CRYPTOGRAPHY

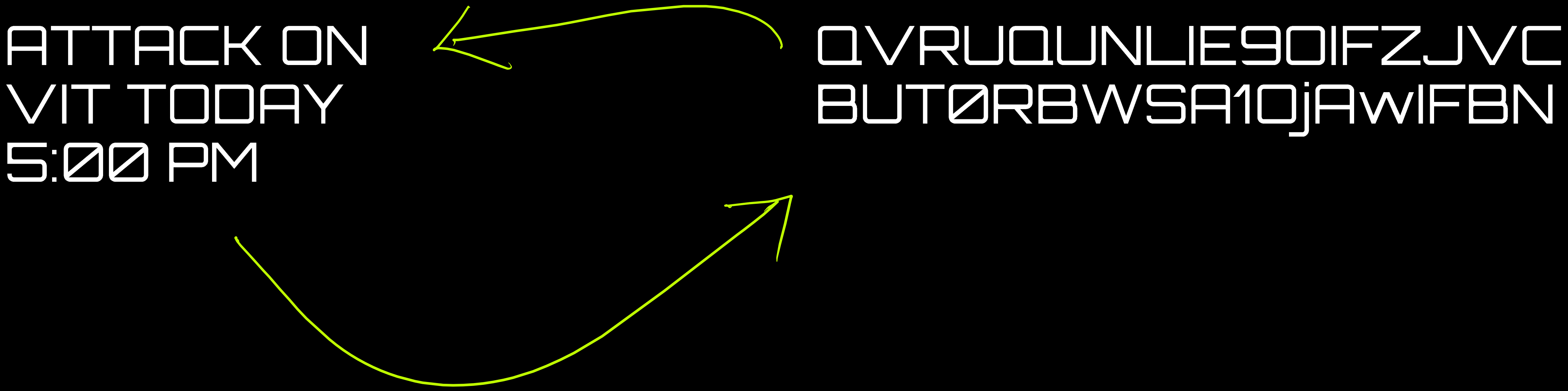
Because Your Crush's DMs Deserve AES, Not ROT13

ENCODING

ENCRYPTION

Encoding
百科事典
IS THIS ENCRYPTION
imgflip.com

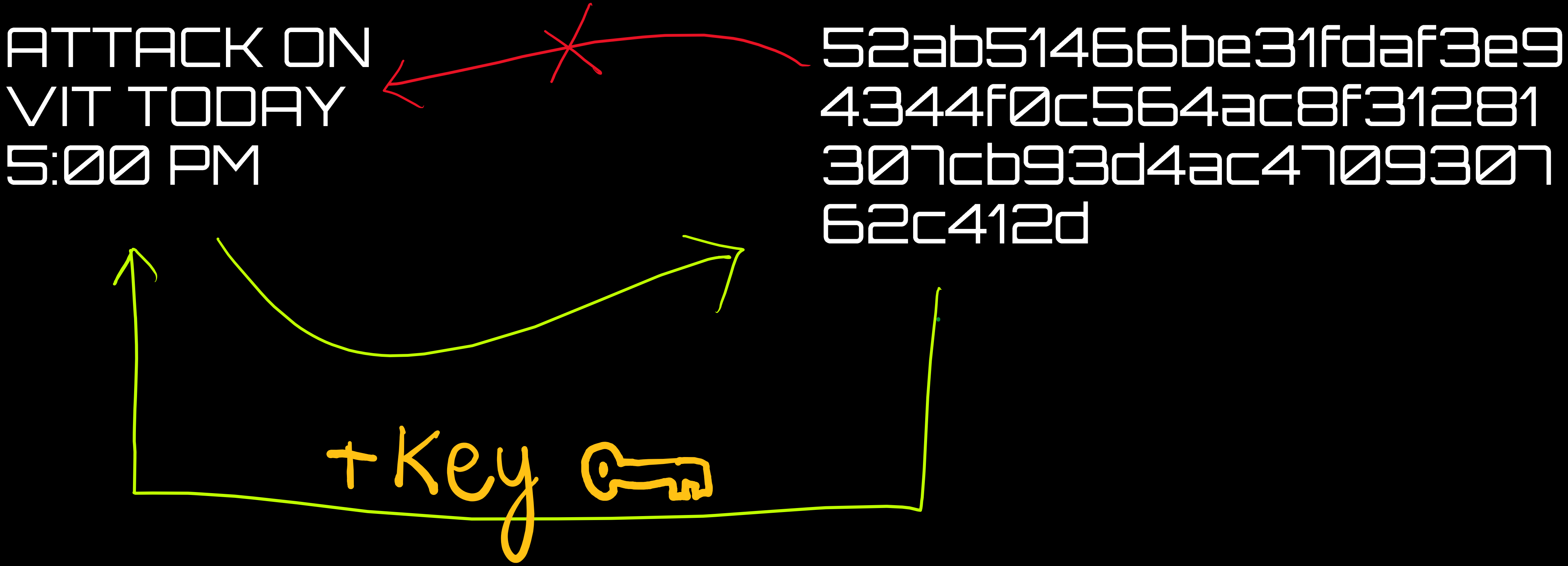
ENCODING (BASE64)



ATTACK ON
VIT TODAY
5:00 PM

QVRUQUNLIE9OIFZJVC
BUT0RBWSA1OjAwIFBN

ENCRYPTION (SHA256)

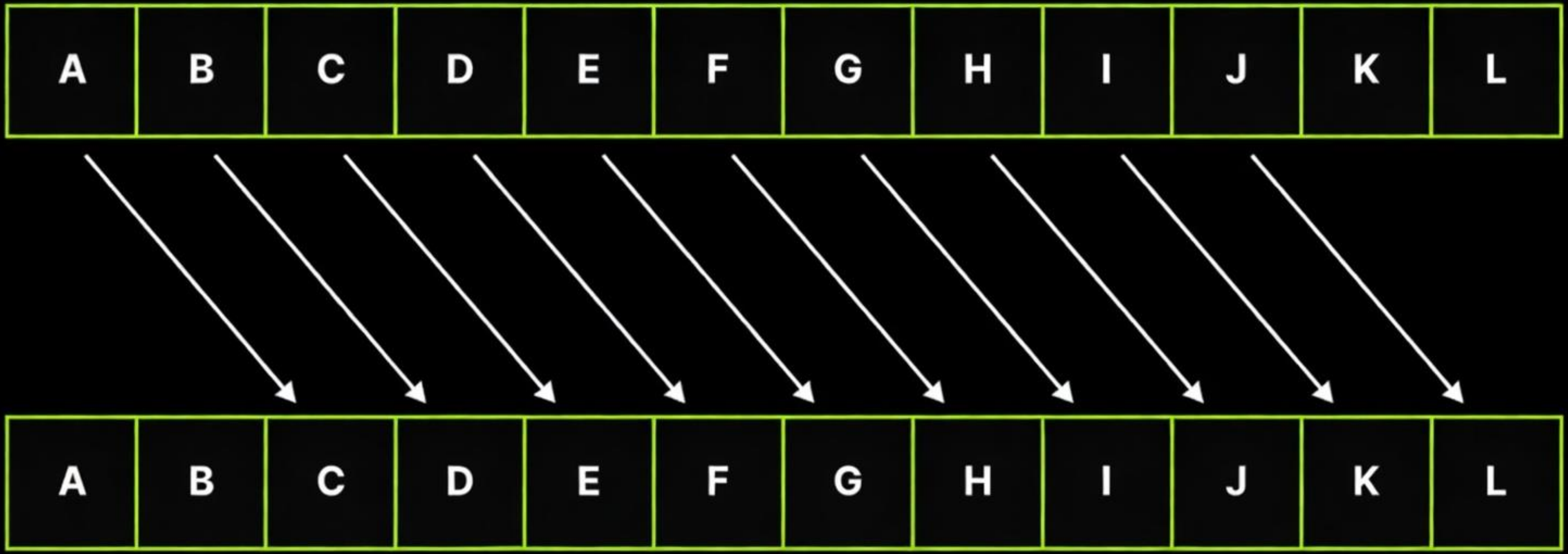


ATTACK ON
VIT TODAY
5:00 PM

52ab51466be31fdaf3e9
4344f0c564ac8f31281
307cb93d4ac4709307
62c412d

CAESAR CIPHER

K = 2 Shifts the alphabet 2 characters to the right



A B C D E F G H I J K L

A B C D E F G H I J K L

EXAMPLE

- Plaintext: HELLO
- Ciphertext: KHOOR
- (H → K, E → H, L → O, L → O, O → R)

GENERAL FORMULA

Encryption:

```
C = (P + shift) % 26
```

Decryption:

```
P = (C - shift) % 26
```

ROT13

- Special case of Caesar cipher with a shift of 13.
- Applying ROT13 twice restores the original text.

EXAMPLE

- Plaintext: HELLO
- Ciphertext: URYYB

(H → U, E → R, L → Y, L → Y, O → B)

- Decryption
- Apply ROT13 again: URYYB → HELLO

VIGENÈRE CIPHER

- Uses a repeating key to shift each letter differently.

EXAMPLE

- Plaintext: NAVIA
- Key: KEY

N	A	V	I	A
13	0	21	8	0
K	E	Y	K	E
10	4	24	10	4

N	A	V	I	A
13	0	21	8	0
K	E	Y	K	E
10	4	24	10	4
		=		
23	4	45	18	4
23	4	19	18	4
X	E	T	S	E

BASE64

Binary -> ASCII

Plaintext: PARROT

Ciphertext: UGFycm90

EXAMPLE

- P → 80 → 01010000
- a → 97 → 01100001
- r → 114 → 01110010
- r → 114 → 01110010
- o → 111 → 01101111
- t → 116 → 01110100

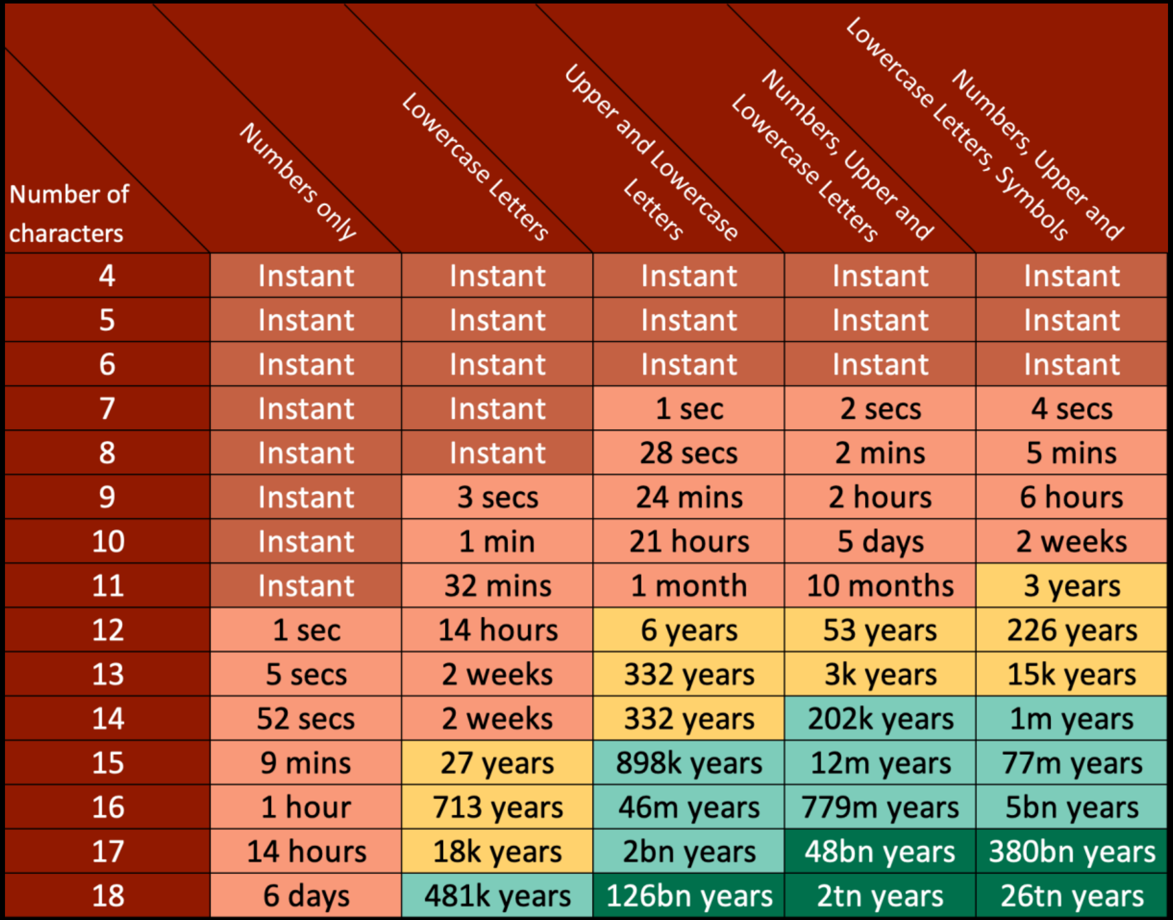
EXAMPLE

- 010100 000110 000101 011100 100111 001001 101111 011101

ENOUGH THEORY!!!!

Let's get to attacking

CHOOSING PASSWORD?

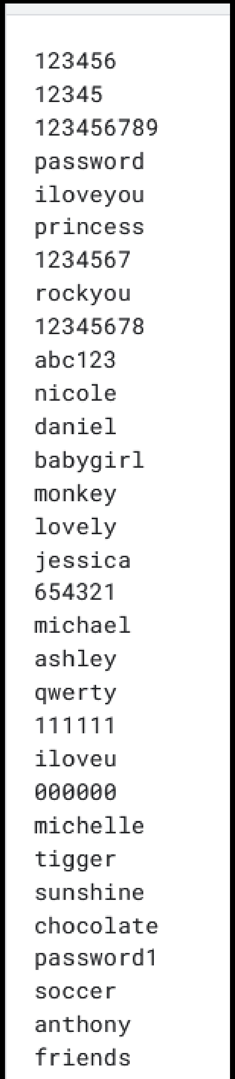


Number of characters	Numbers only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instant	Instant	Instant	Instant	Instant
5	Instant	Instant	Instant	Instant	Instant
6	Instant	Instant	Instant	Instant	Instant
7	Instant	Instant	1 sec	2 secs	4 secs
8	Instant	Instant	28 secs	2 mins	5 mins
9	Instant	3 secs	24 mins	2 hours	6 hours
10	Instant	1 min	21 hours	5 days	2 weeks
11	Instant	32 mins	1 month	10 months	3 years
12	1 sec	14 hours	6 years	53 years	226 years
13	5 secs	2 weeks	332 years	3k years	15k years
14	52 secs	2 weeks	332 years	202k years	1m years
15	9 mins	27 years	898k years	12m years	77m years
16	1 hour	713 years	46m years	779m years	5bn years
17	14 hours	18k years	2bn years	48bn years	380bn years
18	6 days	481k years	126bn years	2tn years	26tn years

BRUTEFORCE IS BAD

What to do?

Dictionary Attack

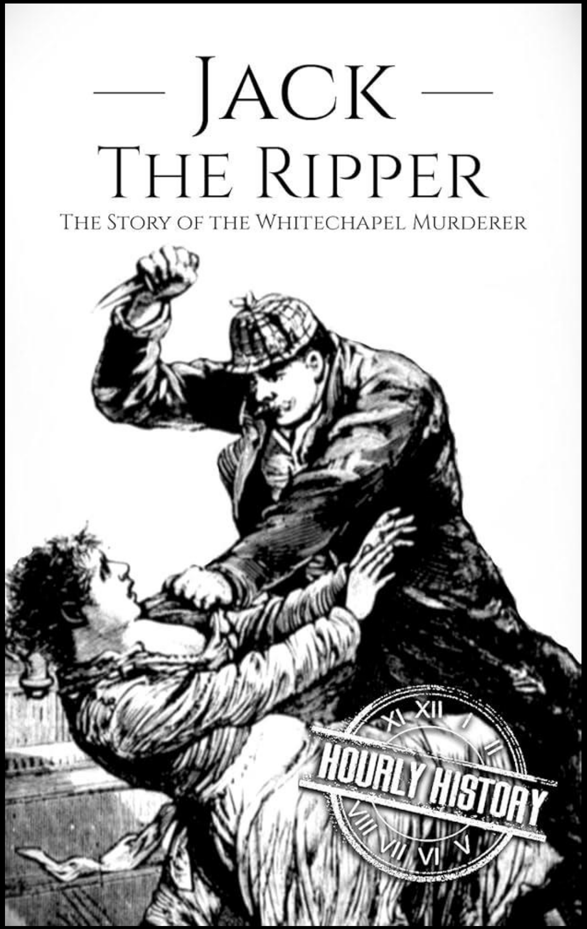


```
123456
12345
123456789
password
iloveyou
princess
1234567
rockyou
12345678
abc123
nicole
daniel
babygirl
monkey
lovely
jessica
654321
michael
ashley
qwerty
111111
iloveu
000000
michelle
tigger
sunshine
chocolate
password1
soccer
anthony
friends
```

HASHCAT

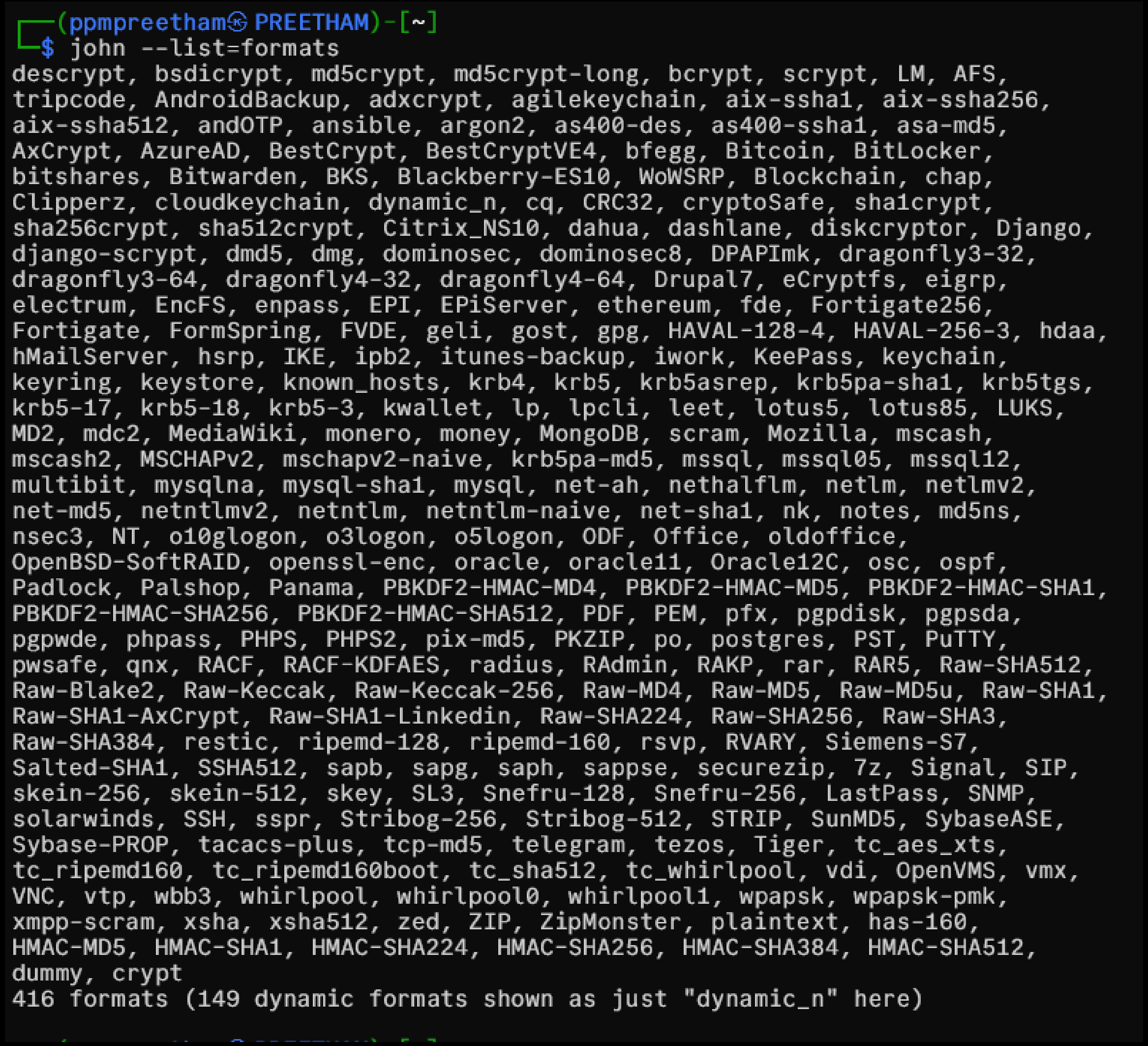
😺

JOHN THE RIPPER



JOHN THE RIPPER

- john --list=formats
- Supports 416 formats



```
┌──(ppmpreetham㉿PREETHAM)-[~]
└─$ john --list=formats
descrypt, bsdicrypt, md5crypt, md5crypt-long, bcrypt, scrypt, LM, AFS,
tripcode, AndroidBackup, adxcrypt, agilekeychain, aix-ssha1, aix-ssha256,
aix-ssha512, andOTP, ansible, argon2, as400-des, as400-ssha1, asa-md5,
AxCrypt, AzureAD, BestCrypt, BestCryptVE4, bfegg, Bitcoin, BitLocker,
bitshares, Bitwarden, BKS, Blackberry-ES10, WoWSRP, Blockchain, chap,
Clipperz, cloudkeychain, dynamic_n, cq, CRC32, cryptoSafe, sha1crypt,
sha256crypt, sha512crypt, Citrix_NS10, dahua, dashlane, diskcryptor, Django,
django-scrypt, dmd5, dmg, dominosec, dominosec8, DPAPImk, dragonfly3-32,
dragonfly3-64, dragonfly4-32, dragonfly4-64, Drupal7, eCryptfs, eigrp,
electrum, EncFS, enpass, EPI, EPiServer, ethereum, fde, Fortigate256,
Fortigate, FormSpring, FVDE, geli, gost, gpg, HAVAL-128-4, HAVAL-256-3, hdaa,
hMailServer, hsrp, IKE, ipb2, itunes-backup, iwork, KeePass, keychain,
keyring, keystore, known_hosts, krb4, krb5, krb5asrep, krb5pa-sha1, krb5tgs,
krb5-17, krb5-18, krb5-3, kwallet, lp, lpcli, leet, lotus5, lotus85, LUKS,
MD2, mdc2, MediaWiki, monero, money, MongoDB, scram, Mozilla, mscash,
mscash2, MSCHAPv2, mschapv2-naive, krb5pa-md5, mssql, mssql05, mssql12,
multibit, mysqlna, mysql-sha1, mysql, net-ah, nethalflm, netlm, netlmv2,
net-md5, netntlmv2, netntlm, netntlm-naive, net-sha1, nk, notes, md5ns,
nsec3, NT, o10glogon, o3logon, o5logon, ODF, Office, oldoffice,
OpenBSD-SoftRAID, openssl-enc, oracle, oracle11, Oracle12C, osc, ospf,
Padlock, Palshop, Panama, PBKDF2-HMAC-MD4, PBKDF2-HMAC-MD5, PBKDF2-HMAC-SHA1,
PBKDF2-HMAC-SHA256, PBKDF2-HMAC-SHA512, PDF, PEM, pfx, pgpdisk, pgpsda,
pgpwde, phpass, PHPS, PHPS2, pix-md5, PKZIP, po, postgres, PST, PuTTY,
pwsafe, qnx, RACF, RACF-KDFAES, radius, RAdmin, RAKP, rar, RAR5, Raw-SHA512,
Raw-Blake2, Raw-Keccak, Raw-Keccak-256, Raw-MD4, Raw-MD5, Raw-MD5u, Raw-SHA1,
Raw-SHA1-AxCrypt, Raw-SHA1-Linkedin, Raw-SHA224, Raw-SHA256, Raw-SHA3,
Raw-SHA384, restic, ripemd-128, ripemd-160, rsvp, RVARY, Siemens-S7,
Salted-SHA1, SSHA512, sapb, sapg, saph, sappse, securezip, 7z, Signal, SIP,
skein-256, skein-512, skey, SL3, Snefru-128, Snefru-256, LastPass, SNMP,
solarwinds, SSH, sspr, Stribog-256, Stribog-512, STRIP, SunMD5, SybaseASE,
Sybase-PROP, tacacs-plus, tcp-md5, telegram, tezos, Tiger, tc_aes_xts,
tc_ripemd160, tc_ripemd160boot, tc_sha512, tc_whirlpool, vdi, OpenVMS, vmx,
VNC, vtp, wbb3, whirlpool, whirlpool0, whirlpool1, wpapsk, wpapsk-pmk,
xmpp-scram, xsha, xsha512, zed, ZIP, ZipMonster, plaintext, has-160,
HMAC-MD5, HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512,
dummy, crypt
416 formats (149 dynamic formats shown as just "dynamic_n" here)
```

JOHN THE RIPPER

GOAL	COMMAND EXAMPLE
Fast dictionary + rules	`john --wordlist=rockyou.txt --rules hashes.txt`
Raw-MD5 hash	`john --format=raw-md5 md5.txt`
Windows NTLM	`john --format=NT ntlm.txt`
Linux sha512crypt	`john shadow_combined.txt`
Cracked passwords	`john --show hashes.txt`
ZIP password	`zip2john file.zip > z.hash && john z.hash`
PDF password	`pdf2john file.pdf > p.hash && john p.hash`
Mask (e.g. Name + year)	`john --mask=?u?l?l?l?d?d?d?d --wordlist=names.txt hashes.txt`
GPU acceleration (OpenCL)	`john --format=sha512crypt-opencl hashes.txt`

EXAMPLE:

This message is in SHA256, decrypt it:

e4ad93ca07acb8d908a3aa41e920ea4f4ef4f26e
7f86cf8291c5db289780a5ae

EXAMPLE:

```
john --format=raw-sha256 smtg.hash
```

FOR PDF?

```
>> pdf2john file.pdf > p.hash
>> john p.hash
```

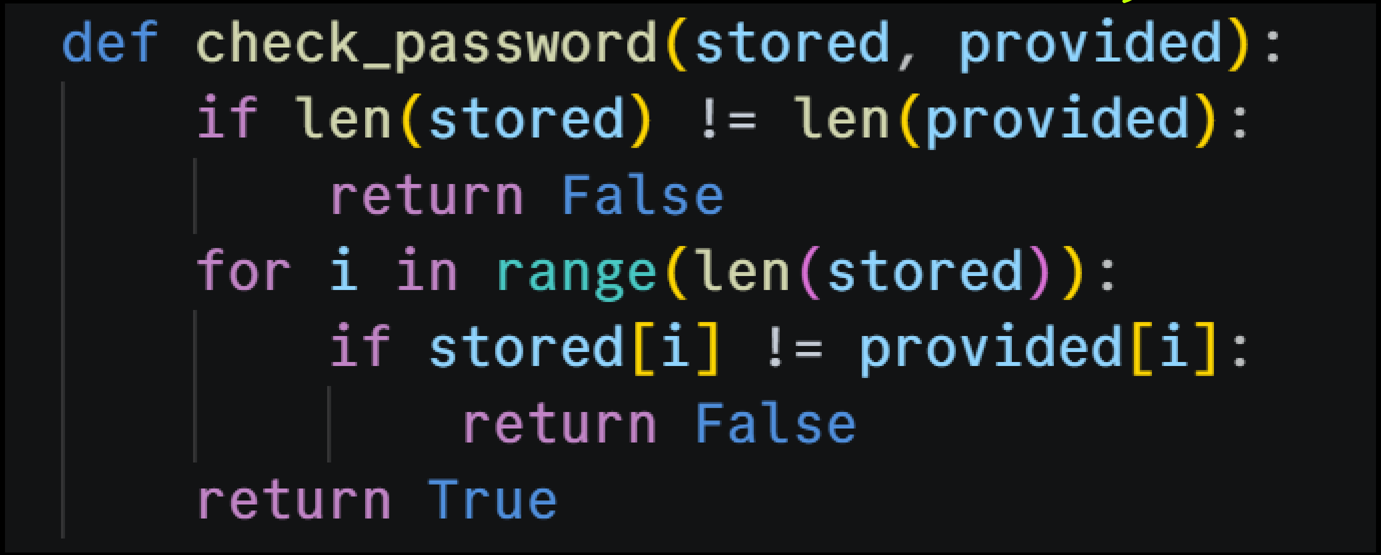
FOR ZIP?

```
>> zip2john file.pdf > p.hash
>> john p.hash
```

SIDE CHANNEL ATTACKS

WAIT WHATTTTT? (BONUS CONTENT)

WHAT'S WRONG HERE?



```python
def check_password(stored, provided):
    if len(stored) != len(provided):
        return False
    for i in range(len(stored)):
        if stored[i] != provided[i]:
            return False
    return True
```

input

Now watch what happens when an attacker sends guesses:

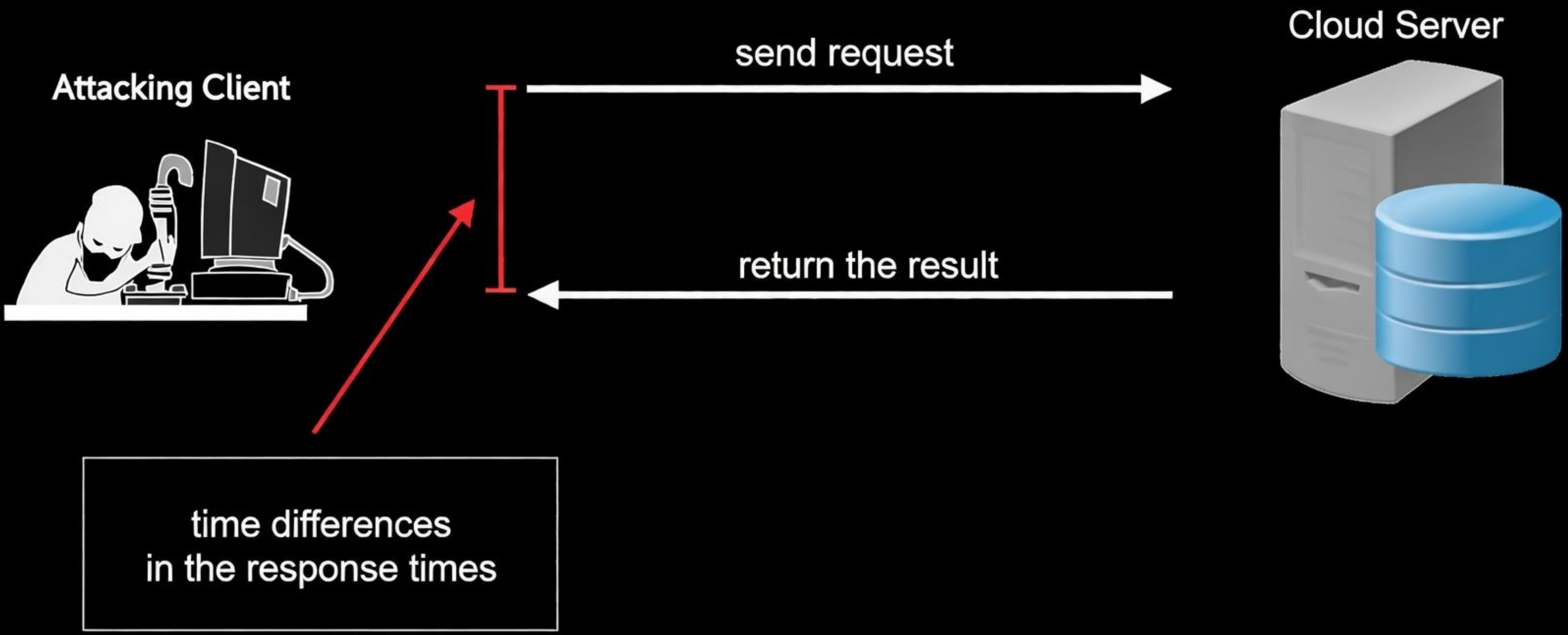
Actual: password1

Guess	Time taken (relative)	Why?
wrong	very fast	length mismatch → immediate return
passwX	fast	differs at position 5
passworX	a bit slower	differs at position 8
passwordX	even slower	differs at position 9
password!	slowest (almost correct)	only fails at last character

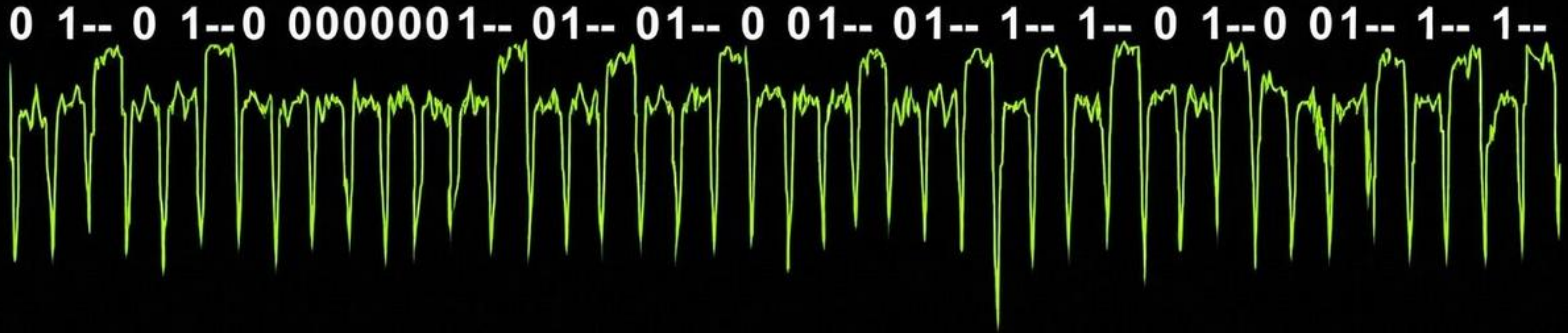
TIMING ATTACK

BRUTEFORCE

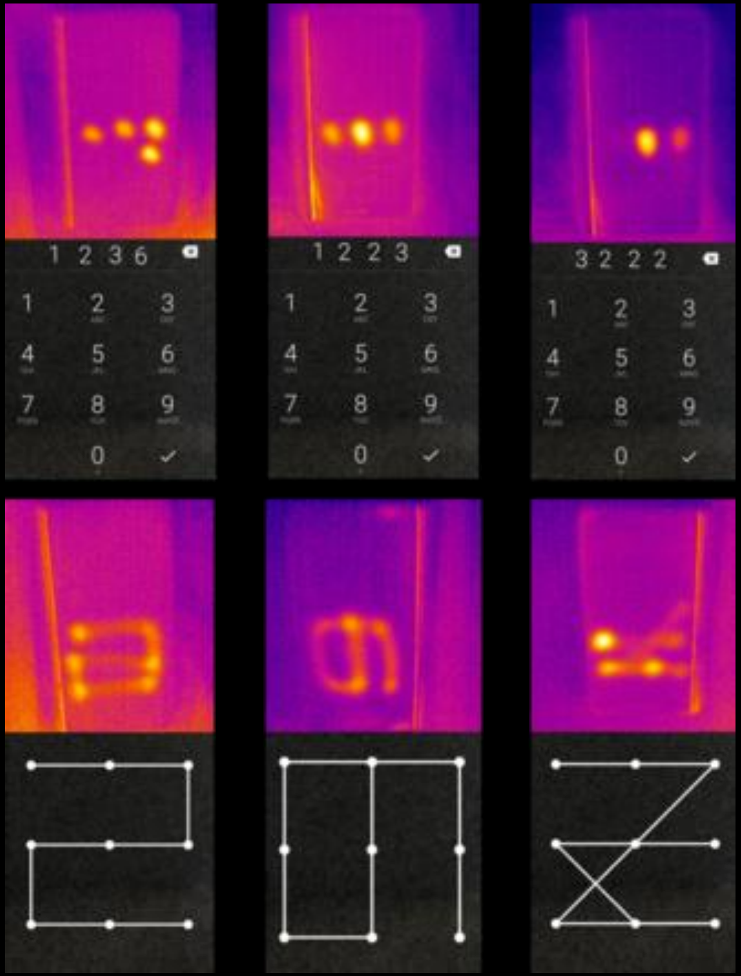
TIMING ATTACK



POWER ANALYSIS (DPA/SPA)

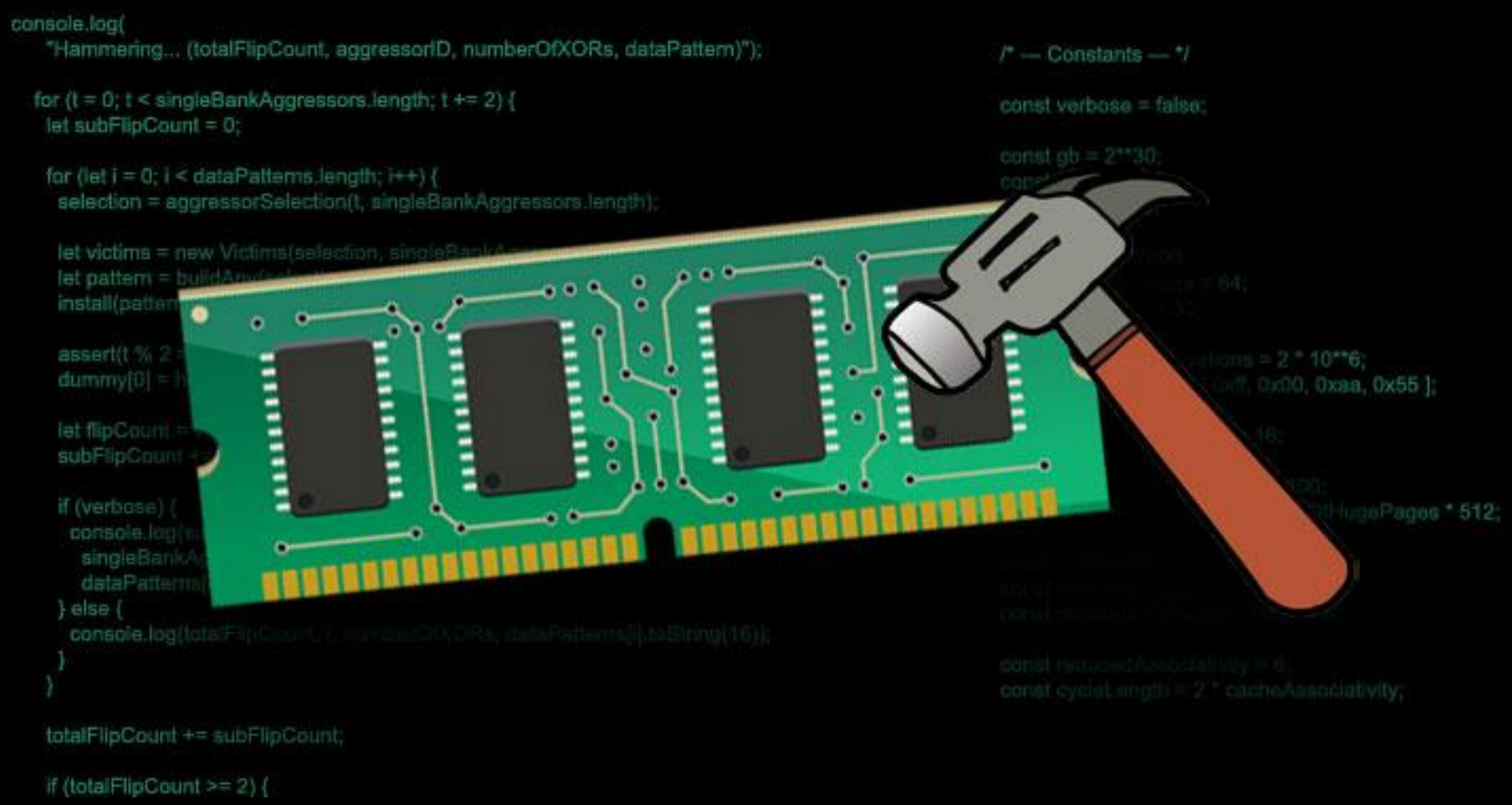


THERMAL ATTACK



ACOUSTIC ATTACK

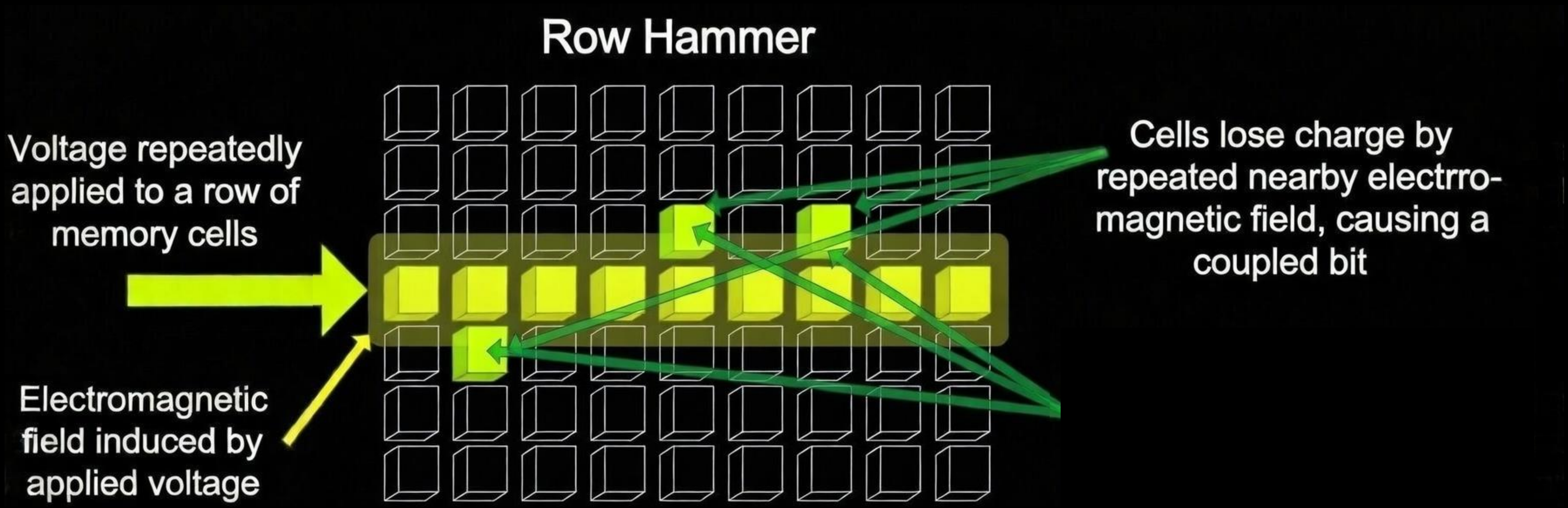
ROW HAMMER ATTACK



```
console.log(
   "Hammering... (totalFlipCount, aggressorID, numberOfXORs, dataPattern)");

  for (t = 0; t < singleBankAggressors.length; t += 2) {
   let subFlipCount = 0;

   for (let i = 0; i < dataPatterns.length; i++) {
    selection = aggressorSelection(t, singleBankAggressors.length);

    let victims = new Victims(selection, singleBankAgg
    let pattern = buildAny(
    install(pattern

    assert(t % 2 =
    dummy[0] = h

    let flipCount =
    subFlipCount +

    if (verbose) {
     console.log(
      singleBankAg
      dataPatterns[
    } else {
     console.log(totalFlipCount, t, numberOfXORs, dataPatterns[i].toString(16));
    }
   }

   totalFlipCount += subFlipCount;

   if (totalFlipCount >= 2) {
```

```
/* — Constants — */

const verbose = false;

const gb = 2**30;
const

                tions
                 = 64;



                stions = 2 * 10**6;
                 0xff, 0x00, 0xaa, 0x55 ];

                    16;

                     100;
                     HugePages * 512;




const reducedAssociativity = 6;
const cycleLength = 2 * cacheAssociativity;
```

ROW HAMMER ATTACK


Row Hammer
Voltage repeatedly applied to a row of memory cells
Electromagnetic field induced by applied voltage
Cells lose charge by repeated nearby electrro-magnetic field, causing a coupled bit