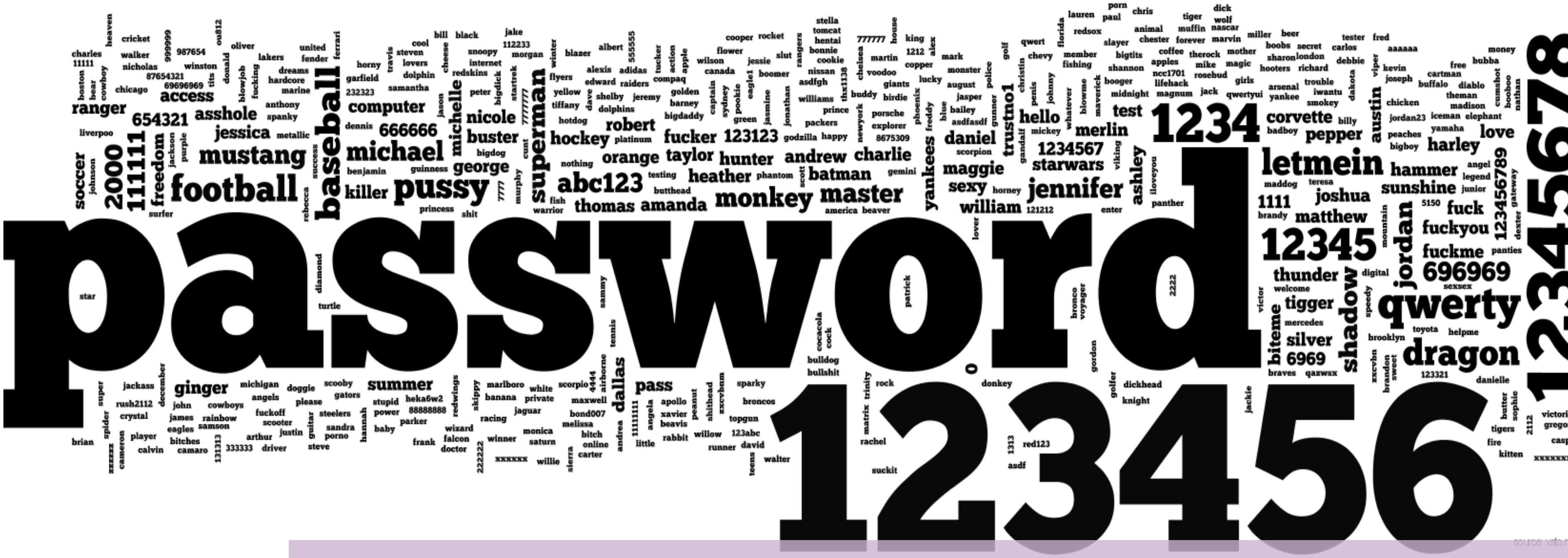


Creation, Authentication and Recovery of Passwords

Dipl.-Ing. Christian Kreuzberger
Institut für Informationstechnologie
Institut für Mathematik

ALPEN-ADRIA UNIVERSITÄT
KLAGENFURT | WIEN GRAZ

FAKULTÄT FÜR TECHNISCHE WISSENSCHAFTEN

Outline

- Introduction
- Master Thesis
- Strength of Passwords
- What we can learn from mistakes

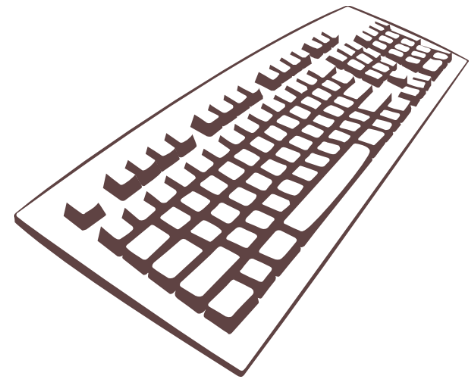
Identification vs. Authentication

- Password – something you **know**
- Chip Card/E-Mail – something you **have**
- Biometrics – something you **are**

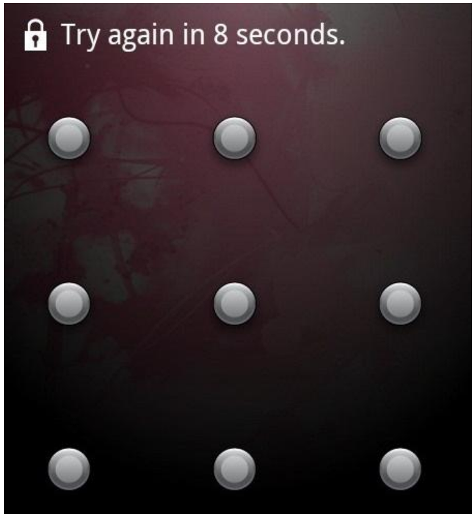
- Password + E-Mail is widely accepted as **Authentication**!
- Better: Combination of all 3

Please enter your password

- Computer
- E-Mail
- Online Banking



- Mobile/Smart Phones
- Buildings/Rooms
- ATMs

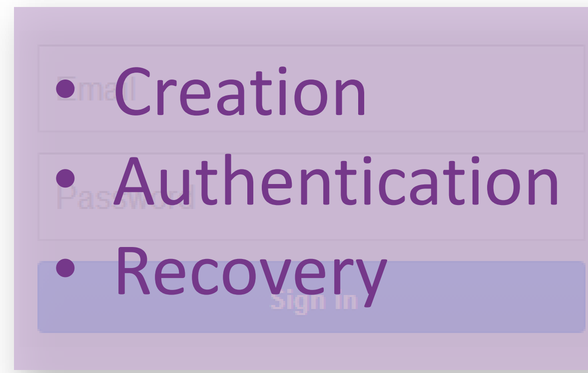


Alternatives

- Iris Scan
- Fingerprint
- Chip Cards
- Gestures
- Images
- Voice Analysis
- …

Problems with Passwords

- Passwords must/should be
  - **easy to remember**,
  - sufficiently **long** and
  - **unique** (do not reuse passwords).

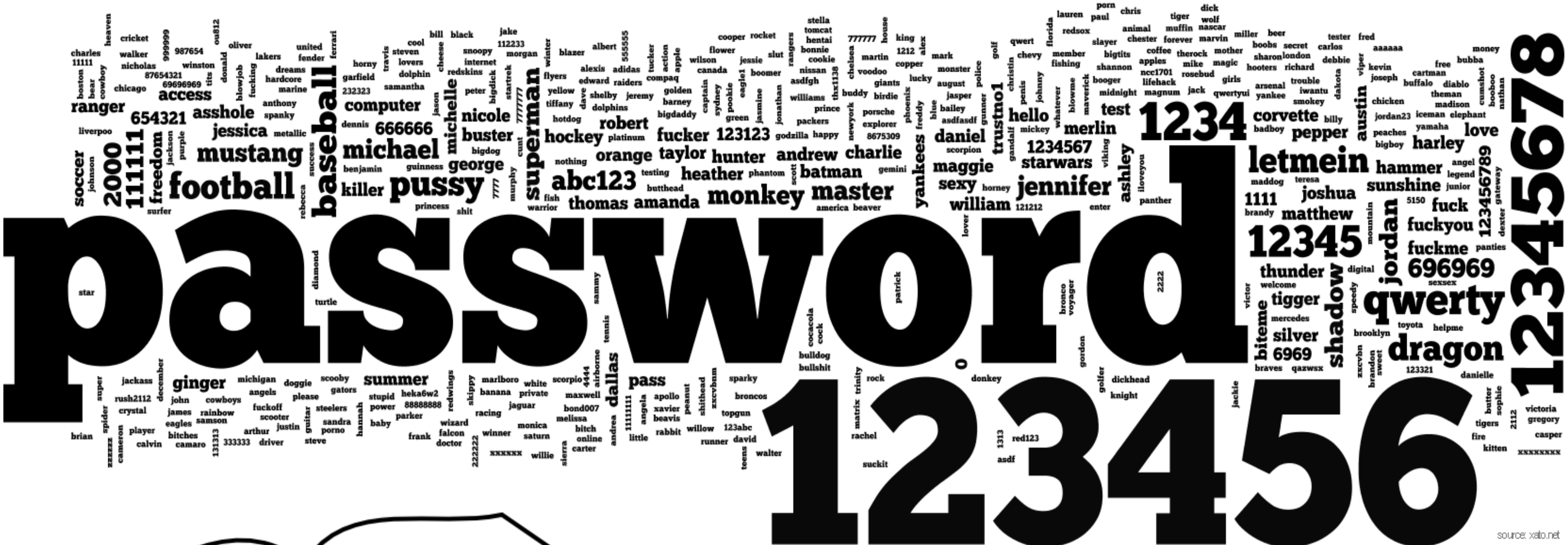


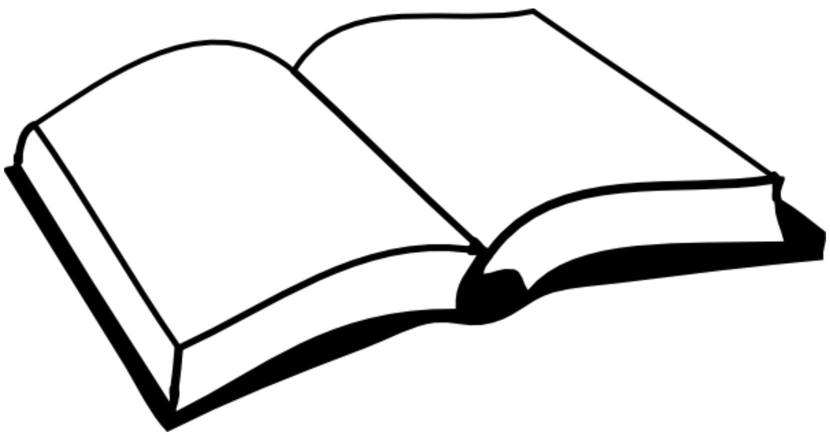
- Creation
- Authentication
- Recovery

- Login-Systems must
  - **create** and **verify** passwords,
  - provide an option to **recover** a forgotten password
  - and **store** and **transmit** passwords in a secure way.

Master Thesis

ALPEN-ADRIA UNIVERSITÄT KLAGENFURT | WIEN GRAZ





Master Thesis

- Intro
  - Strength
  - Creating Passwords (RNG, PUF, KDF), Recovery
- Storing Passwords
  - Websites (Server)
  - Browsers (Client)
  - Operating Systems
  - Chip Cards

Master Thesis

- Attacks
  - Brute Force
  - Dictionaries
  - Rainbow Tables
- Alternatives
  - KeyPass
  - Smartphone + Key Derivation Functions
  - Chipcards

ALPEN-ADRIA UNIVERSITÄT KLAGENFURT | WIEN GRAZ

Strength of passwords

Entropy

Definition 1 (Entropy)**.** *Let N be the size of our alphabet, the amount of different characters we use (e.g. $N = \#\{a, \ldots, z, A, \ldots, Z, 0, 1, \ldots, 9\} = 62$), and L the length in bit of a password we are trying to measure. The Entropy H is given by*

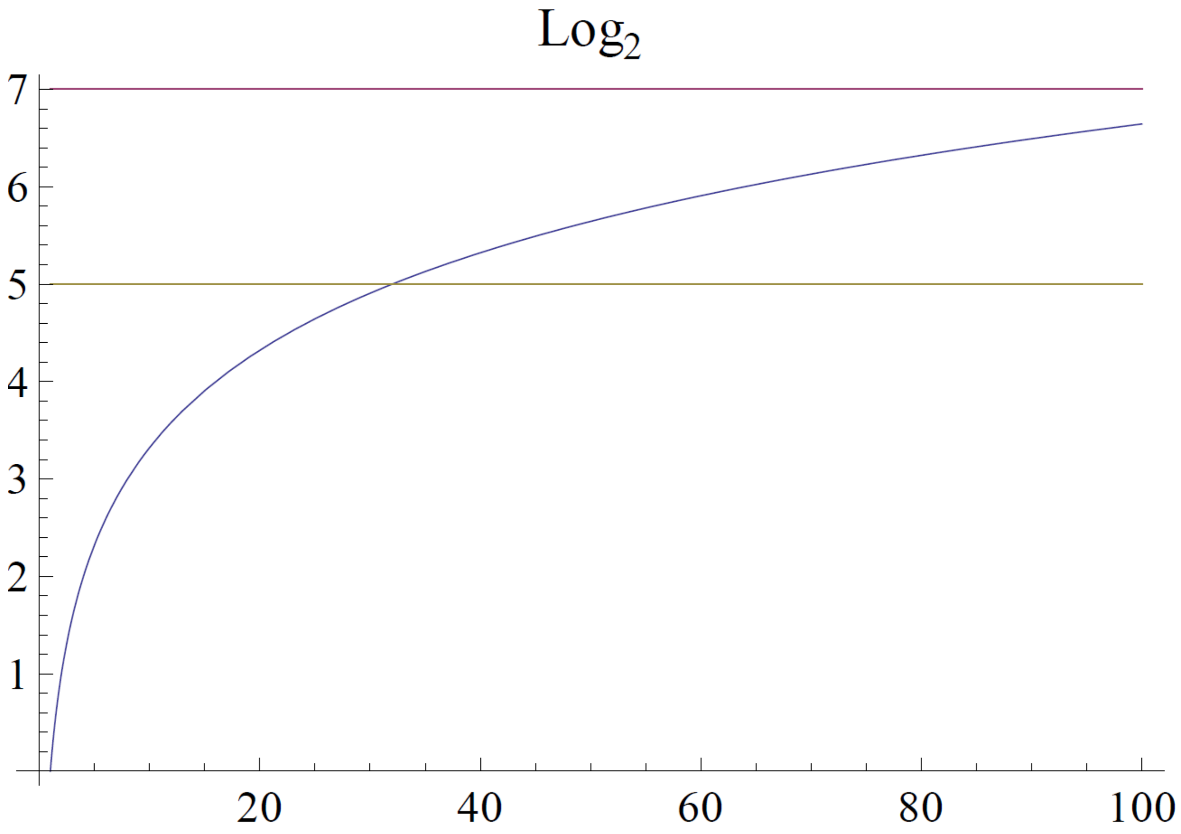
$$H = \log_2 N^L = L \log_2 N \tag{2.1}$$

- Compression: "How many bit do we need to store a string using a limited alphabet"
- Here: "How many bit do we need to guess"
- Common: Alphanumeric Alphabet with

  N = 26 + 26 + 10 = **62 characters**
- ASCII: 95 printable characters (128 total)

Entropy
Length vs Alphabet – ct'ed

- 2^5 = 32
- 2^6 = 64
- 2^7 = 128
- Chinese?



Entropy
Length vs Alphabet – ct'ed 2

- Examples:
  - N=62, L=8: H=47.63 bit
  - N=62, L=12: H=71.45 bit
  - N=84, L=8: H=51.13 bit
  - N=84, L=12: H=76.71 bit
  - N=95, L=8: H=52.56 bit
  - N=95, L=12: H=78.84 bit

$$H = \log_2 N^L = L \log_2 N$$

- Increasing length = increasing security?

NIST
Password strength

- Second approach by NIST: Measuring the strength of a password with rules:
  - First Character: 4 bit
  - Characters 2-8: 2 bit per character
  - **Characters 9-20: 1.5 bit per character**
  - **Above: 1 bit per character**
  - Upper + Lower: +6 bit
  - Dictionary Search: +6 bit
- **Increasing length = increasing redundancy!**

We can learn from mistakes

UNIX Password Generator (1979)

- System supplied "secure" passwords
  - L=8 characters
  - Lower case letters and digits (N=36)
  - Entropy: 41.36 bit (112 years)
- PRNG: 2^15 starting values (Entropy: 15 bit)

`R. Morris, K. Thompson: Password Security: A Case History (Communications of the ACM, Volume 22, 1979)`

What we learned from mistakes

- Use PRNG with a sufficiently large seed space

UNIX Password Store /etc/passwd (197x)

- Username + Password stored in `/etc/passwd`
- Later: `/etc/shadow` + one-way-function
- Everybody on the system could read it

- Everything was fine, until…

```
$> ftp
open target.com
Login: ano@nymous.org
get /etc/passwd
disconnect
```

What we learned from mistakes

- Use PRNG with a sufficiently large seed space

- Use strong(er) one-way functions to store passwords
- NEVER store passwords in plain text
- OS responsible for restricting access to files

Windows Password Store LMHASH (1998)

- Max. 14 OEM-characters
- Input: p' = uppercase(substring(p,0,14))
- If less than 14 bytes, add null-bytes;
- Split password into two halves p' = p1 || p2
- Calculate HASH: h = h1 || h2

$$h_1 = DES(KGS!@\#\$\%, p_1), h_2 = DES(KGS!@\#\$\%, p_2)$$

- Result: 16 byte "hash" value

Windows Password Store LMHASH (1998) – ct'ed

- Max. 14 OEM-characters $Entropy(p) < 83.4 \text{ bit}$
- Input: p' = uppercase(substring(p,0,14))

$$Entropy(p') < 72.4 \text{ bit}$$

- Assuming alphanumeric numbers, we lost 11 bit of entropy, but 72 bit is still a very good result.

Windows Password Store LMHASH (1998) – ct'ed 2

- Split password into two halves p' = p1 || p2
- Calculate "hash": h = h1 || h2

$$h_1 = DES(KGS!@\#\$\%, p_1), h_2 = DES(KGS!@\#\$\%, p_2)$$

- Case 1: Length < 8

h_2 = DES(KGS!@#$%,0x00000000) =
0xAA 0xD3 0xB4 0x35 0xB5 0x14 0x04 0xEE

- Case 2: Length >= 8

$$Entropy(p_1) = Entropy(p_2) \leqslant 7 \log_2 36 = 36.2 \text{ bit}$$

Windows Password Store LMHASH (1998) – ct'ed 3

- Split password into two halves p' = p1 || p2
- Calculate "hash": h = h1 || h2

$$h_1 = DES(KGS!@\#\$\%, p_1), h_2 = DES(KGS!@\#\$\%, p_2)$$

- Case 2: Length >= 8

$$Entropy(p_1) = Entropy(p_2) \leqslant 7\log_2 36 = 36.2 \text{ bit}$$

- Instead of $\log_2(N^{14})$ we now have

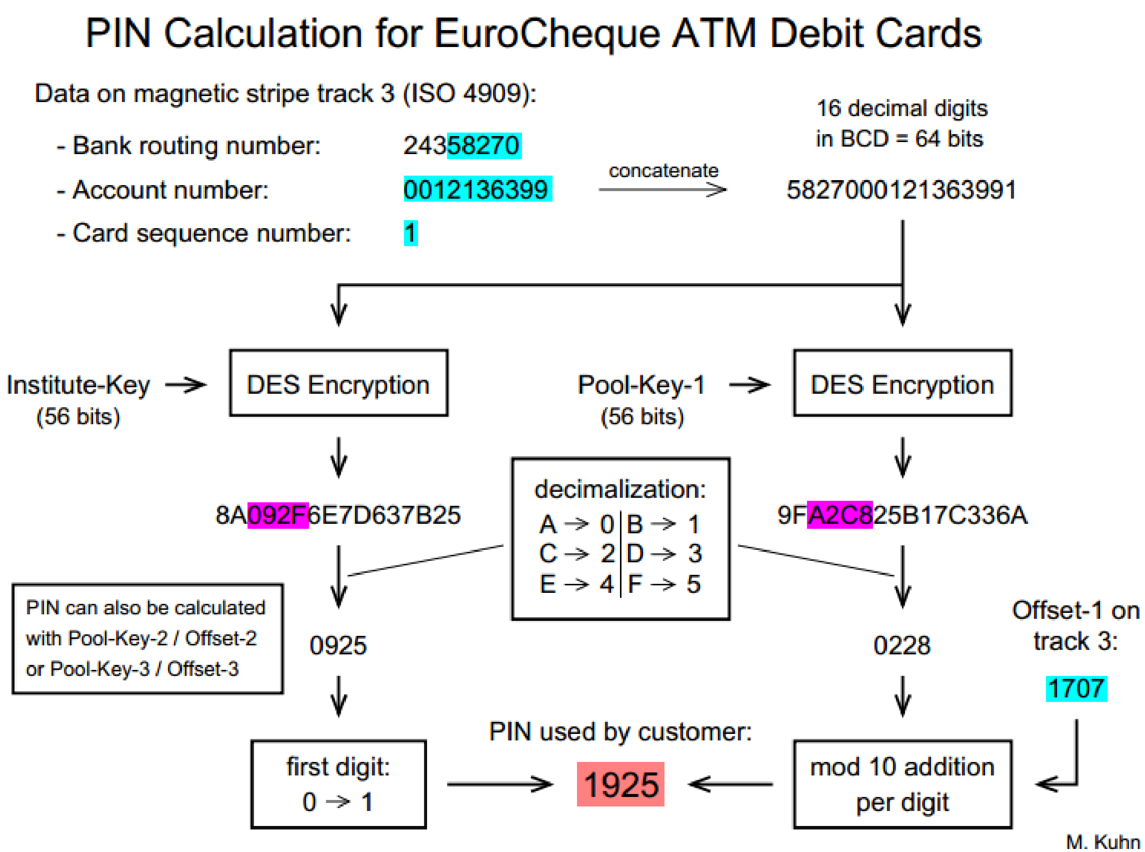
$$\log_2(2 \cdot N^7) = 1 + log_2(N^7) = 1 + 7\log_2 36 = 37.2 \text{ bit}$$

What we learned from mistakes

- Use PRNG with a sufficiently large seed space
- Use strong(er) one-way functions to store passwords
- NEVER store passwords in plain text
- OS responsible for restricting access to files

- Microsoft (20 years later): Use strong(er) one-way functions for authentication

EuroCheque ATM PINs 1981 – 1997 (Germany)



- 1997: M. Kuhn: ***Probability Theory for Pickpockets – ec-PIN guessing***
- Showed that success probability for breaking in can be increased from 0.03 % to 0.7 %

More mistakes…

Input Devices

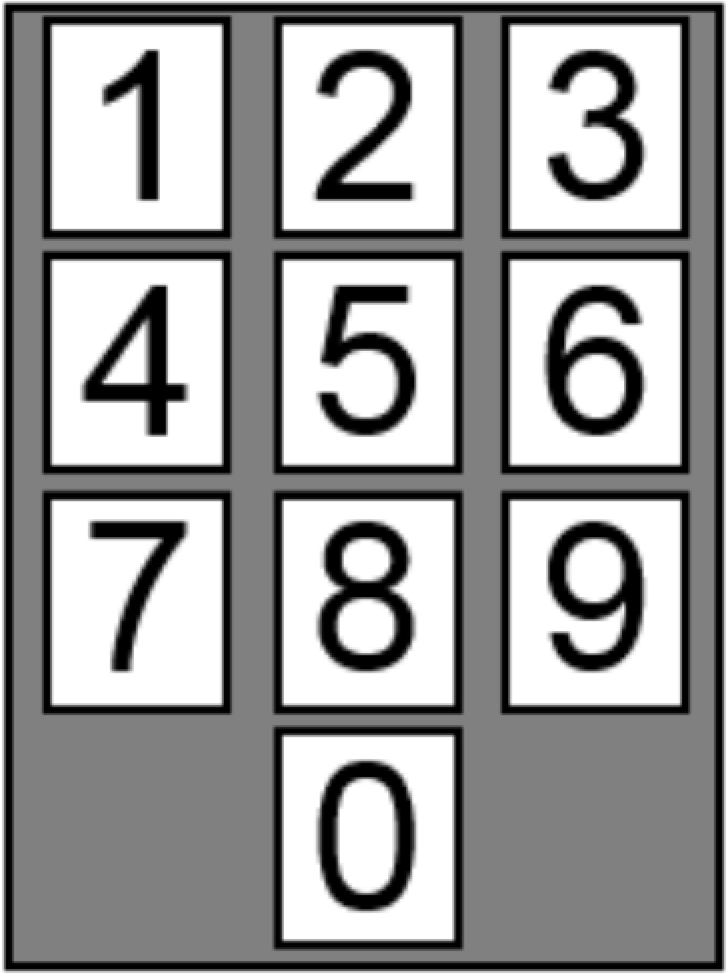
- Smudge Attack
- Thermal Imaging

Most used passwords

#	Password	#	Password
1	password	14	sunshine
2	123456	15	master
3	12345678	16	123123
4	abc123	17	welcome
5	qwerty	18	shadow
6	monkey	19	ashley
7	letmein	20	football
8	dragon	21	jesus
9	111111	22	michael
10	baseball	23	ninja
11	iloveyou	24	mustang
12	trustno1	25	password1
13	1234567		

#	PIN
1	1234
2	0000
3	2580
4	1111
5	5555
6	5683
7	0852
8	2222
9	1212
10	1998

Most used PINs



#	PIN
1	1234
2	0000
3	2580
4	1111
5	5555
6	5683
7	0852
8	2222
9	1212
10	1998

Creation, Authentication and Recovery of Passwords

Dipl.-Ing. Christian Kreuzberger
Institut für Informationstechnologie
Institut für Mathematik

ALPEN-ADRIA UNIVERSITÄT
KLAGENFURT | WIEN GRAZ

FAKULTÄT FÜR TECHNISCHE WISSENSCHAFTEN