Secure Software Engineering

Winterterm 2023/24

Vulnerability Assessment

Dr. Christian Tiefenau

Florin Martius

UNIVERSITÄT BONN

Vulnerability of the day

Cache Poisoning

VOTD: Cache Poisoning – Actually an attack technique

Common Attack Pattern Enumeration and Classification

CAPEC-141: Cache Poisoning

Attack Pattern ID: 141
Abstraction: Standard

Status: Draft
Completeness: Complete

Presentation Filter: Basic

Summary

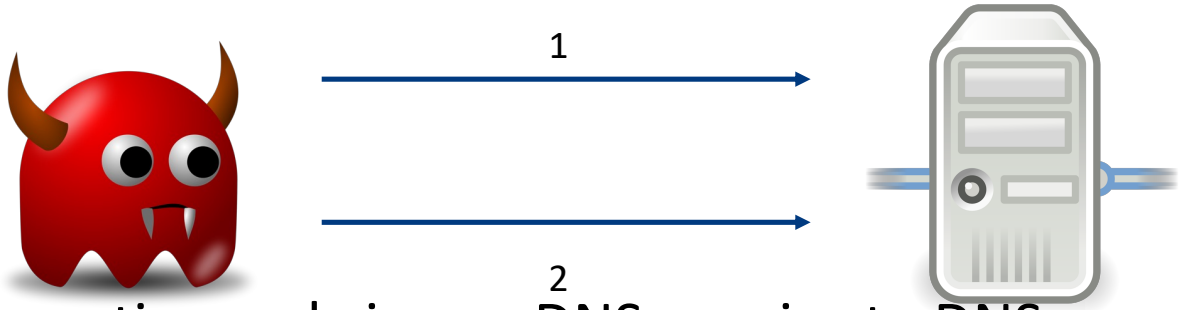
An attacker exploits the functionality of cache technologies to cause specific data to be cached that aids the attackers' objectives. This describes any attack whereby an attacker places incorrect or harmful material in cache. The targeted cache can be an application's cache (e.g. a web browser cache) or a public cache (e.g. a DNS or ARP cache). Until the cache is refreshed, most applications or clients will treat the corrupted cache value as valid. This can lead to a wide range of exploits including redirecting web browsers towards sites that install malware and repeatedly incorrect calculations based on the incorrect value.

Attack Prerequisites

- The attacker must be able to modify the value stored in a cache to match a desired value.
- The targeted application must not be able to detect the illicit modification of the cache and must trust the cache value in its calculations.

http://capec.mitre.org/data/definitions/141.html

Example: BIND DNS Cache Poisoning Attack



1. Attacker continuously issues DNS queries to DNS server
   - Server will delegate up, asking its parent for the record
2. Attacker also sends forged, incorrect responses to server
   - Problem: authenticity of response checked by a too small nonce
3. Once the correct nonce is (coincidentally) sent to the server, server will cache bogus record
4. Henceforth queries for that domain will yield the bogus response

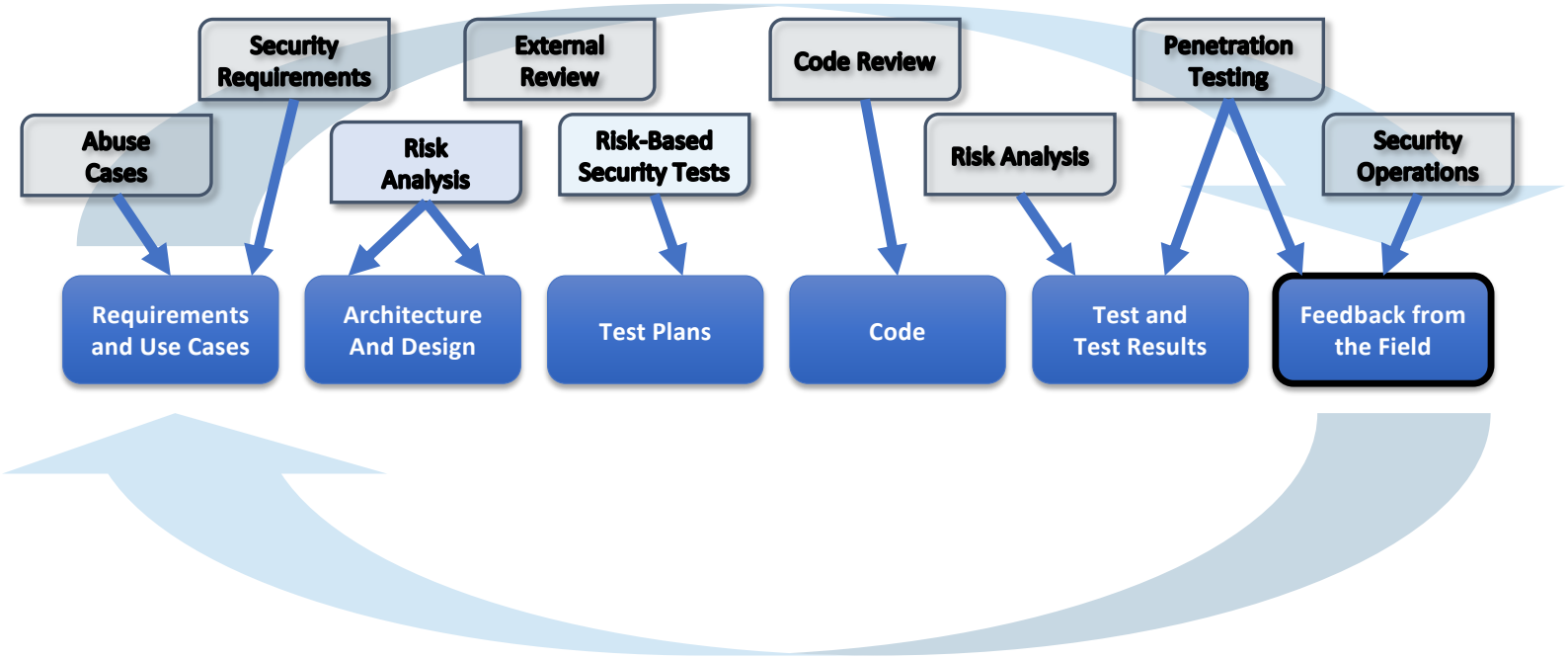
State-of-the-art solution: DNSSec

Avoiding cache poisoning

- If possible, don’t allow users to set their own cache expiration dates
- If possible, don’t allow users much control over caches to begin with
- As always, input validation helps, but it should be complemented with other countermeasures.

Vulnerability Assessment

Today's lecture



Vulnerabilities

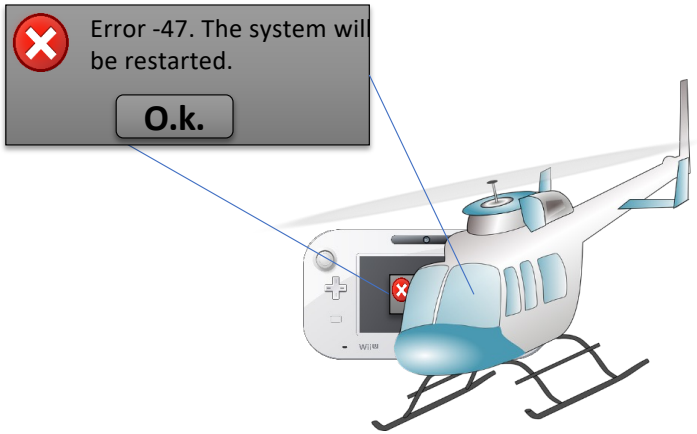
- No system is completely free of vulnerabilities
- What to do, if a vulnerability is found?
  - Investigate how "bad" this vulnerability is
    - In general
    - For your system / deployment

    Today
  - Fix /Patch the problem
  - Share with others

    Next Lecture
- **C**ommon **V**ulnerabilities and **E**xposures (~ 221k)
  - Managed and hosted by MITRE
  - https://cve.mitre.org/
  - We already referred to it every lecture
  - E.g., CVE-2011-1153

CVE-ID	
CVE-2011-1153	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information
Description	
Multiple format string vulnerabilities in phar_object.c in the phar extension in PHP 5.3.5 and earlier allow context-dependent attackers to obtain sensitive information from process memory, cause a denial of service (memory corruption), or possibly execute arbitrary code via format string specifiers in an argument to a class method, leading to an incorrect zend_throw_exception_ex call.	
References	

How Bad is Bad?



- We've seen many vulnerabilities
  - Many of them *can* do catastrophic things
  - Danger really "depends on the situation"

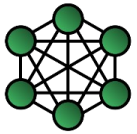
- Many, many situational factors, such as:


Asset exposed, and its relative importance

Remotely, or locally exploitable?



Expertise needed to exploit the vulnerability?

Affects all deployments?


Impact on CIA properties

How much traction did the problem have already?


Rating Vulnerabilities vs. Risk Management

UNIVERSITÄT BONN

- Similar to risk management, you have to rate found vulnerabilities
- How crucial is the vulnerability?

Risk Management

- Starts in early development phases, e.g., design
- Based on potential threats to the system
- Goal: Prevent (important) vulnerabilities

Vulnerability Assessment

- Only applicable for existing systems
- Applied to concrete vulnerabilities and (in the best case) corresponding exploits
- Goal: Fix and prevent further (important) vulnerabilities

If risk management is used/updated throughout the lifecycle, it can also support vulnerability assessment.

Rating Vulnerabilities

- We need a method to rate discovered vulnerabilities
  - Should take all essential factors into account
  - Should be repeatable and deterministic (to a certain degree)
  - Should result in comparable results (order of importance)
  - Should be approved by experts / industry

 Common **V**ulnerability **S**coring **S**ystem (CVSS)

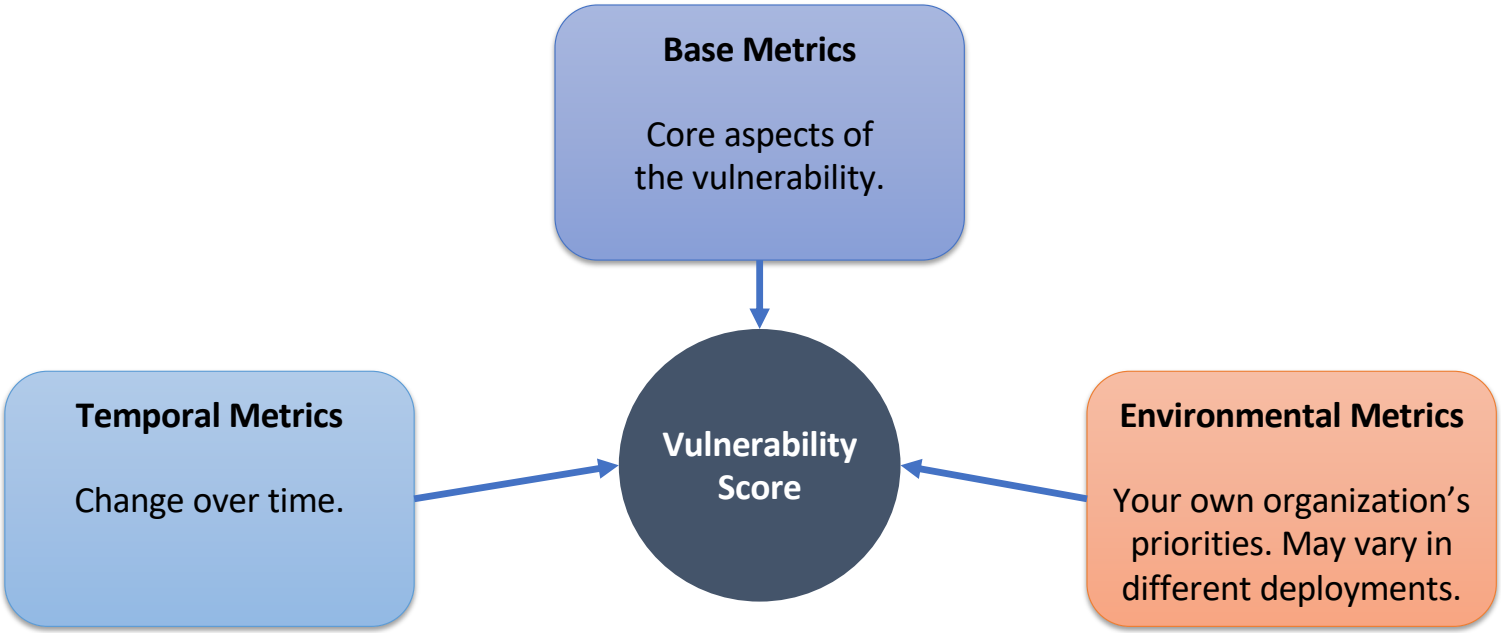
CVSS Common Vulnerability Scoring System

- An open scoring system from FIRST
  - FIRST: Forum for Incident Response & Security Teams
  - http://www.first.org/cvss
  - A group of researchers & practitioners

- Adopted by NIST
- CVSS added in CVE descriptions
  - NVD (NIST) provides CVSS scores for all CVE
- Latest version: v3.1 (2019)
- Mostly applied in industry: v2

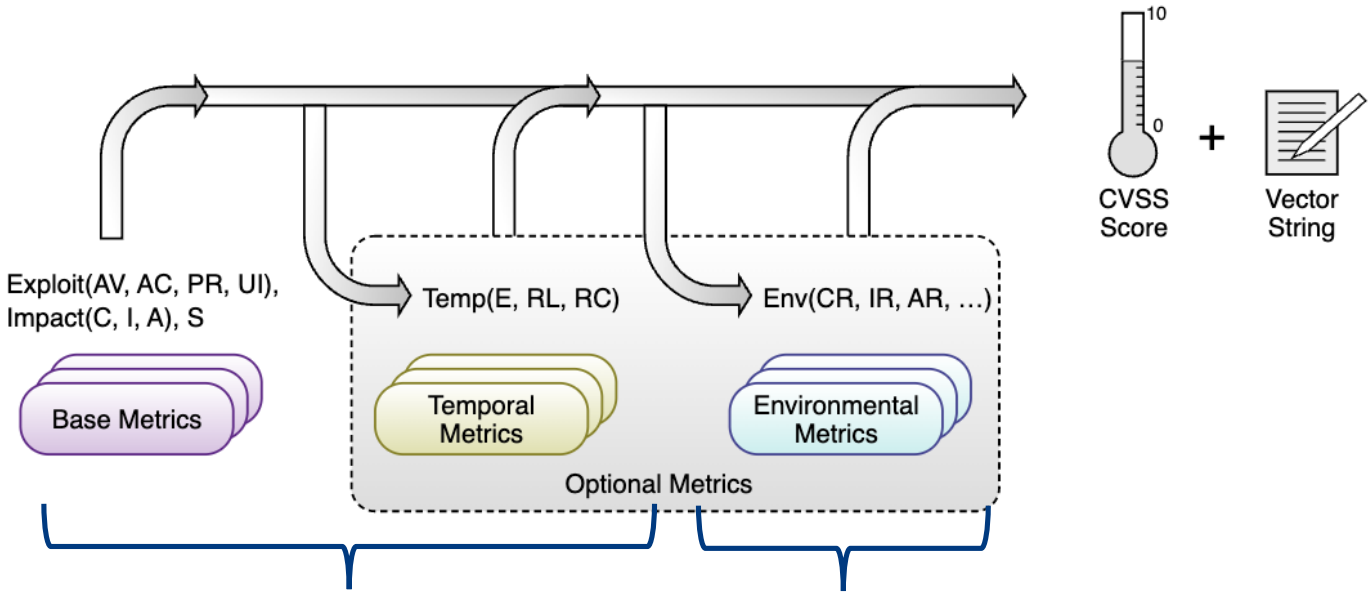
- Provide a set of metrics, and corresponding values and weighting functions

CVSS Metric Groups



https://www.first.org/cvss/specification-document

CVSS Metric Groups



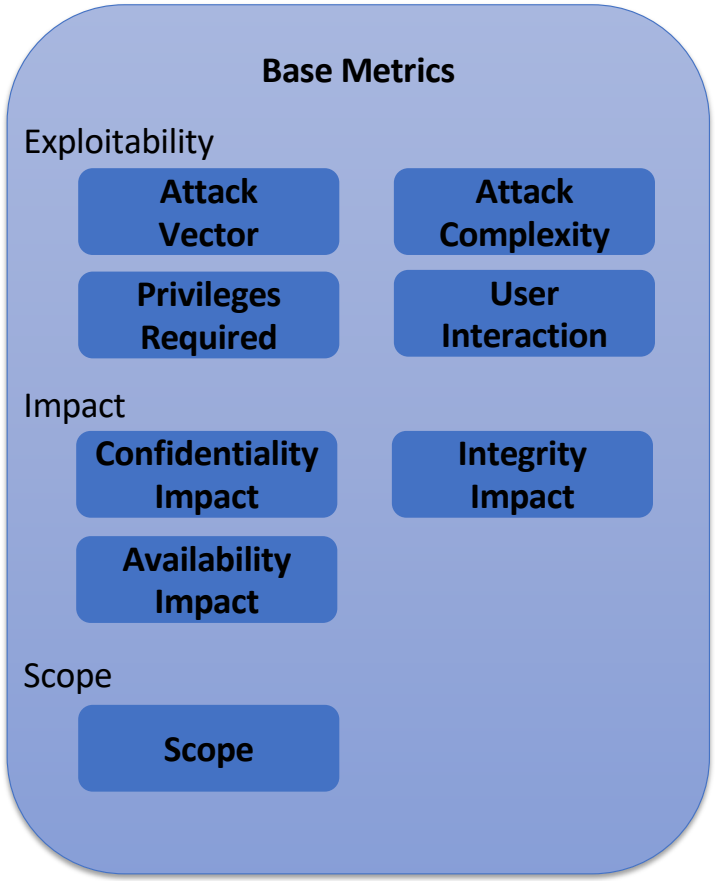
Typically done by vulnerability bulletin analysts, security product vendors, or application vendors

To be done by “user” organisation

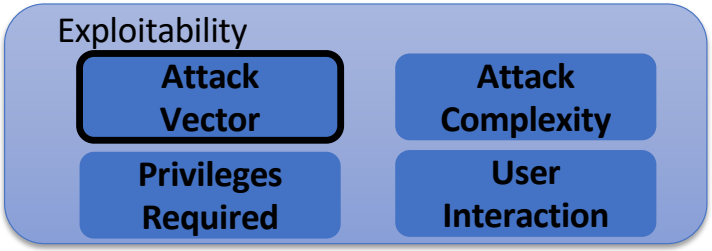
https://www.first.org/cvss/specification-document

Base Metric Group

- Exploitability metrics
  - Characteristics of *how* a given thing is vulnerable
  - Scored relative to vulnerable component
- Impact metrics
  - Represent the consequence to the thing that suffers the impact
- Scope
  - Which parts of the system are affected?



Base Attack Vector (AV)

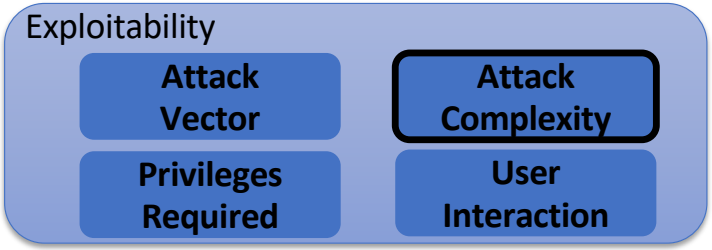


- Through what entry gates can an attacker exploit the vulnerability?
- Metric Value:
  - (P) Physical
  - (L) Local only
  - (A) Adjacent network (e.g. wi-fi, local IP subnet)
  - (N) Network: fully remotely exploitable

- More than one level affected? Go with the worse one
- Client that opens stuff from an untrusted internet source? Go with Network (e.g. zip utility with a buffer overflow)

1. XSS in a webapp? **(N)**
2. No HTTPS on banking website? **(A)**

Base Attack Complexity (AC)

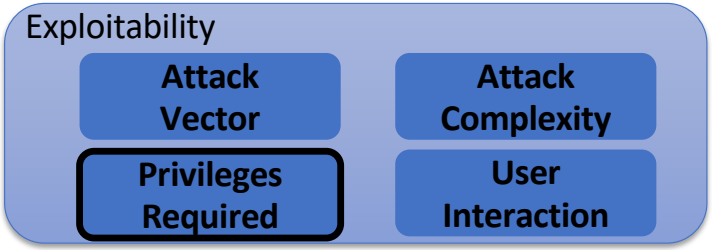


- How complex is the vulnerability to exploit?
  - One step? e.g. SQL Injection
  - Multiple steps?
    e.g. convince an email user to download a sketchy attachment

- Metric value
  - (H) High: Specialized access conditions
    - e.g. overcoming advanced exploit mitigation techniques
    - e.g. knowledge about the environment necessary
    - e.g. man in the middle attack

  - (L) Low: no specialized conditions
    - e.g. default configuration
    - e.g. requires little skill to perform
    - Attacker can expect repeatable success
- Note: Low complexity is bad

Base Privileges Required (PR)



Behavioural Security Group

UNIVERSITÄT BONN

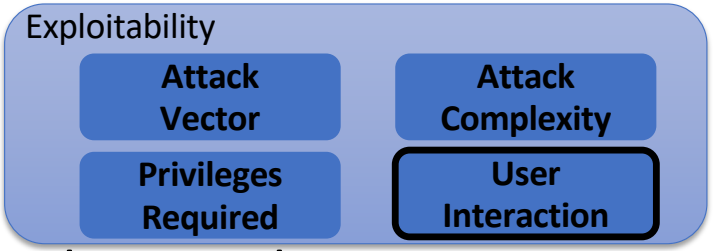
- Level of privileges needed for exploit?

- Metric value
  - (H) privileges that provide significant control
    - (e.g. administrative)
  - (L) privileges that provide basic user capabilities
  - (N) No authorization needed

- In an authentication system itself? (e.g. Kerberos) … go with (N)!

1. Path traversal in photo upload for a Twitter client? **(L)**
2. Insecure PRNG for session IDs? **(N)**

Base User Interaction (UI)

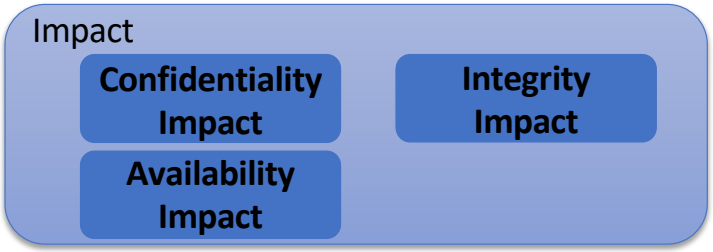


- Do we need a user, other than the attacker, participating in the exploit?

- Metric value
  - (N): can be exploited without interaction from any user
  - (R): requires a user to take action
    - E.g. exploit may only be possible during the installation of an application by a system administrator.

1. Reflected XSS?	**(R)** – must click on a link
2. CSRF?	**(R)** – need victim to create the http request & be logged in

Base CIA Impact



- Any impact on
  - confidentiality, integrity, and/or availability?
  - These are three separate metrics

- Metric Value (for each metric)
  - (N) None
  - (L) Low
    - e.g. disclosing a few database tables
    - e.g. temporary DoS
  - (H) High
    - e.g. reading arbitrary memory locations is High Disclosure
    - e.g. full bypass of plug-in sandbox is High Integrity
    - e.g. root-level access? High on all three metrics

Hardcoded root credentials in blogging software?

C = High | I = High | A = High (Multiple Scenarios possible)

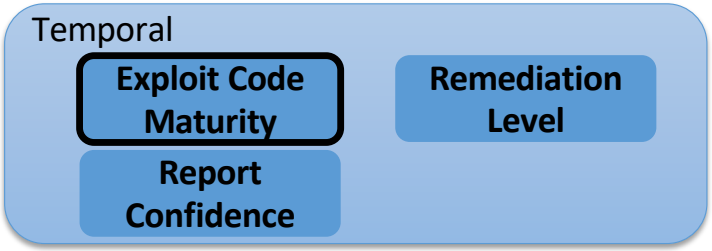
Base: Scope

- The ability for a vulnerability in one software component to impact resources beyond its means, or privileges.

- Metric Value
  - (U): Unchanged
    - The vulnerable component and the impacted component are the same
    - Or both are managed by the same security authority
  - (C): Changed
    - The vulnerable component and the impacted component are different
    - And both are managed by the same security authority

1. Vulnerability in a Linux VM that compromises the host OS? **(C)**
2. Using crafted office file to cause a DoS in office suite? **(U)**

Temporal Exploitability (E)

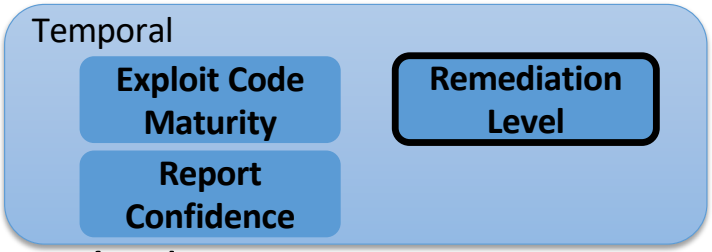


- Is an exploit publicly known?

- Metric Value
  - (U) Unproven, entirely theoretical exploit
  - (POC) Proof-of-concept exists out there, no known maliciously used exploits
  - (F) Functional exploit is available
  - (H) Functional Exploit is widely disseminated
  - (X) Not defined (skip this part of the metric)

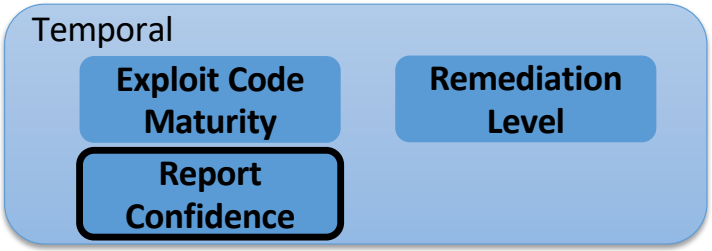
- Notes
  - This usually elevates quickly!
  - Many white-hats will write exploits to make this score go up so that it becomes fixed

Temporal Remediation Level



- What is the *level of remediation (RL)*?
  - How has the vendor reacted?
  - Levels
    - (O) Official Fix is available
    - (TF) Temporary fix is available
    - (W) Workaround is available
      - Unofficial, non-vendor patches
      - Temporary change in configuration
    - (U) Nothing is released yet
    - (X) Not defined

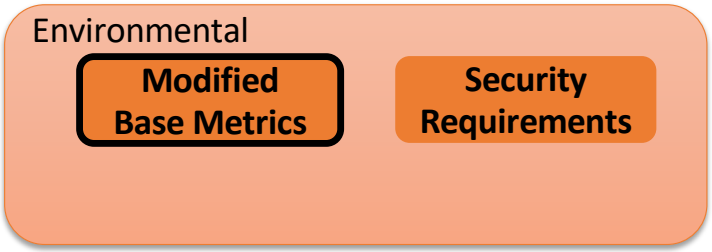
Temporal Report Confidence



- What is the *report confidence (RC)*?
  - (U) Unconfirmed by the source, or there are multiple conflicting reports
  - (R) Significant details are published e.g. proof-of-concept
  - (C) Confirmed by the source
  - (ND) Not defined

Environmental

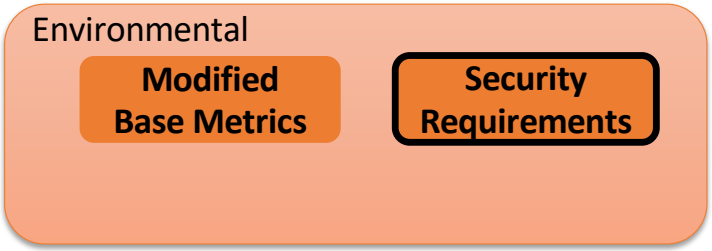
Modified Base Metrics



- Modified Base Metrics
  - Metrics according to modifications that exist within the environment
    - Loss of life, physical assets, productivity
  - Levels:
    - None
    - Low
    - Medium
    - High
    - Not defined

Environmental

Security Requirements



- Security Requirements
  - Confidentiality Requirement (CR)
  - Integrity Requirement (IR)
  - Availability Requirement (AR)
- Reweighting the modified CIA impact metrics
- Metric Value: Loss of CIA has…
    - (X): Not defined. It will not influence the score.
    - (H): A catastrophic adverse effect
    - (M): A serious adverse effect
    - (L): A limited adverse effect

Scoring

Metric Group	Metric Name (and Abbreviated Form)	Possible Values	Mandatory?
Base	Attack Vector (AV)	[N,A,L,P]	Yes
	Attack Complexity (AC)	[L,H]	Yes
	Privileges Required (PR)	[N,L,H]	Yes
	User Interaction (UI)	[N,R]	Yes
	Scope (S)	[U,C]	Yes
	Confidentiality (C)	[H,L,N]	Yes
	Integrity (I)	[H,L,N]	Yes
	Availability (A)	[H,L,N]	Yes
Temporal	Exploit Code Maturity (E)	[X,H,F,P,U]	No
	Remediation Level (RL)	[X,U,W,T,O]	No
	Report Confidence (RC)	[X,C,R,U]	No
Environmental	Confidentiality Requirement (CR)	[X,H,M,L]	No
	Integrity Requirement (IR)	[X,H,M,L]	No
	Availability Requirement (AR)	[X,H,M,L]	No
	Modified Attack Vector (MAV)	[X,N,A,L,P]	No
	Modified Attack Complexity (MAC)	[X,L,H]	No
	Modified Privileges Required (MPR)	[X,N,L,H]	No
	Modified User Interaction (MUI)	[X,N,R]	No
	Modified Scope (MS)	[X,U,C]	No
	Modified Confidentiality (MC)	[X,N,L,H]	No
	Modified Integrity (MI)	[X,N,L,H]	No
	Modified Availability (MA)	[X,N,L,H]	No

https://www.first.org/cvss/specification-document

Scoring – Base Metrics Equations

Attack Vector / Modified Attack Vector	Network
	Adjacent
	Local
	Physical
Attack Complexity / Modified Attack Complexity	Low
	High
Privileges Required / Modified Privileges Required	None
	Low
	High
User Interaction / Modified User Interaction	None
	Required
Confidentiality / Integrity / Availability / Modified Confidentiality / Modified Integrity / Modified Availability	High
	Low
	None

...

Not relevant in the exam

Scoring – Base Metrics Equations

ISS =	**1 - [(1 - Confidentiality) × (1 - Integrity) × (1 - Availability)]**
Impact =	
If Scope is Unchanged	6.42 × ISS
If Scope is Changed	$7.52 \times (ISS - 0.029) - 3.25 \times (ISS - 0.02)^{15}$
Exploitability =	8.22 × AttackVector × AttackComplexity ×
	PrivilegesRequired × UserInteraction
BaseScore =	
If Impact \<= 0	0, *else*
If Scope is Unchanged	Roundup (Minimum [(Impact + Exploitability), 10])
If Scope is Changed	Roundup (Minimum [1.08 × (Impact + Exploitability), 10])

Scoring – Temporal + Environmental Metrics Equations

TemporalScore = Roundup (BaseScore × ExploitCodeMaturity × RemediationLevel × ReportConfidence)

MISS = Minimum (1 - [(1 - ConfidentialityRequirement × ModifiedConfidentiality) × (1 - IntegrityRequirement × ModifiedIntegrity) × (1 - AvailabilityRequirement × ModifiedAvailability)], 0.915)

ModifiedImpact =	
If ModifiedScope is Unchanged	6.42 × MISS
If ModifiedScope is Changed	$7.52 \times (MISS - 0.029) - 3.25 \times (MISS \times 0.9731 - 0.02)^{13}$
ModifiedExploitability =	8.22 × ModifiedAttackVector × ModifiedAttackComplexity × ModifiedPrivilegesRequired × ModifiedUserInteraction

Note that the exponent at the end of the ModifiedImpact sub-formula is 13, which differs from CVSS v3.0. See the User Guide for more details of this change.

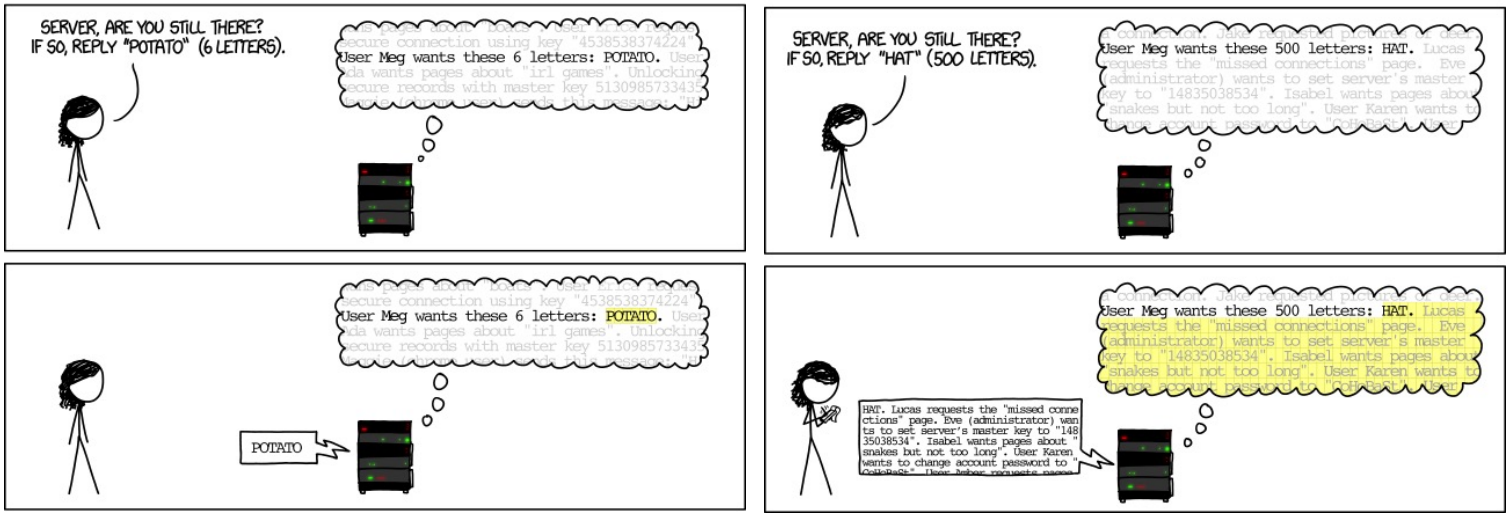
EnvironmentalScore =

If ModifiedImpact \<= 0	0, *else*
If ModifiedScope is	Roundup (Roundup [Minimum ([ModifiedImpact + ModifiedExploitability], 10)] × ExploitCodeMaturity × RemediationLevel × ReportConfidence)
Unchanged	
If ModifiedScope is	Roundup (Roundup [Minimum (1.08 × [ModifiedImpact + ModifiedExploitability], 10)] × ExploitCodeMaturity × RemediationLevel × ReportConfidence)
Changed	

Scoring Tips

- Ignore interactions with other vulnerabilities, score each individually
- Assume the most common or default configuration of the server
- Score the greatest exploitation impact, if there are many

Example: CVE-2014-0160 (Heartbleed Bug)



Example: CVE-2014-0160 (Heartbleed Bug)

UNIVERSITÄT BONN

- CVSS v3.1 Base Score: **7.5**

Metric	Value	Comments
Attack Vector		
Attack Complexity		
Privileges Required		
User Interaction		
Scope		
Confidentiality Impact		
Integrity Impact		
Availability Impact		

https://www.first.org/cvss/examples

Example: CVE-2014-0160 (Heartbleed Bug)

- CVSS v3.1 Base Score: **7.5**

- In CVSS v2.0, Heartbleed had only a score of **5.0**!

Alternative: CWSS

- Common ***WEAKNESS*** Scoring System
  - Relatively recent (~2010) response to CVSS
  - More detailed, but not as widely-adopted
  - http://cwe.mitre.org/cwss/
  - Categories: Base, Attack Surface, Environmental

- Base Finding Metric Group
  - 5 metrics in this group
  - e.g. Acquired Privilege
    - *User-level access acquired. Admin?*
  - e.g. Acquired Privilege Layer
    - *Access to Network, App, entire Enterprise*
  - e.g. Internal Control Effectiveness
    - *Would our internal detection measures have been effective? Would we have known this was exploited?*

Alternative: CWSS (cont.)

- Attack Surface Metric Group
  - 7 metrics in this group
  - e.g. Required Privilege AND Required Privilege Layer
    - *How much authentication was needed?*
  - e.g. Level of Interaction
    How much social engineering is required?

- Environmental Impact Group
  - 6 metrics in this group
  - e.g. Business Impact
  - e.g. Likelihood of Discovery
  - e.g. Likelihood of Exploit
  - e.g. Remediation Effort
    - *Is this a really difficult fix? Should we be worried about this coming up again or being incorrectly fixed?*

Further Reading

- Official Specification
  - https://www.first.org/cvss/specification-document
- User Guide
  - https://www.first.org/cvss/user-guide
- Examples
  - https://www.first.org/cvss/examples
- Calculator
  - https://www.first.org/cvss/calculator/3.1
  - https://www.first.org/cvss/use-design