nexenio

Hacking Adventures

Preparation

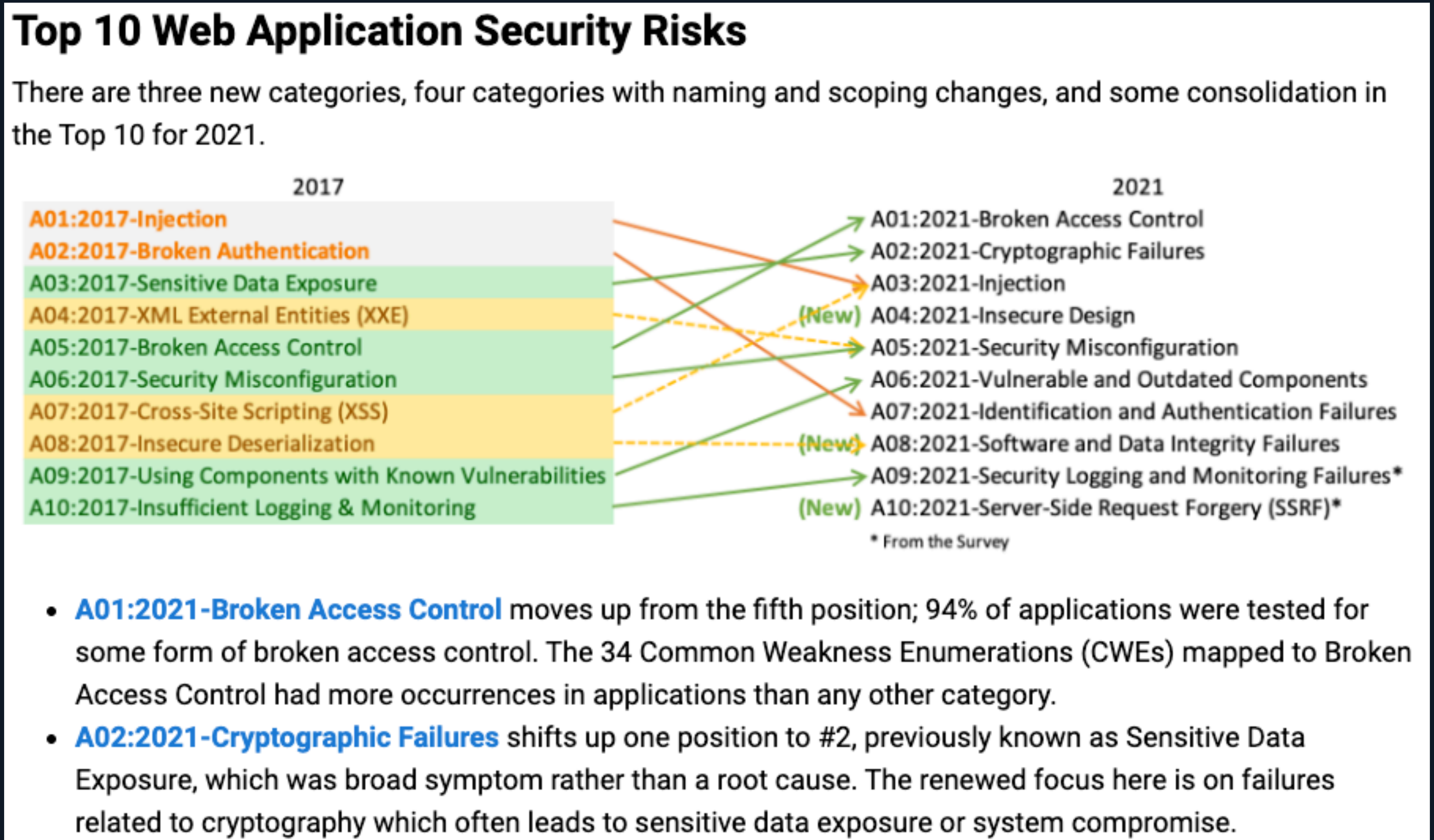
- https://github.com/neXenio/hacking-adventures
  - kotlin/challenge-3
- Install Docker
- Install Postman or similar (optional)

Goal for Today

- Impersonate another user

- Intermediate Steps
  - Complete registration
  - Reset password
  - Mess with other user accounts
  - Gain admin access
  - Mess more with other user accounts

AuthN / AuthZ



Top 10 Web Application Security Risks

There are three new categories, four categories with naming and scoping changes, and some consolidation in the Top 10 for 2021.

2017

A01:2017-Injection
A02:2017-Broken Authentication
A03:2017-Sensitive Data Exposure
A04:2017-XML External Entities (XXE)
A05:2017-Broken Access Control
A06:2017-Security Misconfiguration
A07:2017-Cross-Site Scripting (XSS)
A08:2017-Insecure Deserialization
A09:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring

2021

A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
(New) A04:2021-Insecure Design
A05:2021-Security Misconfiguration
A06:2021-Vulnerable and Outdated Components
A07:2021-Identification and Authentication Failures
(New) A08:2021-Software and Data Integrity Failures
A09:2021-Security Logging and Monitoring Failures*
(New) A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey

- **A01:2021-Broken Access Control** moves up from the fifth position; 94% of applications were tested for some form of broken access control. The 34 Common Weakness Enumerations (CWEs) mapped to Broken Access Control had more occurrences in applications than any other category.
- **A02:2021-Cryptographic Failures** shifts up one position to #2, previously known as Sensitive Data Exposure, which was broad symptom rather than a root cause. The renewed focus here is on failures related to cryptography which often leads to sensitive data exposure or system compromise.

Oversights

- No authentication
- No authorization
- Bad authorization
- Unlimited tries

- https://zerforschung.org/posts/gorillas-en/

MFA

- Factors
  - Knowledge: something you know (password)
  - Posession: something you have (smartphone)
  - Inherent: something you are (biometrics)
  - Location: where you are

Hacks

- Social Engineering
  - Man-in-the-middle (MITM) attack: https://github.com/kgretzky/evilginx2
  - Recovery Question Attacks: Teenager hacks CIA director’s account source
  - Shoulder Surfing
- Technical Hacks
  - Buggy MFA, e.g. using two identical factors
  - SIM swap attacks
  - Brute Force
  - Duplicate Code Generators
  - Malware

Hints

- Read into the rudimentary documentation
- Check the repository for secrets

Hints

- Exploit missing authentication
  - We can create as many users as we want to
- Exploit missing authorization
  - We can access sensitive information of User B by authenticating as User A
  - We can overwrite other users’ passwords
- Exploit bad password policies
  - We can guess or reverse engineer password hashes

Hints

- Timing attacks against MFA
  - if the correct OTP is 123456, checking 123400 takes longer than 023456
  - hijack the VM’s clock with libfaketime
- MFA Brute force
  - 10k attempts in 30s → 1% success / 99% failure
  - run this 70x → ~50% success → cracked in < 1h

Hints

- Rogue / Compromised Admin
  - admin can recover users’ MFA secret