSG View

OWASP
The Open Web Application Security Project

- Tooling support is limited
  - Situation is better on the IS/auditing side
- Example: MS ThreatModelingTool2014
  - Good at capturing data flows, components



ARA – SSG View

OWASP
The Open Web Application Security Project

But... Developers on their own can not translate generic threat entries into relevant attacks!!!

24. Potential Process Crash or Stop for Database [State: Not Started] [Priority: High]

Category: Denial of Service happens when the process or a datastore is not able to service incoming requests or perform up to spec.
Description: Database crashes, halts, stops or runs slowly; in all cases violating an availability metric.
Justification: <no mitigation provided>

25. Data Flow Generic Data Flow Is Potentially Interrupted [State: Not Started] [Priority: High]

Category: Denial of Service happens when the process or a datastore is not able to service incoming requests or perform up to spec.
Description: An external threat agent interrupts data flowing across a trust boundary in either direction.
Justification: <no mitigation provided>

26. Database May be Subject to Elevation of Privilege Using Remote Code Execution [State: Not Started] [Priority: High]

Category: A user subject gains increased capability or privilege by taking advantage of an implementation bug.
Description: Data may be able to remotely execute code for Database.
Justification: <no mitigation provided>

27. Elevation by Changing the Execution Flow in Database [State: Not Started] [Priority: High]

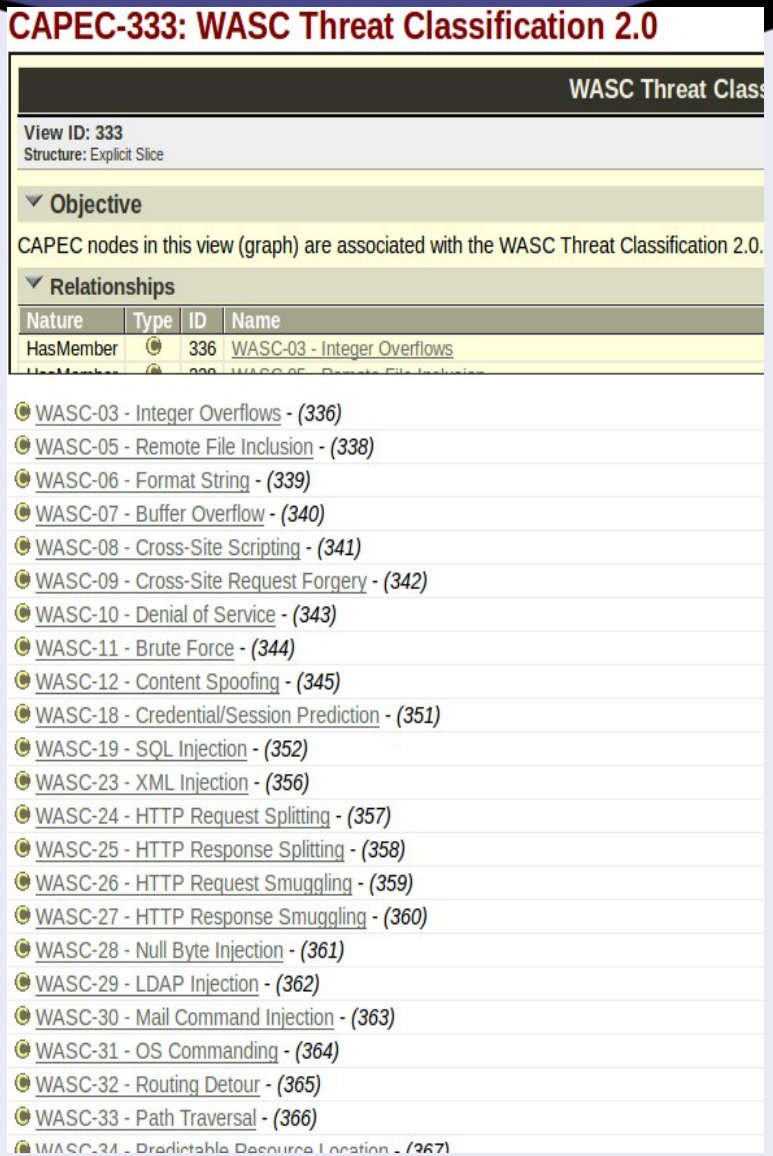
Category: A user subject gains increased capability or privilege by taking advantage of an implementation bug.
Description: An attacker may pass data into Database in order to change the flow of program execution within Database to the attacker's choosing.
Justification: <no mitigation provided>

ARA – SSG View

OWASP
The Open Web Application Security Project

Attack Knowledge Bases

- "The WASC Threat Classification"
  - Mostly for Web Apps
  - Good starting point
  - Not intended for automation
- MITRE CAPEC
  - ~800 entries
  - Maps to WASC, CWE, CVE



CAPEC-333: WASC Threat Classification 2.0

WASC Threat Class

View ID: 333
Structure: Explicit Slice

Objective

CAPEC nodes in this view (graph) are associated with the WASC Threat Classification 2.0.

Relationships

Nature	Type	ID	Name
HasMember		336	WASC-03 - Integer Overflows

- WASC-03 - Integer Overflows - *(336)*
- WASC-05 - Remote File Inclusion - *(338)*
- WASC-06 - Format String - *(339)*
- WASC-07 - Buffer Overflow - *(340)*
- WASC-08 - Cross-Site Scripting - *(341)*
- WASC-09 - Cross-Site Request Forgery - *(342)*
- WASC-10 - Denial of Service - *(343)*
- WASC-11 - Brute Force - *(344)*
- WASC-12 - Content Spoofing - *(345)*
- WASC-18 - Credential/Session Prediction - *(351)*
- WASC-19 - SQL Injection - *(352)*
- WASC-23 - XML Injection - *(356)*
- WASC-24 - HTTP Request Splitting - *(357)*
- WASC-25 - HTTP Response Splitting - *(358)*
- WASC-26 - HTTP Request Smuggling - *(359)*
- WASC-27 - HTTP Response Smuggling - *(360)*
- WASC-28 - Null Byte Injection - *(361)*
- WASC-29 - LDAP Injection - *(362)*
- WASC-30 - Mail Command Injection - *(363)*
- WASC-31 - OS Commanding - *(364)*
- WASC-32 - Routing Detour - *(365)*
- WASC-33 - Path Traversal - *(366)*
- WASC-34 - Predictable Resource Location - *(367)*

ARA – SSG View

OWASP
The Open Web Application Security Project

But CAPEC...

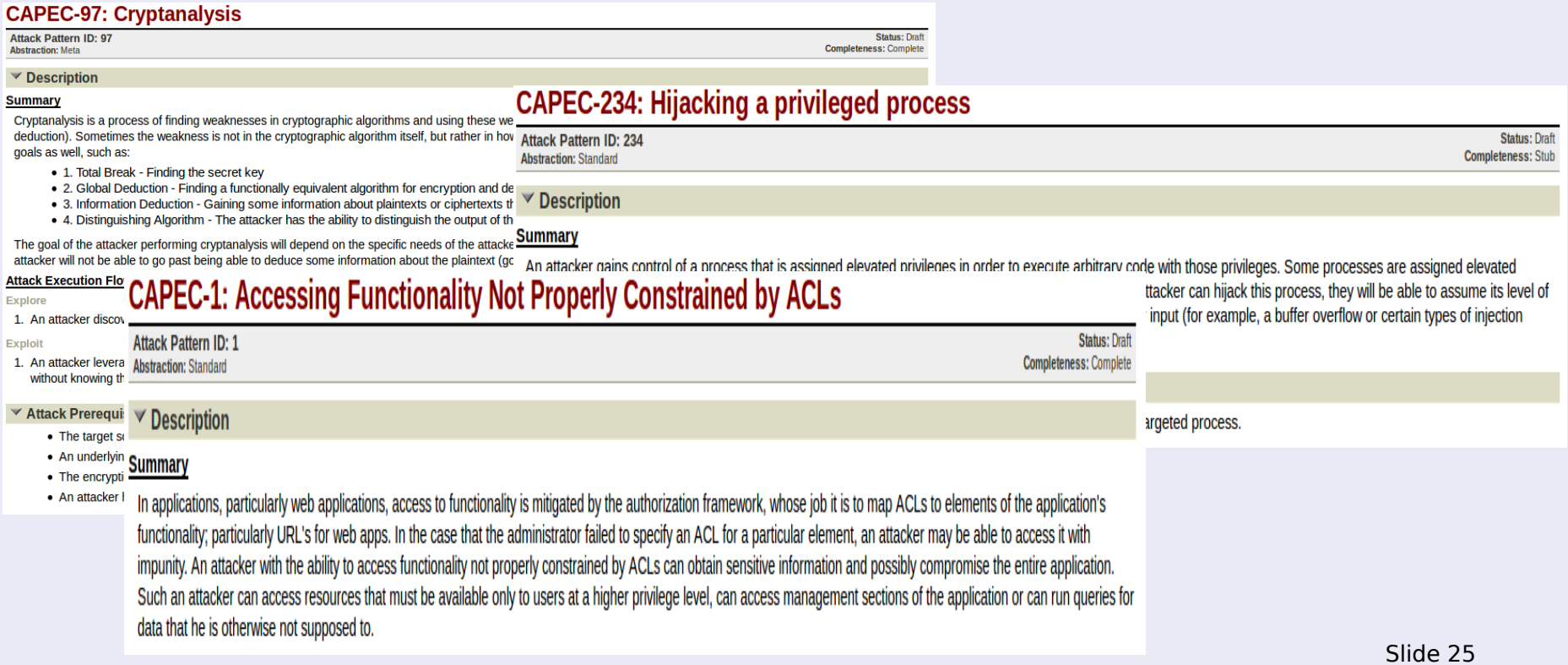
- Is impractical, merely a bag of ideas
  - Selection criteria are unclear
  - Lacks views by technology, job function, etc
- Many entries are simply inapplicable to dev teams!!!

- Gain Physical Access - *(436)*
  - Bypassing Physical Security - *(390)*
    - Bypassing Physical Locks - *(391)*
      - Lock Bumping - *(392)*
      - Lock Picking - *(393)*
      - Using a Snap Gun Lock to Force a Lock - *(394)*
    - Bypassing Electronic Locks and Access Controls - *(395)*
  - Physical Theft - *(507)*
- Alter System Components - *(526)*
- Manipulate System Users - *(527)*
  - Target Influence via Social Engineering - *(416)*
    - Target Influence via Perception of Reciprocation - *(417)*
    - Target Influence via Perception of Scarcity - *(420)*
    - Target Influence via Perception of Authority - *(421)*
    - Target Influence via Perception of Commitment and Consistency - *(422)*
    - Target Influence via Perception of Liking - *(423)*
    - Target Influence via Perception of Consensus or Social Proof - *(424)*
    - Target Influence via Framing - *(425)*
    - Target Influence via Manipulation of Incentives - *(426)*
    - Target Influence via Psychological Principles - *(427)*

ARA – SSG View

OWASP
The Open Web Application Security Project

- CAPEC entries content is very uneven
  - Many entries are stubs or of questionable value
- True even for some mappings from SANS Top 25

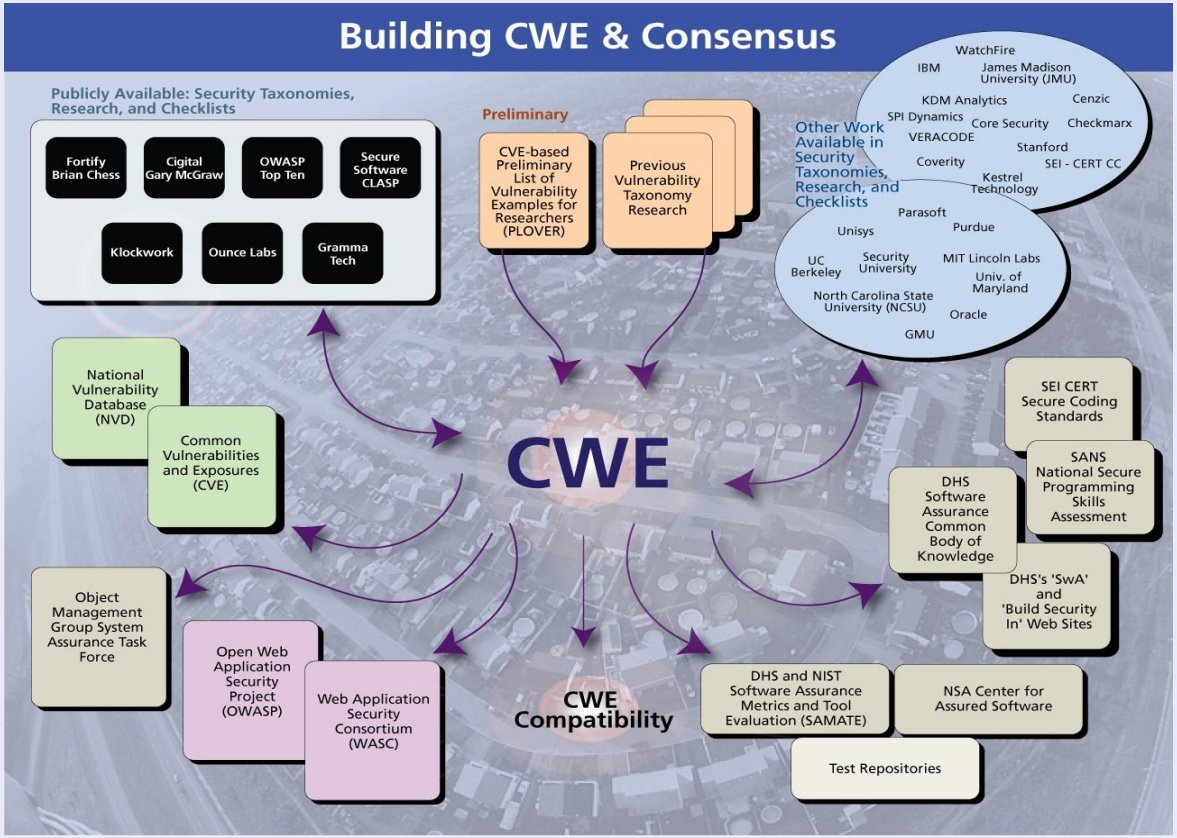


ARA – SSG View

OWASP
The Open Web Application Security Project

Contrast with CWE/CVE management...

- Well-defined structure
  - Suitable for automation
- Common terminology
- CWE ↔ CVE mapping



Source: http://cwe.mitre.org/index.html

OWASP
The Open Web Application Security Project

Where to go from here?

Next Steps

OWASP
The Open Web Application Security Project

- Expect the need for investment
  - No ready solutions
- Develop a custom threat/attack library
  - Can be industry- or technology-specific (BSIMM AM 2.2)
  - Problem – result will be non-standardized, likely - repeated work

Next Steps

OWASP
The Open Web Application Security Project

- Develop tooling to aid developers
  - Can use WASC/CAPEC as starting point, requires heavy polishing
- A wizard-style approach
  - Technology-specific questions using terminology familiar to developers
  - Filter by applicable component properties to make questions more targeted

Next Steps

OWASP
The Open Web Application Security Project

- Fix CAPEC!!!
  - Define target audience(s) and make the content suitable for them
  - Create criteria-based views
- Standards/industry organizations
  - Define commonly accepted threat/attack profiles (i.e. - "CWE/SANS 25" for attacks)
  - Can serve as basis for automation

OWASP
The Open Web Application Security Project

Thank you!

Questions?

E-mail: denis.pilipchuk@oracle.com