

Capture The Flag (CTF) Idea Bank

Generated for: Abdelrahman Wael

September 26, 2025

Abstract

This document collects CTF challenge ideas organized by domain and difficulty, intended for use in designing practice problems or a full competition. Each entry contains a short description and optional hints, suggested files, and a canonical flag format. Hyperlinks are added for external resources to deepen understanding.

Contents

1	Introduction	3
2	Web Exploitation	3
2.1	Beginner: Getting Started	3
2.2	Intermediate: Web Exploits	3
2.3	Advanced: Complex Attacks	3
3	Cryptography	4
3.1	Beginner: Classic Ciphers	4
3.2	Intermediate: Applied Crypto	4
3.3	Advanced: Break Real Systems	4
4	Reverse Engineering	4
4.1	Beginner: Crackmes	4
4.2	Intermediate: Obfuscation	4
4.3	Advanced: Anti-Debug	4
5	Forensics	4
5.1	Beginner: Hidden in Plain Sight	4
5.2	Intermediate: Disk and Memory	4
5.3	Advanced: Memory Dumps	5
6	OSINT	5
6.1	Clues from the Open World	5
7	Pwn / Binary Exploitation	5
7.1	Beginner: Memory Basics	5
7.2	Intermediate: Control Flow	5
7.3	Advanced: Kernel Level	5
8	Misc / Fun Challenges	5
8.1	Creative Ideas	5

9 Challenge Template (Copy for Each Problem) **5**

10 Next Steps **6**

1 Introduction

This bank covers common CTF domains: Web, Crypto, Reverse Engineering, Forensics, OSINT, Pwn, and Misc. For each domain we list Beginner, Intermediate, and Advanced challenge ideas with: (1) description, (2) suggested artifacts/files, (3) sample flag format (e.g., FLAG...), and (4) short hints/solutions outline.

Useful external resources:

- [CTFtime](https://ctftime.org/CTFtime)](<https://ctftime.org/CTFtime>) – calendar and scoreboard of international CTFs.
- [picoCTF](https://picoctf.org/picoCTF)](<https://picoctf.org/picoCTF>) – beginner-friendly CTF with practice problems.
- [CryptoHack](https://cryptohack.org/CryptoHack)](<https://cryptohack.org/CryptoHack>) – learn cryptography with CTF challenges.
- [OverTheWire](https://overthewire.org/wargames/OverTheWire)](<https://overthewire.org/wargames/OverTheWire>) – wargames for Linux, pwn, and networking.

2 Web Exploitation

2.1 Beginner: Getting Started

- **SQL Injection: Login Bypass** – vulnerable login form accepting unsanitized input. Resource: [PortSwiggyer](https://portswigger.net/web-security/sql-injection)](<https://portswigger.net/web-security/sql-injection>) PortSwiggyer SQLi Guide. *Flag: FLAGsql_injection_success*
- **XSS (Reflected)** – search query reflected without proper escaping. Resource: [OWASP](https://owasp.org/www-community/attacks/xss)](<https://owasp.org/www-community/attacks/xss>) OWASP XSS Overview. *Flag: FLAGxss_relected*
- **Cookie Tampering** – important user role stored in client cookie. *Flag: FLAGcookie_admin*

2.2 Intermediate: Web Exploits

- **CSRF Token Manipulation** – predictable CSRF token generation. Resource: [CSRF](https://portswigger.net/web-security/csrf)](<https://portswigger.net/web-security/csrf>) CSRF Explanation.
- **SSRF to Internal File** – server-side request forgery. Resource: [SSRF](https://portswigger.net/web-security/ssrf)](<https://portswigger.net/web-security/ssrf>) SSRF Walkthrough.
- **Insecure File Upload** – bypass content type checks. Resource: [OWASP](https://owasp.org/www-community/vulnerabilities/UnrestrictedFileUpload)](<https://owasp.org/www-community/vulnerabilities/UnrestrictedFileUpload>) OWASP File Upload.

2.3 Advanced: Complex Attacks

- **Deserialization Exploit** – unsafe object deserialization. Resource: [OWASP](https://owasp.org/www-community/vulnerabilities/Deserialization_of_untrusted_data)](https://owasp.org/www-community/vulnerabilities/Deserialization_of_untrusted_data) OWASP Deserialization.
- **Prototype Pollution (Node.js)** – mutate prototype to escalate privileges. Resource: [Prototype](https://learn.snyk.io/lessons/prototype-pollution/javascript)](<https://learn.snyk.io/lessons/prototype-pollution/javascript>) Prototype Pollution.

3 Cryptography

3.1 Beginner: Classic Ciphers

- **Caesar / ROT** – simple substitution cipher. Resource: [CryptoHack](https://cryptohack.org/CryptoHack).
](<https://cryptohack.org/CryptoHack>.
)
- **XOR Single-Byte** – recover with crib-dragging.

3.2 Intermediate: Applied Crypto

- **RSA with Reused Primes** – exploit shared primes with gcd. Resource:

3.3 Advanced: Break Real Systems

- **Lattice Attack** – small-exponent RSA. Resource: [Lattice](https://en.wikipedia.org/wiki/Lattice_reduction)](https://en.wikipedia.org/wiki/Lattice_reduction). Reduction.
- **Timing Side-Channel** – recover key bits from timing leaks. Resource: [Timing](https://crypto.stackexchange.com/questions/11377/what-are-timing-attacks)](<https://crypto.stackexchange.com/questions/11377/what-are-timing-attacks>) Attacks.

4 Reverse Engineering

4.1 Beginner: Crackmes

- **Password Check Binary** – reverse to find password. Resource: [Crackmes.one](https://crackmes.one/).
](<https://crackmes.one/>Crackmes.one.
)

4.2 Intermediate: Obfuscation

- **Packed Executable** – UPX or custom packer.

4.3 Advanced: Anti-Debug

- **Anti-Debugging VM** – binary uses anti-debug tricks or custom VM.

5 Forensics

5.1 Beginner: Hidden in Plain Sight

- **Image Metadata** – extract EXIF. Resource: [ExifTool](https://exiftool.org/ExifTool).
](<https://exiftool.org/ExifTool>.
)
- **PCAP Analysis** – HTTP capture with flag. Resource: [Wireshark](https://www.wireshark.org/Wireshark).
](<https://www.wireshark.org/Wireshark>.
)

5.2 Intermediate: Disk and Memory

- **Hidden Partition** – disk image with hidden files.

5.3 Advanced: Memory Dumps

- **Memory Forensics** – analyze RAM dump. Resource: [Volatility](#).
](<https://www.volatilityfoundation.org/Volatility>).
)

6 OSINT

6.1 Clues from the Open World

- **Image Geolocation** – landmarks in photo. Resource: [Google](#)](<https://www.google.com/maps>) Google Maps.
- **Username Correlation** – accounts across platforms. Resource: [WhatsMyName](#).
](<https://whatsmyname.app/WhatsMyName>).
)

7 Pwn / Binary Exploitation

7.1 Beginner: Memory Basics

- **Classic Buffer Overflow**. Resource: [Exploit](#)](<https://exploit.education/Exploit>) Education.

7.2 Intermediate: Control Flow

- **ROP Chain** – bypass NX. Resource: [ROP](#)](<https://ropemporium.com/ROP>) Emporium.

7.3 Advanced: Kernel Level

- **Kernel Exploit**.

8 Misc / Fun Challenges

8.1 Creative Ideas

- **Stego in QR Code** – flag hidden in QR. Resource: [QR](#)](<https://www.qrstuff.com/QR>) Stuff.
- **AI Prompt Injection** – coax model into revealing flag. Resource: [Prompt](#)](<https://arxiv.org/abs/2402.11>) Injection Research.
- **Smart Contract** – reentrancy exploit. Resource: [Ethernaut](#).
](<https://ethernaut.openzeppelin.com/Ethernaut>).
)

9 Challenge Template (Copy for Each Problem)

1. **Title:**
2. **Category:**

3. **Difficulty:** Beginner / Intermediate / Advanced
4. **Description:**
5. **Files:**
6. **Flag format:** e.g., FLAG...
7. **Hints:**
8. **Author notes / solution outline:**

10 Next Steps

Use this Overleaf file to expand each idea into a full problem statement, create attachments, and prepare test infrastructure (docker, VM images, or web hosts).