Session 01

Introduction to Systems Security

Security of Information Systems (SIS)

Computer Science and Engineering Department

October 4, 2023

iOS Jailbreak: unc0ver

`https://unc0ver.dev/`

History of OWASP Top 10

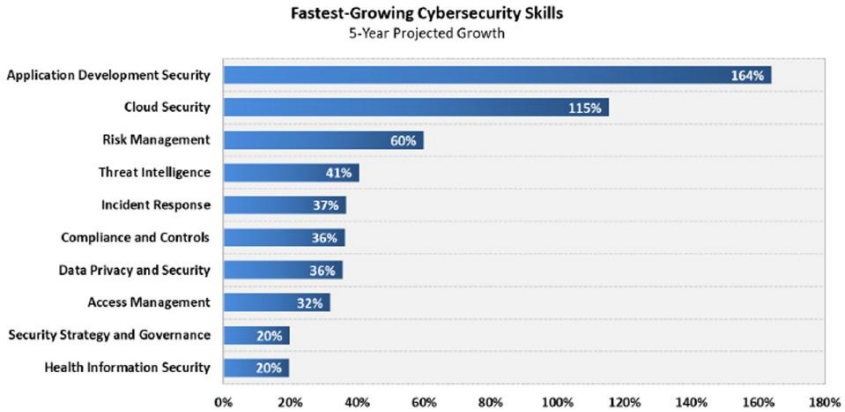
`https://www.hahwul.com/cullinan/history-of-owasp-top-10/`
2021, 2017, 2013, 2010, 2007, 2003
The more things change, the more they stay the same.

- Injection (3, 1, 1, 1, 2, 6)
- XSS (-, 7, 3, 2, 1, 4)
- Broken Authentication and Session Management (7, 2, 2, 3, 7, 3)
- Security Misconfiguration (5, 6, 5, 6, -, -)

Cybersecurity Jobs

- `https://cybersecurityventures.com/jobs/`
- *350 percent growth in open cybersecurity positions from 2013 to 2021*
- *3.5 million unfilled cybersecurity jobs globally by 2021, up from one million positions in 2014*

Cybersecurity Skills

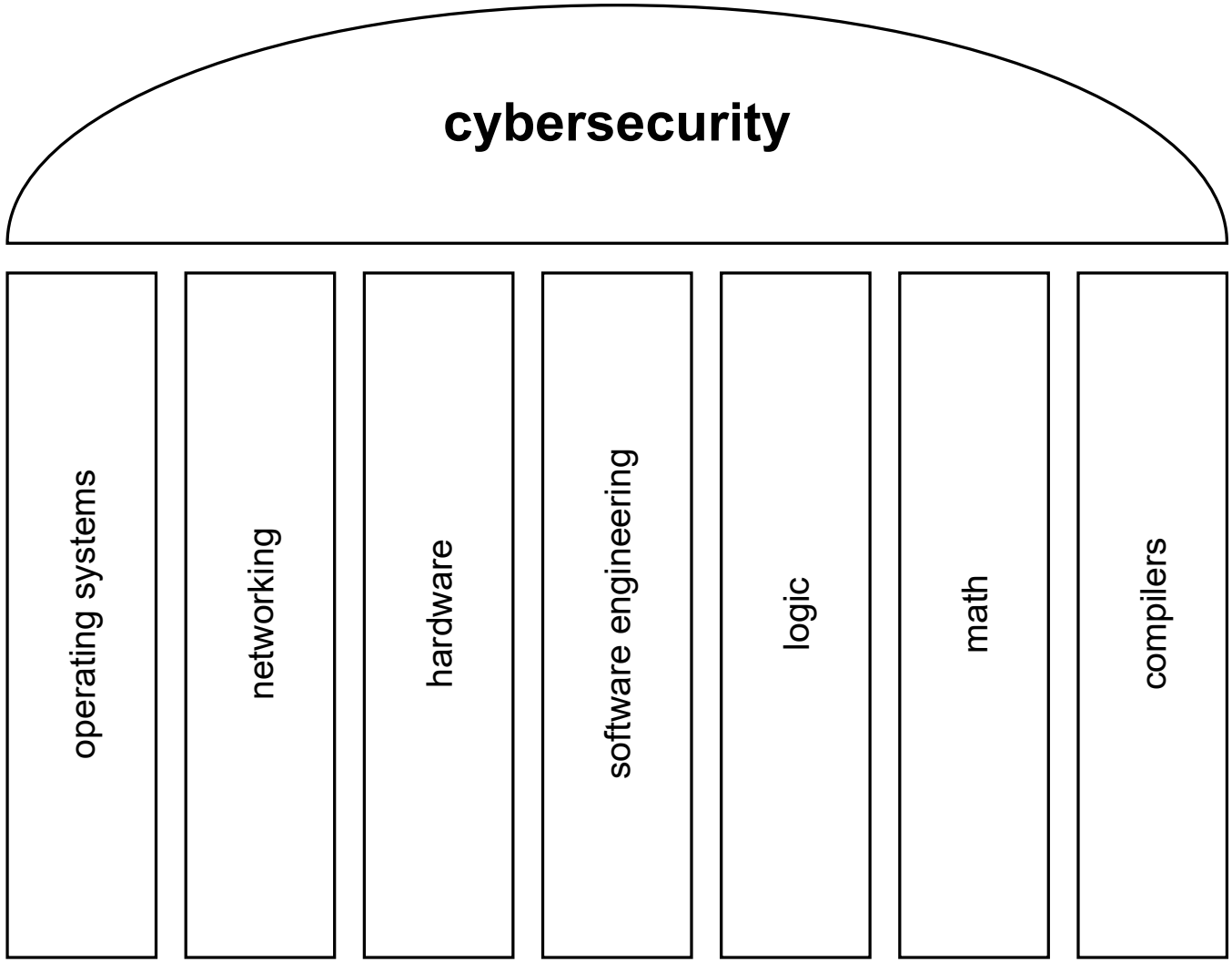


Based on the Burning Glass analysis of employer demand in postings, the skills projected to grow fastest in the next five years are:

Skill	5-Year Projected Growth	Postings Oct. 2019-Sept. 2020	Salary Premium
Application Development Security	164%	29,635	$12,266
Cloud Security	115%	19,477	$15,025
Risk Management	60%	57,967	$13,379
Threat Intelligence	41%	60,039	$9,609
Incident Response	37%	23,497	$5,683
Compliance and Controls	36%	54,770	$12,423
Data Privacy and Security	36%	88,527	$5,256
Access Management	32%	118,096	$6,451
Security Strategy and Governance	20%	82,952	$7,735
Health Information Security	20%	413,687	$738

`https://www.forbes.com/sites/louiscolumbus/2020/11/01/what-are-the-fastest-growing-cybersecurity-skills-in-2021/`

System / Application Cybersecurity



Papers

- Reflections on Trusting Trust: `https://dl.acm.org/doi/10.1145/358198.358210`
- Understanding the Mirai Botnet: `https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis`

Systems Security

- IT security, cyber security, computer security
- applications, operating system, infrastructure, networking
- hardware, software, information

Security of Information Systems

- focus on computer, operating system, software stack and infrastructure security
- little focus on networking, crypto, hardware
- research focus

Topics

- authentication, authorization
- attack, mitigation, bypass
- isolation, sandboxing
- fuzzing, symbolic execution
- defensive programming
- software analysis, verification and validation

Systems Security Research

- research: more on finding the right question, than the right answer
- not boring, but uncertain
- high risk, high impact
- secure build, attack, defend, repeat
- actual systems: software, hardware, infrastructure

Systems Security Research Venues

- big four: IEEE S&P (Oakland), ACM CCS, NDSS, Usenix Security Symposium
- ESORICS, RAID, ACSAC
- AsiaCCS, CNS, ACNS
- `http://faculty.cs.tamu.edu/guofei/sec_conf_stat.htm`
- `http://s3.eurecom.fr/~balzarot/notes/top4_v1/`

Research Results

- white papers: company product information
- workshops: preliminary work, networking
- conferences: complete work, novel results, networking
- journals: aggregated work, archive
- CVEs
- features in existing products
- start ups
- public / open source

Resources

- content website: `http://elf.cs.pub.ro/sis/`
- discussions, feedback: `https://curs.upb.ro/`
- MS Teams
- CTF platform: `https://sis-ctf.security.cs.pub.ro/`
- VM: `https://ocw.cs.pub.ro/courses/cns/resources/vm`

Logistics

- lecture: Wednesday, 4pm-6pm, PR706
- labs
  1. group 1: Wednesday, 6pm-8pm, EG403
  2. group 2: Tuesday, 8pm-10pm, PR706
- first lecture takes place on Wednesday, October 4, 2023, 4pm-6pm
- first lab (group 1) takes place on Wednesday, October 4, 2023, 6pm-8pm
- first lab (group 2) takes place on Tuesday, October 10, 2023, 8pm-10pm

Team

- Rareș Visalom
- Răzvan Deaconescu

Typical Lecture Session

- lecture + discussions
- research insights, trends
- demos, highlights
- DO ask questions, DO take part in discussion, DO challenge ideas, DO rebut ideas, DO open interesting topics (covering systems security)

Typical Lab Session

- lab/practical focused on problem solving, on approaches, not on tools
- CTF-like activity in lab: have problem, find solution
- challenge focused, not tutorial focused
- need focus, patience and perseverence
- we expect you to know the basics of: Linux/command line, network protocols and infrastructure, programming, assembly language, operating systems

CNS vs. SIS

- focus on offensive/exploiting vs focused on a building, attacking, defending
- narrow, focused, practical vs. diverse, holistic, conceptual
- binary/runtime application security vs. systems security
- engineer-centric vs. research-centric
- well-known steps vs. trends and models
- skill vs. approach

Grading

- 2p: lab activity
- 4p: assignments/projects (bonuses included)
- 4p: final exam (oral exam)
- lecture activity as a bonus (DO take part in lectures, see live and offline quizzes and surveys)

Contents

Storyline

- we have a system/infrastucture
- one way to attack it is to crack or abuse authentication
- assuming valid entry points (i.e. authentication) are secure, the attacker looks for security holes (i.e. vulnerabilities)
- the attacker exploits vulnerabilities, we use defensive mechanisms, the attacker tries to bypass them
- we assume attacks are unavoidable, we aim to confine them; we confine applications
- since applications interact with other applications and system components, we isolate the system
- proper isolation relies on separation and marking roles and validating the information flow

Storyline (cont.)

- ideally, our application or system will be secure by specification, design, implementation, validated offline
- we use static analysis and dynamic analysis (including fuzzing and symbolic execution) to validate an application offline
- a thorough approach is to use (semi)formal validation on application specifications and build it securely from those

Prerequisites

- good understanding of operating systems
- good Unix/Linux command line abilities
- good C programming skills
- fair knowledge of C shell scripting
- fair knowledge of Unix/Linux development tools
- basic understanding networking
- basic knowledge of computer architecture and assembly language

Covering Missing Know-How

- Linux/command line: USO labs: `http://ocw.cs.pub.ro/courses/uso`
- networking: RL labs: `http://ocw.cs.pub.ro/courses/rl`
- assembly: `http://ocw.cs.pub.ro/courses/iocla`
- operating systems: `http://ocw.cs.pub.ro/courses/so`

Attacker Perspective

- attacker goals
- attacker arsenal: threat model
- attacker steps: attack vector

System Model

- application control flow graph
- applications + configuration + hardware
- privileges level
- input (and output)

Scenarios

- attack iOS Device
- attack social network
- attack bank website
- attack national power grid

Defender Actions

- prevent
- isolate
- react & mitigate

Scope

- application
- operating system
- hardware
- I/O
- networking, infrastructure

Security Objectives

- privacy
- safety
- anonimity
- integrity
- confidentiality
- availability

Security vs. Safety

- safety is state
- security is process
- safety is outcome of a secure process/procedure
- security: umbrella, safety: warm and not getting wet

Trust

- validation among parties
- chain of trust: bottom-up trust
- trust anchor
- certificate chain, digital signature, certification authority

A*

- authentication: letting an entity be part of the system
- authorization: providing privileges to a given entity (subject) for resources (object)
- access control: checking whether subject can access object
- audit: inspecting action history, validating behavior

A* commands

- authentication: `login`
- authorization: `chmod`
- access control: `ls`
- audit: `find`, `auditd`

Least Privilege

- each subject is only provided the permissions it needs
- if it's not mandatory, ditch it
- sandboxing, isolation

Privilege Separation/Escalation

- split privileges among entities
- reduced actions for super user
- may require escalation (i.e. one subject may get priviliges from another)
  - be really careful about that
  - sudo, setuid, capabilities

Trusted Computing Base

- security-critical parts of the computer system
- bugs inside TCP compromise security
- i.e. priviliged parts of the computer system
- aim to reduce TCB

Attacker Mindset

- maximize profit
- time is on your side
- brute force may be an option
- you don't care as long as it works
- target many, get one

Attacker Objectives

- control
- cripple
- steal (information leak)

Threat/Adversarial Model

- decompose application, identify threats, determine coutermeasures
- classify possible attacker actions and attacker flow

Vulnerability

- misconfiguration, weakness
- exposes a risk that may be exploited for unintended outcome
- i.e. buffer overflow, integer overflow, unsanitized input

Exploit

- method to use a vulnerability for malicious outcome
- take advantage
- i.e. ret-to-libc, ROP, send injection code

Attack Vector

- steps/path to deliver a malicious outcome
- composed of attack steps (attack gadgets)
- usually chaining together exploits towards outcome
- i.e. SQL injection, actual real world attacks

Attack Surface

- parts of the system exposed to attacks
- entry points into system
- either valid entry points that may be used invalidly
- or invalid entry points
- goal is to reduce attack surface (think TCB)

Defender Mindset

- malicious vs. negligence
- time is limited
- proactive before reactive
- prevent
- monitor
- defense in depth

Proactive vs. Reactive

- similar to medicine, treating disease
- analyze threats beforehand
- deploy mitigation solutions
- react when sh*t happens
- have solutions in place

Defense in Depth

- multiple defense layers
- redundancy in defense
- assume one layer falls, others take its place

Phases for Defensive Mechanisms

- harden
- prevent
- confine
- treat

Harden

- analysis, enhancement
- off-line
- static, dynamic analysis
- verification and validation
- fuzzing, symbolic execution
- remove, treat **vulnerabilities**

Prevent

- make it difficult / impossibile for exploits to happen
- on-line, during runtime
- i.e. ASLR, DEP, stack guard, ASan

Confine

- when sh*t happens, reduce damage
- limit resources use, limit interface
- as reduced privileges as possible
- ideally customed (tailored) per app/system
- sandboxing, virtualization, access control

Treat

- if sh*t happens, solve it
- remove threat, remove damaged resources
- replace components
- apply lessons learned

Monitor

- know when sh*t happens ASAP
- proper monitoring levels; otherwise you ignore it
- monitor as much as you can
- key for availability

Keywords

- systems security
- Security of Information Systems
- goals
- safety
- trust
- trusted computing base
- least privilege
- threat model
- vulnerability
- exploit
- attack vector
- attack surface
- defense in depth
- harden
- prevent
- confine
- treat
- monitor