

Homework

Research, identification, and benchmarking tasks to deepen your hash type knowledge.

Submission: Document your findings in a text file or PDF. Include screenshots of hashcat output where requested. Be ready to present your cheat sheet in the next class.

Task 1 — Research 5 Additional Hash Types

What to do

Research and document 5 hash types **not covered in the lecture**. For each hash type, provide all the information listed below.

Requirements

For each of the 5 hash types, document:

- **Algorithm name** (e.g., Keccak-256, RIPEMD-160, Whirlpool)
- **Hashcat mode number** (`-m` value)
- **Hash length and format** (hex length, any prefix, encoding)
- **Where it is commonly found** (which applications, operating systems, or protocols use it)
- **Whether it is salted** and the salt format if applicable
- **An example hash** (from `hashcat --example-hashes`)

Suggested hash types to research

- PBKDF2-HMAC-SHA256 (mode 10900)
- Kerberos 5 TGS-REP (mode 13100)
- MySQL 4.1+ (mode 300)
- MSSQL 2012/2014 (mode 1731)

- Drupal7 (mode 7900)
- LM (mode 3000)
- NetNTLMv2 (mode 5600)
- scrypt (mode 8900)

```
# Use this command to find example hashes for any mode
hashcat --example-hashes | grep -A6 'MODE: 10900'
```

```
# Or search by name
```

```
hashcat --example-hashes | grep -i 'kerberos' -B2 -A6
```

Self-check: Can you explain to a teammate why each hash type is used in its specific context? Do you know how salting works for each one?

Task 2 — Build a Hash Identification Cheat Sheet

What to do

Create a one-page reference document (table format recommended) that you can use during CTF competitions or penetration tests to quickly identify an unknown hash.

Requirements

Your cheat sheet must include at least 15 hash types and these columns:

Algorithm	Mode	Length	Prefix	Salted	Speed	Recommended Attack
MD5	0	32 hex	none	No	Fast	Large wordlist + rules
bcrypt	3200	60 chars	\$2a\$ / \$2b\$	Yes	Very Slow	Small targeted wordlist

... continue for at least 13 more types ...

Bonus: Decision flowchart

Optionally, create a text-based decision flowchart for hash identification:

```
Unknown hash
├─ Starts with $?
│   ├── $2a$ / $2b$ / $2y$ → bcrypt (3200)
│   ├── $6$ → SHA-512crypt (1800)
│   ├── $5$ → SHA-256crypt (7400)
│   ├── $1$ → MD5crypt (500)
│   └── $apr1$ → Apache MD5 (1600)
├─ Starts with WPA*?
│   └── WPA/WPA2 (22000)
├─ Pure hex?
│   ├── 32 chars → MD5 (0) or NTLM (1000)
│   ├── 40 chars → SHA-1 (100)
│   ├── 64 chars → SHA-256 (1400)
│   └── 128 chars → SHA-512 (1700)
└─ Contains colon (:)?
    └── hash:salt → check salted variants
```

Self-check: Does your cheat sheet cover every hash type from the lecture plus 5 from Task 1? Could a teammate use it without any additional explanation?

Task 3 — Benchmark and Feasibility Analysis

What to do

Run hashcat benchmarks on your hardware for 3 different hash types (one fast, one medium, one slow). Then calculate the maximum brute-force keyspace you can exhaust in 24 hours for each.

Step 1: Run benchmarks

```
# Benchmark three hash types
hashcat -b -m 0          # MD5 (fast)
hashcat -b -m 1400       # SHA-256 (medium)
hashcat -b -m 3200       # bcrypt (slow)
```

```
# Record the Speed.#1 value for each
```

Step 2: Calculate keyspace feasibility

For each hash type, calculate the maximum brute-force keyspace you can cover in 24 hours (86,400 seconds):

```
# Formula
```

```
Max keyspace in 24h = Speed (H/s) × 86,400
```

```
# Then determine the maximum password length you can fully brute-force:
```

```
# Lowercase only (26 chars): 26^N candidates for N-length passwords
```

```
# Lowercase + digits (36 chars): 36^N candidates
```

```
# Full printable ASCII (95 chars): 95^N candidates
```

Fill in the results table

Hash Type	Speed (H/s)	24h Keyspace	Max Length (a-z)	Max Length (a-z0-9)	Max Length (all printable)
MD5					
SHA-256					
bcrypt					

Step 3: Write a brief analysis

Answer these questions in 2-3 sentences each:

1. For MD5, what is the longest all-lowercase password you can brute-force in 24 hours? What about with the full printable charset?
2. For bcrypt, is brute-force of even a 6-character lowercase password feasible in 24 hours? Show the math.
3. Based on your results, what is the minimum password length you would recommend for systems using bcrypt? For systems using unsalted MD5?

Self-check: Do your numbers make sense? MD5 should allow brute-forcing much longer passwords than bcrypt. If bcrypt brute-force of 8+ chars is

feasible on your hardware in 24h, double-check your math.

Grading Criteria

Task	Points	Key Criterion
Task 1 — Research	30	5 hash types fully documented with all 6 required fields
Task 2 — Cheat sheet	35	15+ hash types, all columns filled, usable as a quick reference
Task 3 — Benchmarks	35	Correct calculations, reasonable analysis, screenshots of benchmark output