

Homework

Three tasks that deepen your understanding of mask design, attack benchmarking, and multi-stage cracking strategies.

Complete all five lab exercises before starting the homework. These tasks build on the skills you practiced in the lab.

Task 1 — Custom Charset Design

Objective

Design a set of custom charsets tailored to a specific password policy, then use them to crack test hashes.

Instructions

1. Choose a password policy (real or hypothetical). Example: "8 characters minimum, must contain at least one uppercase letter, one lowercase letter, one digit, and one of the following special characters: `!@#$`"
2. Create 5 test passwords that comply with this policy. Generate their MD5 hashes.
3. Design masks using custom charsets (`-1`, `-2`, etc.) that model the policy constraints. Write at least 3 different masks targeting different structural patterns:
 - Example mask 1: `-1 !@#$ ?u?l?l?l?l?d?d?1` (uppercase + 4 lower + 2 digits + special)
 - Example mask 2: `-1 !@#$ ?1?u?l?l?l?l?l?d` (special + uppercase + 5 lower + digit)
 - Example mask 3: `-1 !@#$ ?d?d?u?l?l?l?l?l?1` (2 digits + uppercase + 4 lower + special)

4. Run each mask against your test hashes. Record which passwords each mask cracks.
5. Write a brief explanation (3-5 sentences) of why multiple masks are needed even with custom charsets.

Deliverable

A text file or document containing: the password policy, your 5 test passwords, the 3+ masks with their commands, results, and your explanation.

Task 2 — Attack Speed Benchmarking

Objective

Compare the performance of mask, hybrid, and dictionary attacks on identical hash sets to understand when each mode is most efficient.

Instructions

1. Create a set of 10 test passwords that include a mix of:
 - Pure dictionary words (e.g., "sunshine")
 - Words with appended digits (e.g., "sunshine123")
 - Pattern-based passwords (e.g., "Abcd1234")
 - Words with prepended digits (e.g., "99sunshine")
2. Generate their MD5 hashes into `bench_hashes.txt`.
3. Run the following attacks and record the time, speed, and number of passwords cracked for each:

```
# Dictionary attack
hashcat -m 0 -a 0 bench_hashes.txt rockyou.txt

# Hybrid append
hashcat -m 0 -a 6 bench_hashes.txt rockyou.txt ?d?d?d

# Hybrid prepend
hashcat -m 0 -a 7 bench_hashes.txt ?d?d rockyou.txt
```

```
# Mask attack (8-char mixed)
hashcat -m 0 -a 3 bench_hashes.txt ?u?l?l?l?d?d?d?d
```

4. Create a results table:

Attack Mode	Passwords Cracked	Time	Speed (H/s)
Dictionary (-a 0)	?/10	?	?
Hybrid append (-a 6)	?/10	?	?
Hybrid prepend (-a 7)	?/10	?	?
Mask (-a 3)	?/10	?	?

5. Write a brief analysis (5-8 sentences): Which attack mode cracked the most passwords? Which was fastest? Were there passwords that only one mode could crack? What does this tell you about real-world cracking strategy?

Tip: Use `--potfile-disable` between runs so previously cracked hashes don't get skipped. Alternatively, clear the potfile before each run.

Task 3 — Multi-Stage Cracking Script

Objective

Write a shell script that chains multiple attack modes in order of efficiency, stopping when all hashes are cracked or all attacks are exhausted.

Instructions

1. Create a Bash script called `crack_chain.sh` that accepts a hash file as an argument.
2. The script should run the following attacks in sequence:
 1. **Stage 1:** Dictionary attack with `rockyou.txt`
 2. **Stage 2:** Hybrid append — wordlist + `?d?d`

3. **Stage 3:** Hybrid append — wordlist + `?d?d?d`
 4. **Stage 4:** Hybrid prepend — `?d?d` + wordlist
 5. **Stage 5:** Mask for 4-digit PINs: `?d?d?d?d`
 6. **Stage 6:** Mask for 6-digit PINs: `?d?d?d?d?d?d`
 7. **Stage 7:** Mask for name+digits: `?u?l?l?l?l?d?d`
3. Between each stage, print a status message showing which stage just completed and how many hashes have been cracked so far.
 4. Test the script against a set of 10+ test hashes covering diverse password patterns.

Starter Template

```
#!/bin/bash
# crack_chain.sh - Multi-stage hashcat cracking script

HASHFILE="$1"
WORDLIST="rockyou.txt"
HASHTYPE=0 # MD5

if [ -z "$HASHFILE" ]; then
    echo "Usage: $0 <hashfile>"
    exit 1
fi

echo "=== Stage 1: Dictionary Attack ==="
hashcat -m $HASHTYPE -a 0 "$HASHFILE" "$WORDLIST"
echo "Stage 1 complete."

echo "=== Stage 2: Hybrid Append ?d?d ==="
hashcat -m $HASHTYPE -a 6 "$HASHFILE" "$WORDLIST" ?d?d
echo "Stage 2 complete."

echo "=== Stage 3: Hybrid Append ?d?d?d ==="
hashcat -m $HASHTYPE -a 6 "$HASHFILE" "$WORDLIST" ?d?d?d
echo "Stage 3 complete."

# Continue with remaining stages...
echo "=== Stage 4: Hybrid Prepend ?d?d ==="
```

Enhancement ideas: Add a function that counts remaining uncracked hashes after each stage and skips further stages if all are cracked. Add a `-quiet` mode that suppresses Hashcat's verbose output. Add timing information for each stage.

Submission Checklist

- Task 1: Password policy defined; 5 test passwords + hashes; 3+ custom masks; results and explanation
- Task 2: 10 test passwords benchmarked across 4 attack modes; results table; written analysis
- Task 3: `crack_chain.sh` script created; tested against 10+ hashes; stage output visible