Human in the loop: Exploring human vulnerabilities of authentication

Half Day Tutorial by:

Dr Theo Tryfonas MBCS CITP, CISA
Bristol Cryptography Group
Faculty of Engineering University of Bristol Bristol, BS8 1TR, UK
e theo.tryfonas@bristol.ac.uk
w bristol.ac.uk/engineering/people/theo-tryfonas

Agenda

1.30-1.40 Welcome and introductions

1.40-3.10 **Session 1.** "Basic authentication methods, vulnerabilities and user issues"

3.10-3.25 Break

3.25-4.10 **Session 2.** "Pattern Screen-Lock Methods, Security vs. Usability and Soft Side Channel Attacks"

4.10-4.55 **Session 3.** Practical session on the design of a useable lock mechanism for a mobile device

4.55-5.00 Wrap up and close

Human in the loop: Exploring human vulnerabilities of authentication

Session 1. Basic authentication methods, vulnerabilities and user issues

Dr Theo Tryfonas, MBCS CITP, CISA

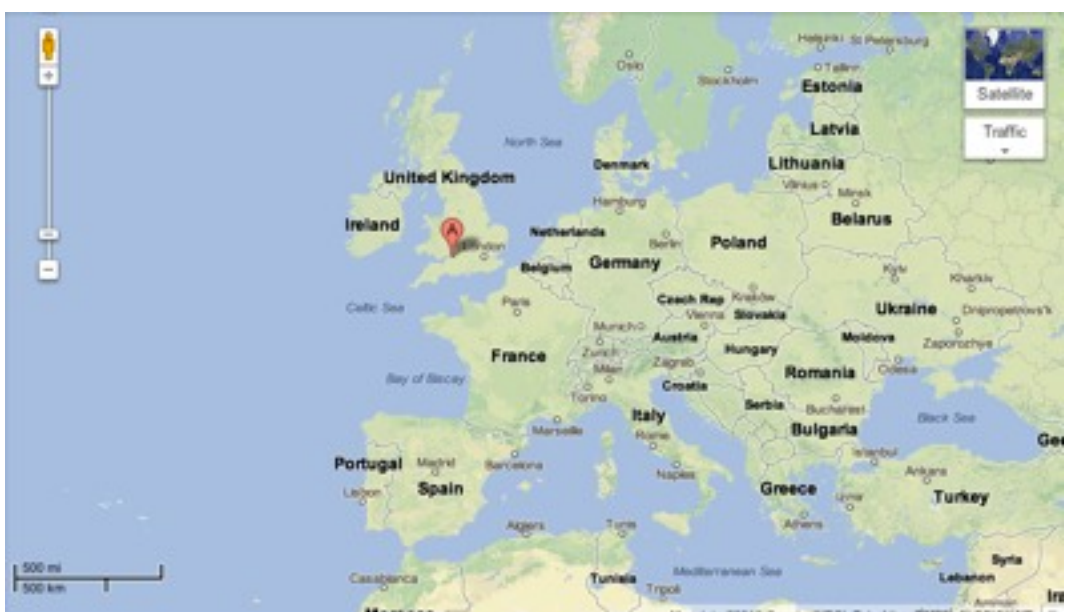
@theotryfonas

HCI Intl 2013 tutorial
Tue., 23rd July 2013

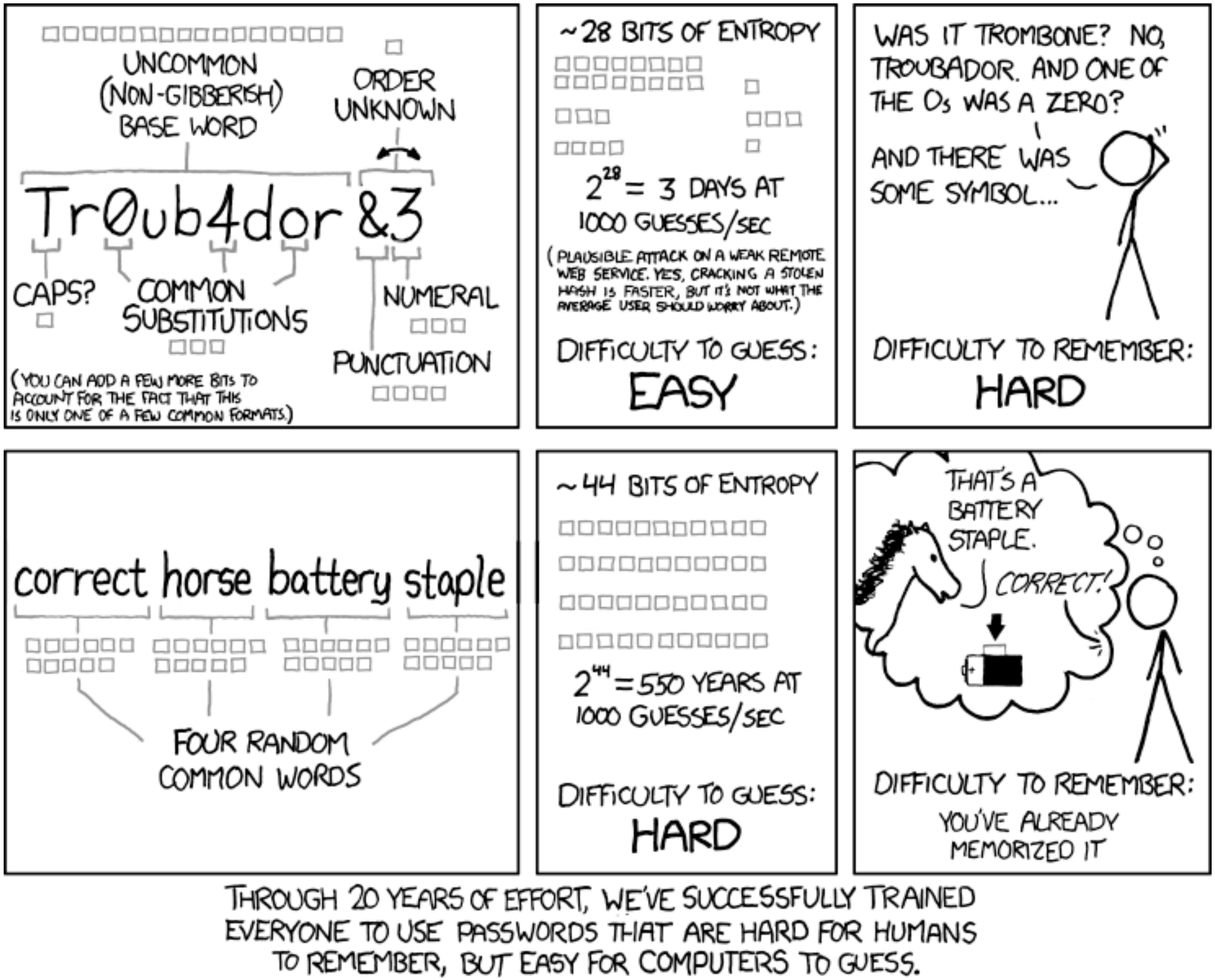
Where do I come from?

University of BRISTOL

CRYPTO



http://xkcd.com/936/



Session outline

- Password schemes and attacks
- Tokens and two-factor authentication
- Biometrics and design challenges
- Authentication controls and their effective integration with an information system

Authentication of users

- A computer bases much of its protection on knowing who its user is - OSs need to make ‘safe’ assumptions about the users of their resources
    - as in real life, e.g. providing an ID to buy alcohol
- **Entity authentication** - the process of verifying the identity claimed by some system entity
    - this can be done in a number of ways

Authentication of users

(cont'd)

- Using something the user *knows* - codeword
- Using something the user *has* - card
- Using something the user *is* - fingerprint
- Using something the user *does* - typing rate

Password-based authentication

- The most popular means of authentication - we should all have extensive experience of use
- Mutually agreed upon codewords assumed to be known only by the user and the system
  - can be user or system set
- Relatively cheap to implement mechanism, comes with almost every OS, software application, web site etc.
- The user provides an identifier and a password and the system verifies the former and compares the latter to a previously known form

Use of passwords

- Password use suffers from various issues, e.g.:
  - Additional burden in use of resources
  - They can be lost or forgotten
  - Need to be maintained in secrecy
  - Need secure bootstraping/initialisation

Password implementation

- Retaining passwords happens essentially through a minimum two-column system table associating IDs with codewords
- Storing user password lists in clear text is not historically unheard of
  - there's an obvious vulnerability there!
- Thus at least the password column was usually retained in encrypted format
  - the system decrypts and compares the provided password - not ideal (what is the vulnerability here?)

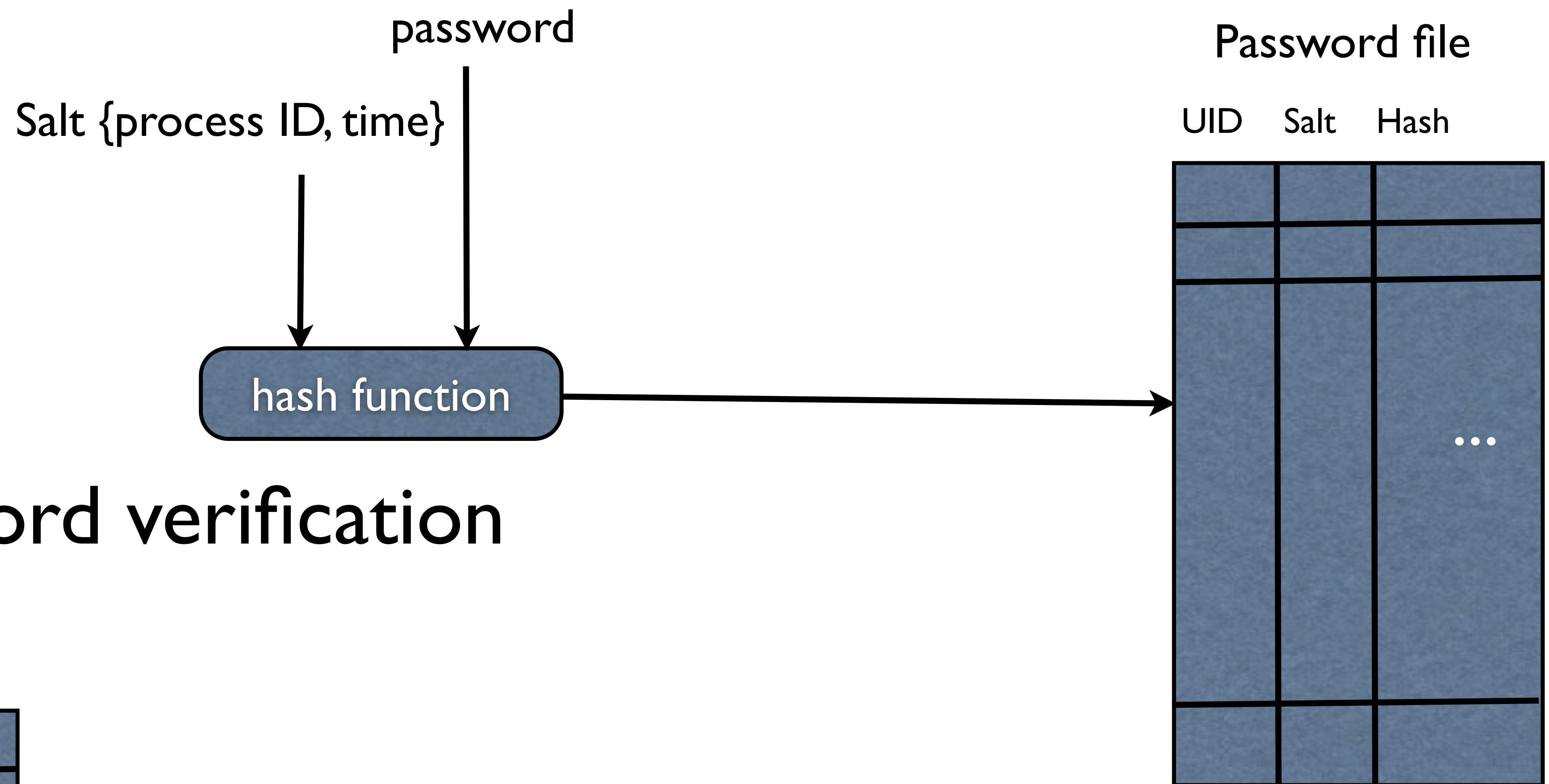
Password implementation
(cont'd)

- A safer approach uses one-way cryptographic hash functions
- *A one-way function is a function that is relatively easy to compute but significantly harder to undo or reverse. That is, given x it is easy to compute $f(x)$, but given $f(x)$ it is hard to compute x*
- There can still be issues
  - users using the same password: $f(x)$ and $f(x')$ will match!
  - user uses the same password across systems

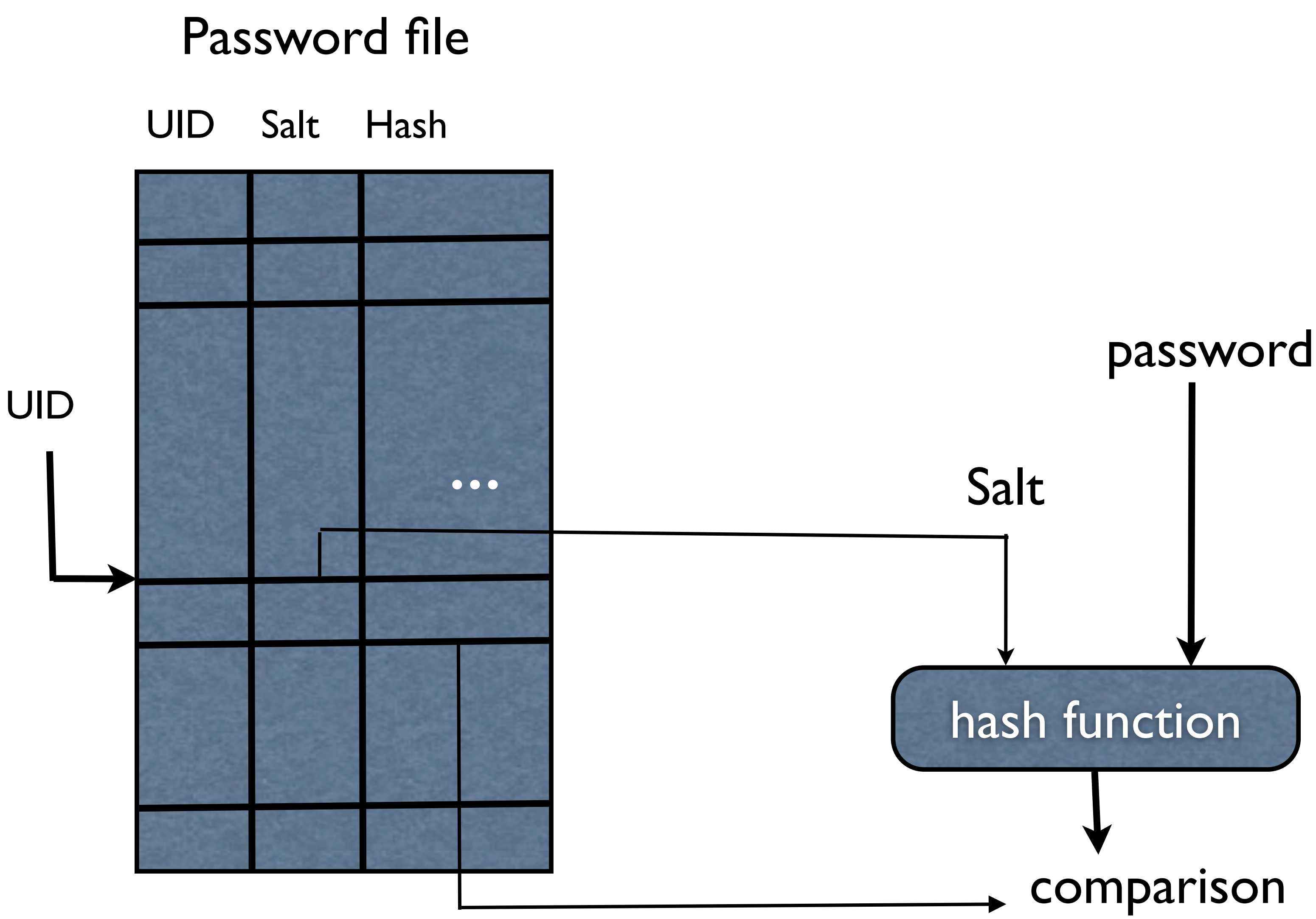
Salt value and traditional implementation

- Originally a 12-bit number derived from process ID and time of password creation
- Original password length restrictions of 8 chars
- Creates a unique entry per password and relieves the pressure of protecting the entire file
  - `/etc/passwd` is world-readable and shell etc. information can be read
  - still vulnerable to a number of attacks though, esp. if weak passwords used - shadow file use `/etc/shadow` accessible only by root

Password generation



Password verification



Modern implementation on UNIX variants

- Hash based on MD5 hash algorithm
- Salt of up to 48 bits
- No limitations on password length
- 128-bit hash value

General attacks on passwords

- Despite the improved security of the previous implementation scheme there is plenty of scope for successful attacks; an attacker may
  - Try all possible passwords;
  - Try frequently used passwords;
  - Try passwords likely for the user;
  - Search for a system list of passwords;
  - Obtain it from the user directly.

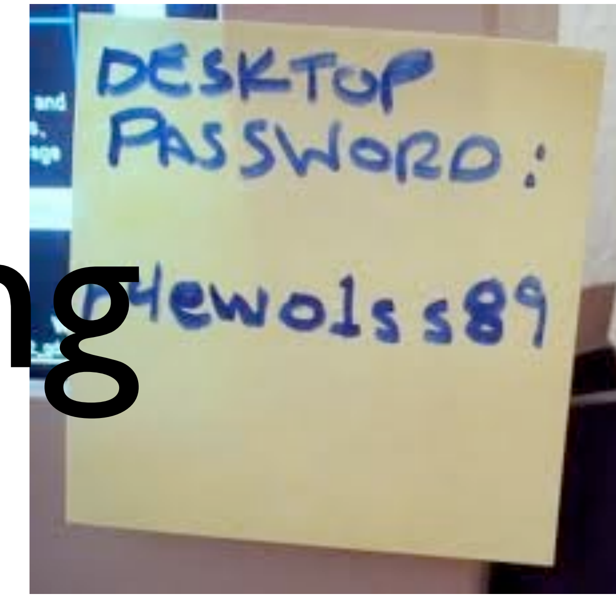
Exhaustive search attacks

- The size of the password space is at least $|A|^n$, where n is the minimal password length and $|A|$ the size of the char set used for password generation
- Assume passwords can be created from 26 chars A-Z of any length 1-8; there are $26^1+26^2+...+26^8=26^9$-$1\approx5*10^{12}$
- 1 passwd/msec gives around 150 years of effort; but 1 passwd/µsec - 2 months
- Assuming that all possible passwords were evenly distributed and we search to recover a single one, not all, then we can expect some success within half the password space

More bad news

- People need to remember their passwords; they may also need to use far too many (short passwords)
    - There are only $26^1+26^2+26^3 = 18,278$ possible passwords, i.e. 18.278sec to go through all; length of 4 or 5 would be 8min and 3.5hr respectively
- We assumed that passwords are generated using chars that are statistically independent - e.g. *vwtxb* is equally probable to *notes* (memorable passwords)
    - People use passwords meaningful to them, even if they appear strong; studies show that people use names, car number plates etc. - trying an entire dictionary of 80k words under our assumptions would take around 80sec

Password guessing



- Based on numerous studies of what sort of passwords we tend to use, how often we change them etc. security testers have identified reasonable ways for reducing the complexity of password recovery - so have hackers
  - no password
  - same as UID
  - is or derived from the user's name (or pet, or car make and model etc.)
  - common word list, plus common names and patterns
  - various dictionaries - not only English!
  - capitalisations and meaningful substitutions e.g. 3 for e, 0 for o etc.
  - brute force, full char set

Space for time tradeoffs

- An alternative to building up dictionaries of possible passwords and using each one to derive hash values combined with each salt etc., is to pre-compute potential hash values
- Trading off space for time is an approach where the attacker generates a large dictionary of possible passwords and for each one the hash values associated with each possible salt value are generated and stored
  - Result: huge table of hash values (rainbow table)
  - Researchers demonstrated a 99.9% crack rate of all alphanumeric Windows password hashes using 1.4GB of data
  - Countermeasure: large salt value and large hash length

A historical example: Windows LM hash

- Users' passwords were restricted to 14 ASCII chars - 95^{14} (about 2^{92})
- Input was converted to all uppercase
- Input was broken up to two 7-byte segments and hashed separately - NO salting
- <8 chars passwords were padded with 0xAAD3B435B51404EE
- See http://ophcrack.sourceforge.net for implementation details

(Some) good password control practices

- Limiting login attempts; lockdown or timeout after each incident
- Password ageing; expiry date with or without memory of previous password hashes to avoid password reuse, within a reasonable time frame, e.g. last 5 used
- Limiting the amount of ‘left-over’, potentially useful to attackers, information; e.g. name of machine, previous logged username on prompt
- Automated password generation - debatable

(Some) good password control practices (cont'd)

- Periodic password auditing
- Providing users with information such as when last logged in etc.
- User education - generate strong passwords, avoid writing them down, sharing them, using the same across systems, etc. etc.
  - Understanding the user perspective and psychology is important, e.g. requesting password changes before holiday periods will probably lead to problems

Vulnerabilities of password schemes

- Obviously storing or sending passwords in the clear is not a good practice...
  - but has historically happened (e.g. the telnet protocol)
- ... but neither is exchanging the corresponding password hashes.
- All vulnerable to eavesdropping and offline guessing attacks.

One-time authentication (challenge-response)

- A one-time password changes every time is used; instead of using a fixed phrase, the system uses a fixed mathematical function, e.g.
  - $f(x) = x+1$: system provides x, user returns $x+1$; $f(x) = p_x$: p_x is x-th prime etc.
  - $f(x) = r(x)$: x is used by a random number generator by both and then comparing results
  - etc. etc.
- This may be obviously onerous for a user, but the intension here is to be generated by devices (tokens)

Security protocols

- A password scheme is in essence a security protocol
- Analysing properties of password schemes as protocols may reveal a lot of security issues
- Example notation (#1) - e.g. for a token to access and open a garage door

$$T \rightarrow G : T, \{T, N\}_{KT}$$

T - token (and token's ID)

G - garage door access mechanism

N - number used once (nonce)

KT - T's encryption key

Challenge-response

- Assume we have an engine controller E and a car key with a transponder T; E sends an n-bit challenge to T using short range radio. In a simple auth. scheme the car key can simply compute a response by encrypting the challenge as below:

$$E \rightarrow T\text{: } N$$

$$T \rightarrow E\text{: } \{T, N\}_K$$

Vulnerability analysis of a simple challenge-response scheme

Consider this simple password-based challenge-response protocol run between a user A and a server S. P_A denotes A's password, n is a random nonce generated by the server, and h is a known cryptographic hash function. The notation (#2) $eK(i)$ means i is encrypted under key K using a known encryption algorithm.

1. $S \rightarrow A$: $eP_A(n)$
2. $A \rightarrow S$: $eP_A(h(n))$

If the attacker intercepts the two messages and offline guesses passwords to use as potential decryption keys, by decrypting both messages one gets two values x and y. If for some candidate password happens to be $y = h(x)$, then the guessed password is likely to be correct.

Man in the middle attacks

- Applies to the on-line challenge-response scheme: say here for simplicity, simple encryption under one's key
- A malicious entity spoofs (i.e. assumes the identity of) a legitimate server and simultaneously opens a connection to a server pretending to be a user
  - Retrieves the challenge N from the server and passes it to the user
  - Retrieves the response $\{N\}_K$ from the user and passes it to the server
  - Hijacks the authenticated session

Multiple factor authentication

- Additional system information may be used to increase confidence of successful user ID and verification
  - e.g. contextual information on where the user connects from, what time etc. or transaction-related data
- An additional form of authentication may be required too
  - e.g. presenting a token, as in the use of cash machines
  - the combination of something the user knows/has is fairly popular

Sample two-factor authentication scheme

- Assume a bank web server S, a user U, a password generator device P and the user's PIN for it (PIN).
- A possible protocol could be

$$S \rightarrow U\colon N$$

$$U \rightarrow P\colon N, PIN$$

$$P \rightarrow U\colon \{N, PIN\}_K$$

$$U \rightarrow S\colon \{N, PIN\}_K$$

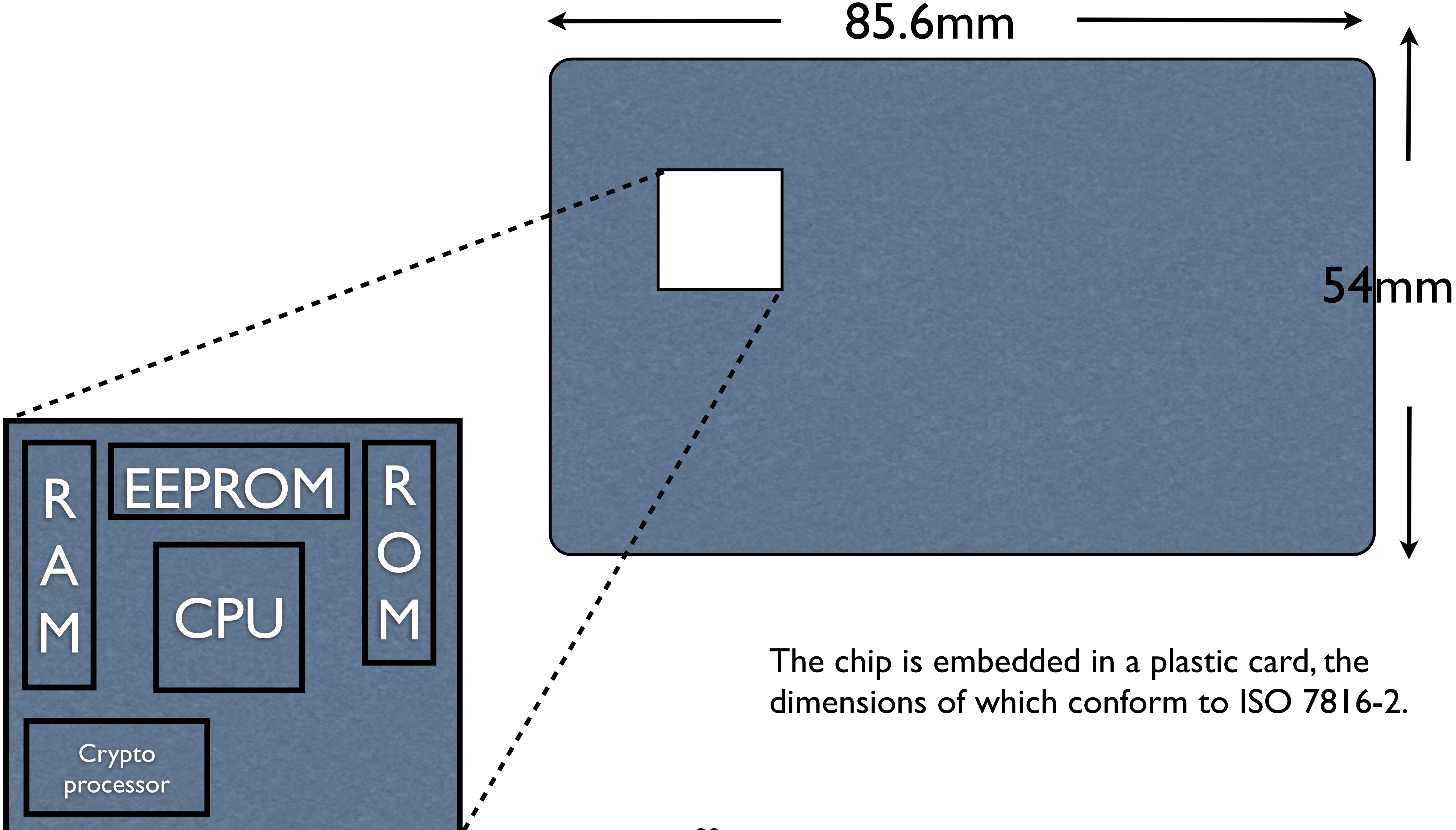
Token-based authentication

- Popular devices that facilitate authentication can be cards (memory or smart) and USB dongles
- Memory cards only hold data in magnetic stripes
- USB dongles is a cheaper alternative to the smartcard and can hold authentication credentials for OSs to verify etc.

Smartcards

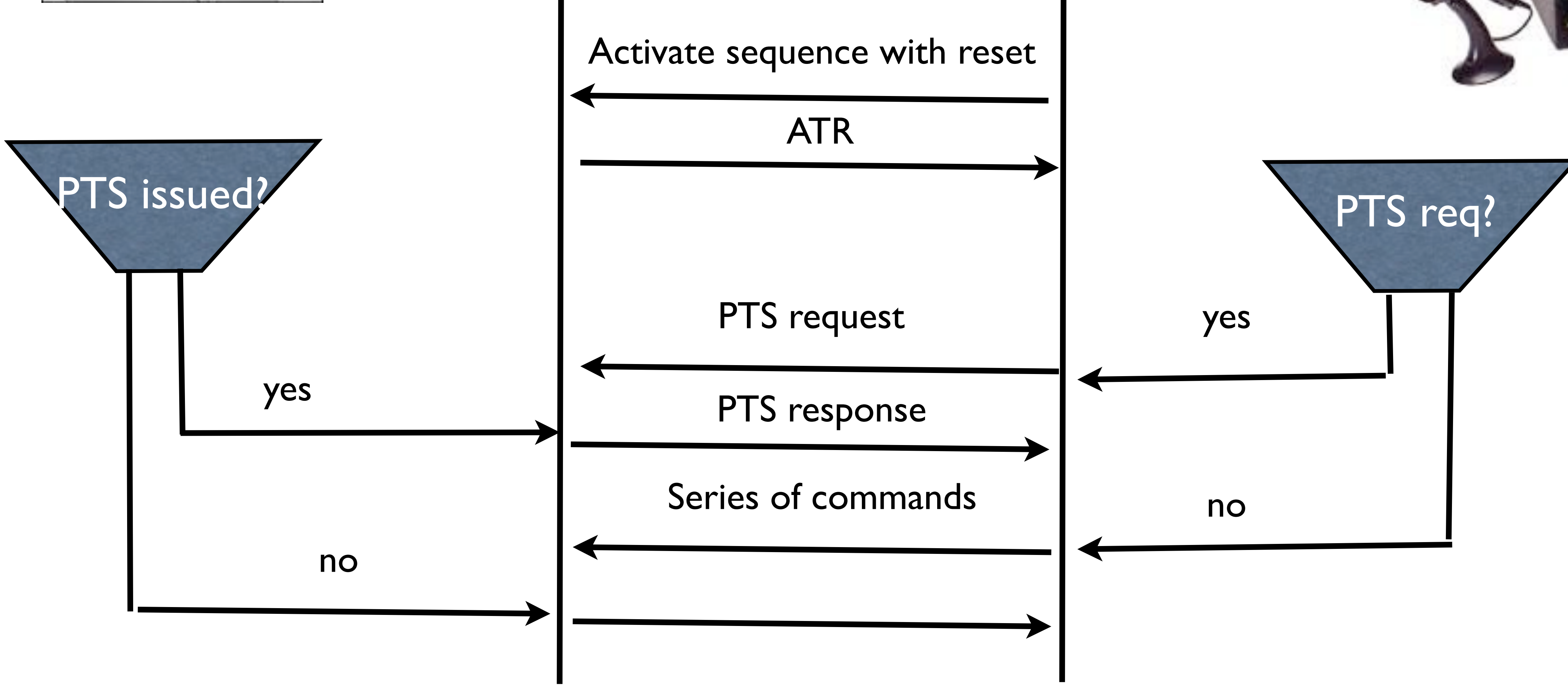
- Include an embedded microprocessor
- Categorised in three groups, based on their implementation of an authentication protocol
  - Static - similar to a memory card
  - Dynamic password generator - the token generates a unique password periodically; synchronisation is required between token and authentication server
  - Challenge-response

Smartcards (cont'd)



The chip is embedded in a plastic card, the dimensions of which conform to ISO 7816-2.

Smartcards (cont'd)



ATR - answer to reset
PTS - Protocol type selection

Single sign on

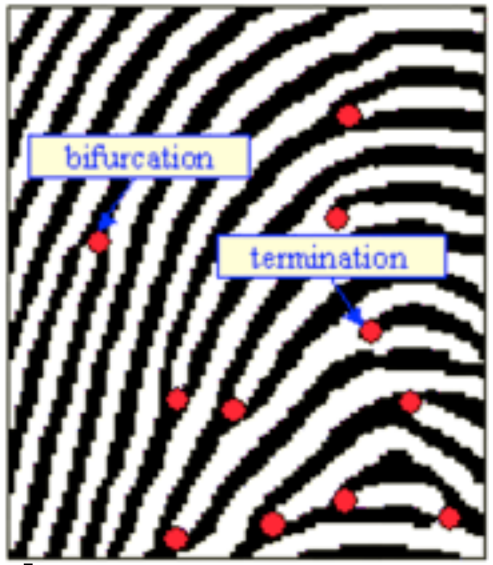
- Multiple authentication requests during a user's interaction experience with a computer can be cumbersome - users may note passwords down, use the same password across systems etc.
- Some systems implement the concept of single-sign on, where the system retains authentication credentials for the user and provides them to subsequent systems or processes as needed
  - Issue: protecting the credentials in storage and during hand-over; web services may achieve it by using cookies or storing them on-line (MS Passport); OSs via encrypted local storage (OS X's Keychain Access app) etc.
  - MyBristol implements some sort of SSO

Biometric authentication

- Schemes that use unique physical characteristics to perform two functions
  - $1{:}n$ identification, i.e. match a person to a database of n people
  - $1{:}1$ verification, i.e. match a given user to their known credentials (e.g. in a token)
- Physical characteristics can be
  - 'hard', e.g. recognition based on hand geometry, fingerprints, face or iris features, or
  - dynamic, such as rate of keystrokes - which has been proved effective in distinguishing users

Fingerprint recognition

- The pattern of the ridges of the fingerprint serves as a unique characteristic
- Samples are collected from users and are stored in analogue or digital form as templates for reference
  - info re shape of curves, locations of bifurcations, positions where ridges end etc. (called *minutiae*)



- For greater accuracy, templates from more than one finger can be obtained

Fingerprint recognition
(cont'd)

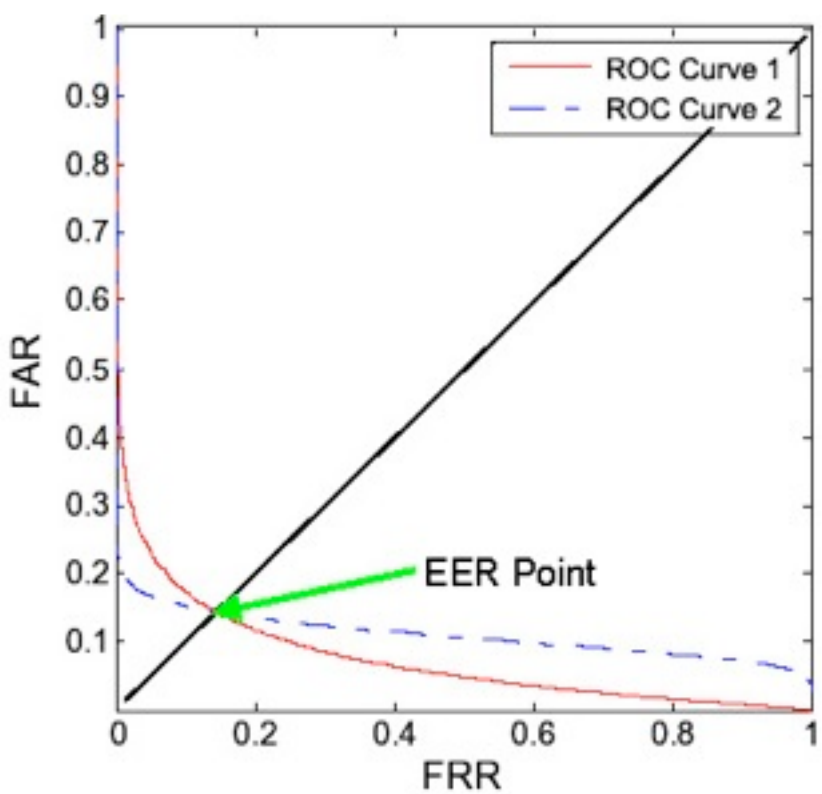
- Registering users' features is called **enrolment;** it is important to note the error rate (failure to enrol, FER) of this process, as user registration may fail due to dirty or worn out fingerprints, reader errors etc.
- When in operation, the system obtains samples from users and tries to match them to known templates; if a match is found a successful authentication has taken place
- The nature of the match is however always within a range - no binary decision as in the case of password use is possible here
  - e.g. reading error due to faulty scanner, dirty surface or fingers, or otherwise worn-out fingertips etc.

Biometrics effectiveness

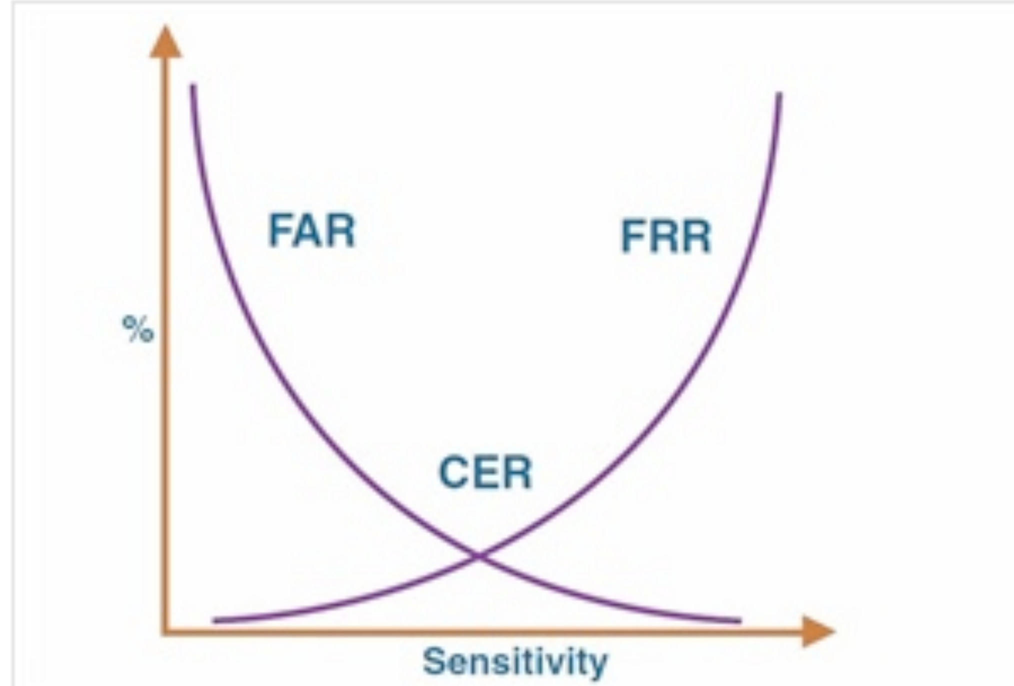
- Due to many reasons there can be
  - false positive identifications (security breach)
  - false negatives (denial of service)
- Various metrics can help us to design and set right levels of acceptance or rejection rates based on the performance of the technology and the security requirements
  - False acceptance rate (FAR) = # false positive identifications/total identifications
  - False rejection rate (FRR) = # false negative identifications/total identifications

Biometrics effectiveness

(cont'd)



Receiver operator characteristics curves



Biometric sensor sensitivity is constantly challenged with keeping both False Acceptance Rates (FAR) and False Rejection Rates (FRR) very low, especially with large enrollee populations. CER is the acronym for Crossover Error Rate.

http://bit.ly/R8cdh3

People issues with use of biometrics

- Perception of criminalisation - fingerprinting
- Physically intrusive - e.g. iris scanners
- Psychologically intrusive - e.g. keystroke monitoring

Recap

- Password schemes and attacks
- Tokens and two-factor authentication
- Biometrics and design challenges
- Authentication controls and their effective integration with an information system

Sources

- D Gollmann Computer Security 3rd Ed. Wiley, 2010 ISBN: 978-0470862933; pp. 49-64 (whole of ch. 4)
- Pfleeger and Pfleeger Security in Computing 4/E Prentice Hall, 2006 ISBN: 978-0132390774; pp. 219-236 (section 4.5)
- R Anderson Security Engineering 2nd Ed. John Wiley & Sons, 2008 ISBN: 978-0470068526; pp. 17-62 (whole of ch. 2)
- B Schneier, The Psychology of Security, http://www.schneier.com/essay-155.html, January 21, 2008

Human in the loop: Exploring human vulnerabilities of authentication

Session 2. Pattern Screen-Lock Methods, Security vs. Usability and Soft Side Channel Attacks

Dr Theo Tryfonas, MBCS CITP, CISA

@theotryfonas

HCI Intl 2013 tutorial
Tue., 23rd July 2013

Session Outline

- Graphical password authentication
  - Android pattern lock mechanism
- Physical attacks
  - Thermal camera to detect swiped pattern heat emission
  - Optical camera, microscope to detect swiped pattern oily residues (smudges)
- Pattern-setting research: security vs. usability perceptions of android users
  - Web-based survey results
  - Physical side-channel attack validation
- Further work

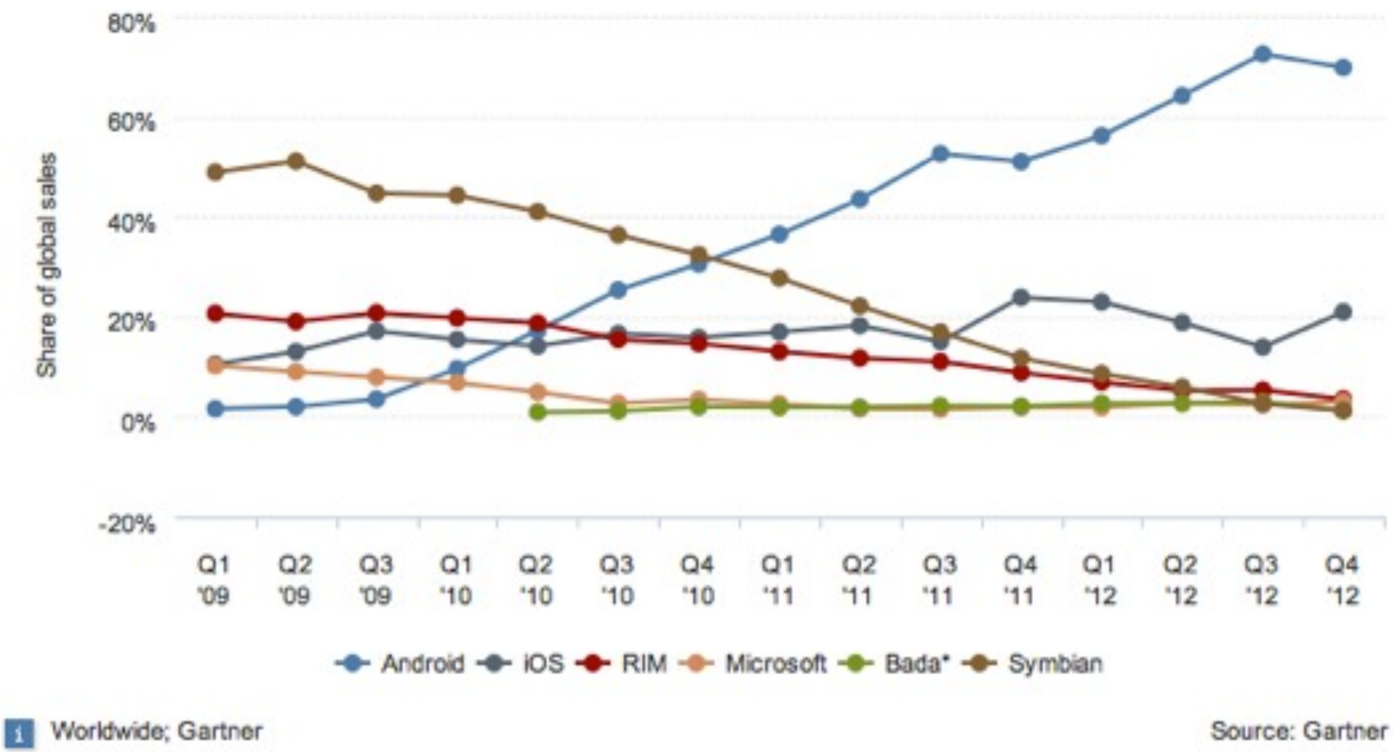
Authentication with graphical passwords

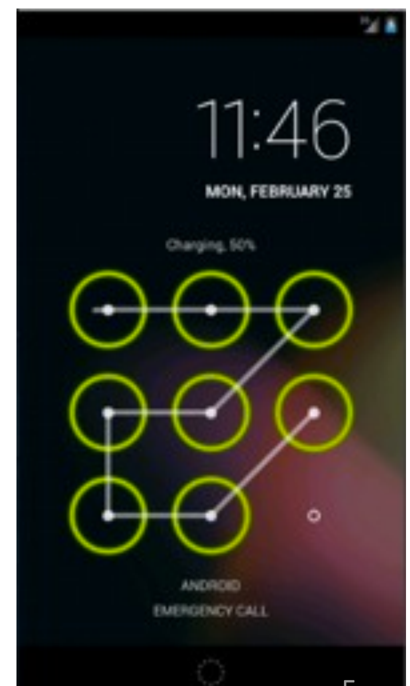
- Existing attacks concentrate on
  - ‘hot spot’ identification (areas of used image concentration)
  - Dictionary style attacks taking into account ‘password’ length, number of components, symmetry

Authentication with graphical passwords (cont'd)

- Studies detected some cognitive biase in choosing graphical passwords
  - as in e.g. the Passfaces system, with attraction and race preference
  - 10% of male passwords were guessable in **two attempts!**

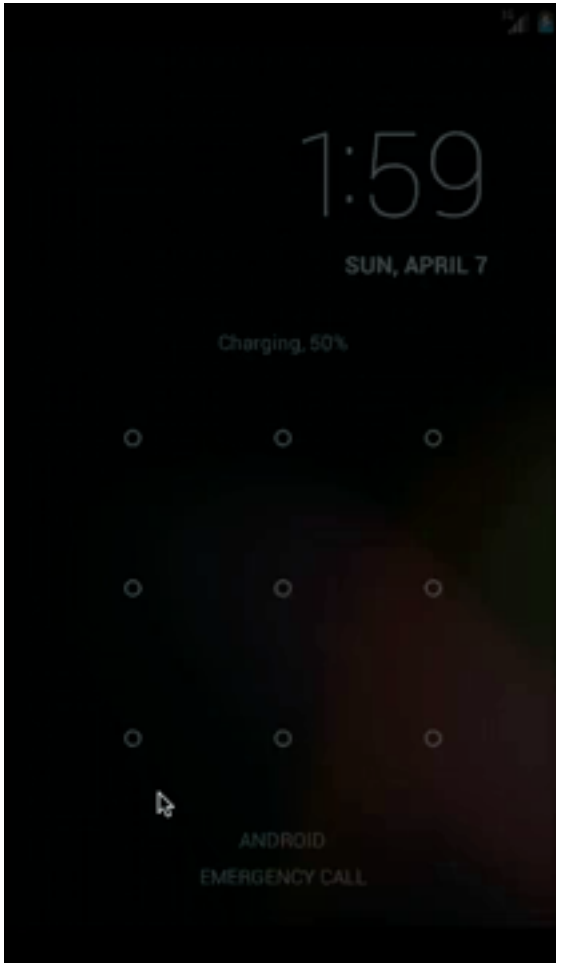
Motivation: Android's popularity and pattern lock mechanism use





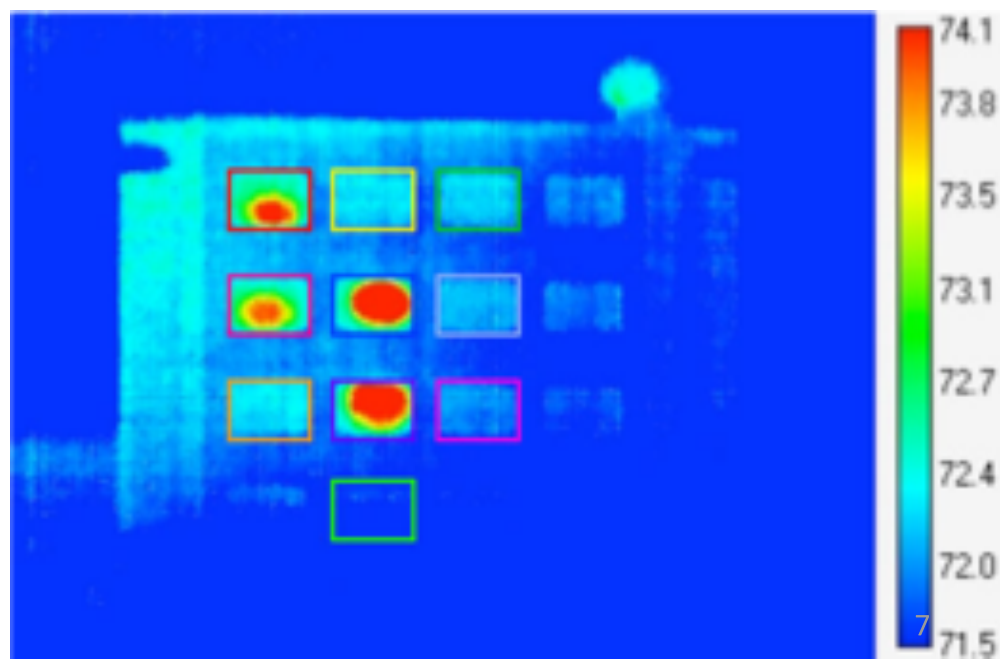
The Android Pattern Lock

- Min 4 and Max 9 nodes to create a pattern.
- Nodes can be visited only once.
- Total number of possible patterns is 389,112.

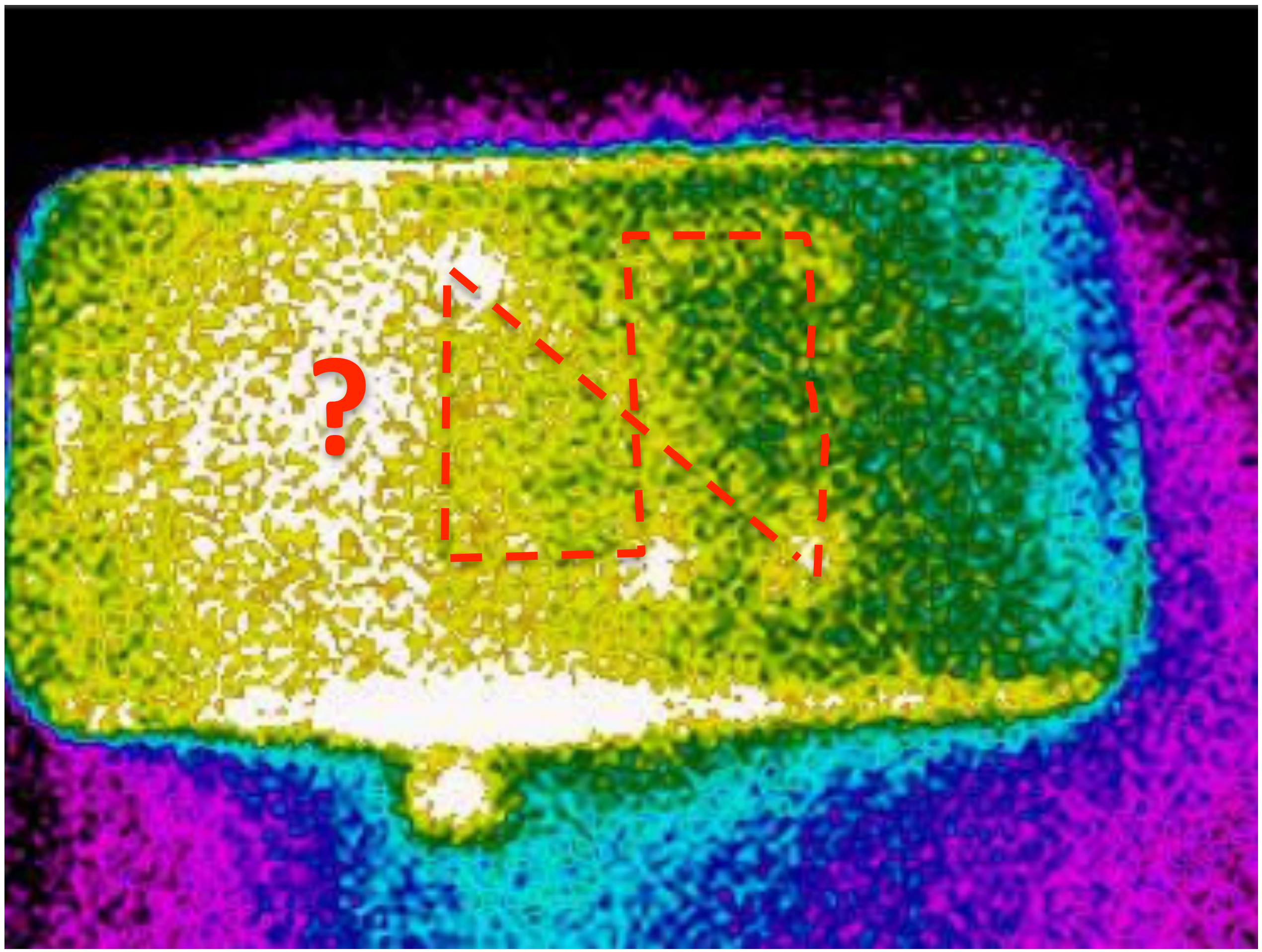


‘Side channel’ attacks on pattern locks

- Attacks based on information gained from the physical implementation of a security scheme are called side channel attacks
  - E.g. existing thermal attacks on ATMs



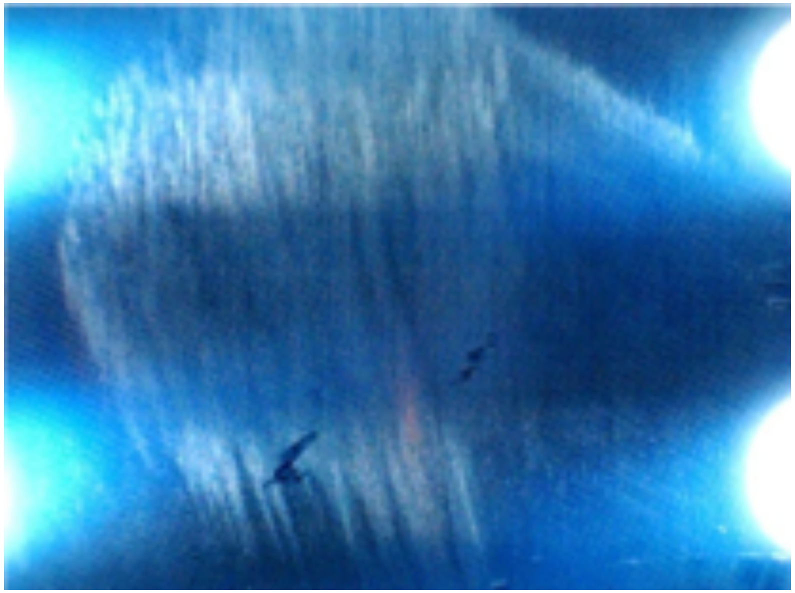
Thermal emission detection



Oily residue detection

Figuring out the swiped pattern

- With a hi res camera
- With a microscope



Detecting directionality

Despite oleophobic coating!

Survey Objectives

- Understand how perceptions of security or usability affect the effectiveness of the mechanism
- Detect biases in the setting of the patterns as graphical passwords
- Facilitate the recovery of locking patterns for forensics and intelligence purposes

Survey instrument

- Done on-line
  - Webpage was live at http://patternsurvey.biz/
- Key questions (pilot) included
  1. Demographics (gender, age)
  2. Experience with smartphones
  3. Use of patterns or not
  4. Asked to set a secure pattern
  5. Asked to set a usable pattern
  6. Preference of pattern between those and why

Data Analysis

- Calculated average pattern lengths
- Calculated average number of direction changes
- Computed entropy per node (frequency metric)
  - i.e. probability of being selected as start or end point or monogram selected in the pattern
- Computed conditional entropy of *n*-grams (Shannon's formula)
  - i.e. most frequently used bi-grams, tri-grams, four-grams (as sub-patterns of swiped paths)

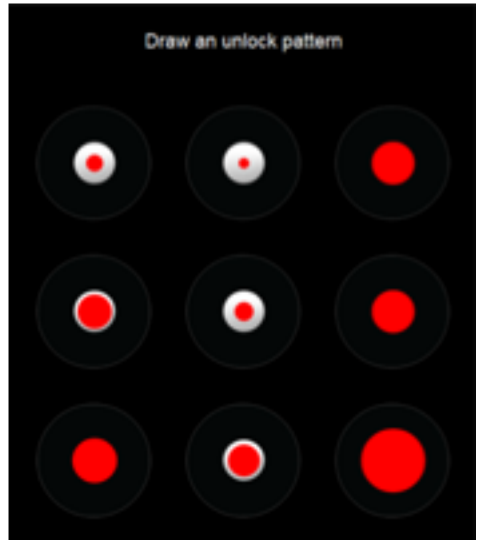
$$F_N = -\sum_{i,j} p(b_i, j) \log_2 p(b_i, j) + \sum_i p(b_i) \log_2 p(b_i)$$

Survey results

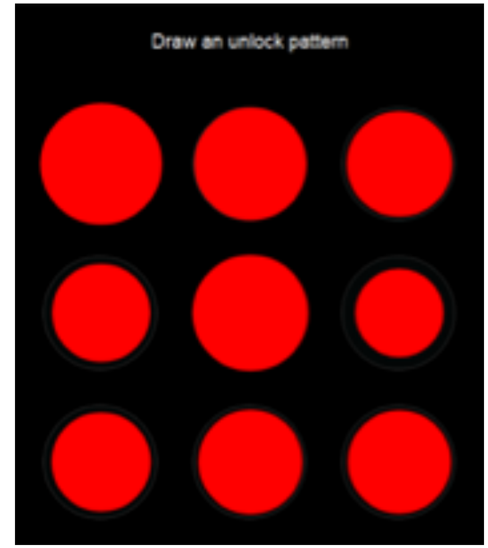
- 144 unique participants
- Gender: **Male** 66%, **Female** 34%
- Age: **18-29** 81%, **30-49** 15%
- 92% own a smartphone of which 40% use Android
- Less than half (47%) use any type of lock, primarily to
  - Protect personal data (secrecy)
  - Prevent fiddling (integrity)
- …

Survey results (cont'd)

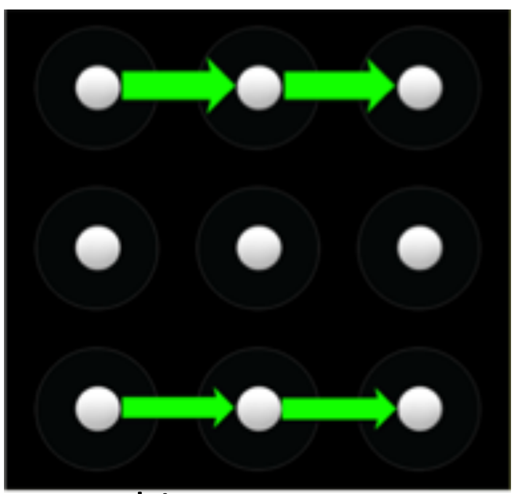
start points



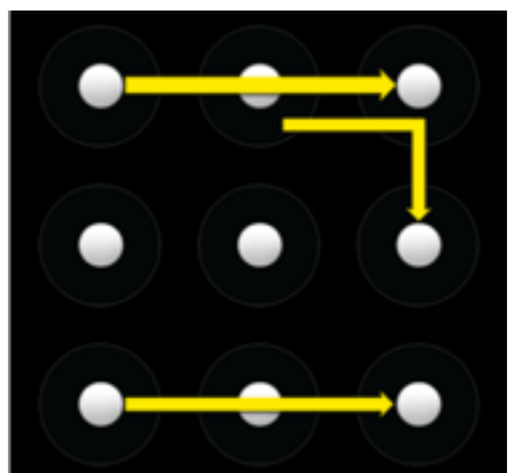
finish points



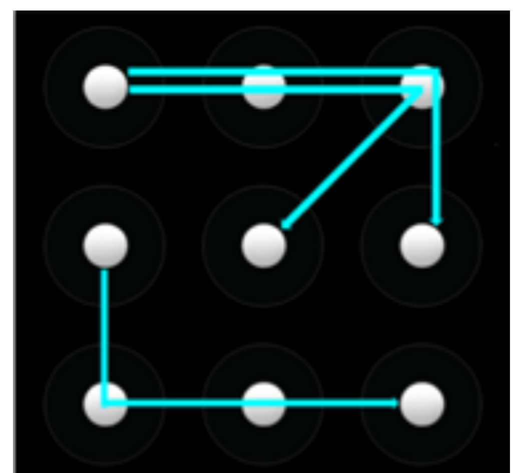
monograms



bi-grams



tri-grams



four-grams

Survey results (cont'd)

Table 1: Average pattern lengths and standard deviations.

Group	Average Length		Standard Deviation	
	Secure	Easy	Secure	Easy
Females	6.16	5.94	1.87	1.75
Males	6.89	6.32	1.91	1.94
Total	6.64	6.19	1.92	1.88

Table 2: Average number of direction changes (all users).

Average Changes		Standard Deviation	
Secure	Easy	Secure	Easy
3.57	2.74	1.65	1.59

Preliminary validation: performing side channel attacks (physical/behavioral)

- 22 participants:
  - Male: 68%, Female: 32%
- Origin:
  - Europe: 59%, Asia: 32%, America: 9%.
- Apply a secure pattern lock on device.
- Take photo with DSLR camera.

Preliminary validation (cont'd)

Optical Attack	Number	Percentage
0 - 49% of pattern	5/22	22.73%
50 - 99% of pattern	5/22	22.73%
100% of pattern	12/22	54.54%
Total Recovery	18/22	81.82%

Phychological	Number	Percentage
Start point	18/22	81.82%
End point	11/22	50.00%
Bigrams	12/22	54.54%
Trigrams	7/22	31.81%
Fourgrams	4/22	18.18%
Direction (C)	14/22	63.63%
Total Retrieval	20/22	90.9%

Further work

- Extended data set
- Add more detailed demographics (mother tongue, dexterity, location)
- Further analytics (e.g. symmetry detection, other cognitive biases)
- Validate the gender bias observation (over ⅓rd of the pilot sample were women)
- Link with decision-making theory (e.g. prospect theory) to develop profiles of pattern preferences per decision-making type (suspect type)

Thank You

Any Questions?

Theo.Tryfonas@bristol.ac.uk

Panagiotis Andriotis, Theo Tryfonas, George Oikonomou, Can Yildiz.
"A Pilot Study on the Security of Pattern Screen-Lock Methods and Soft Side Channel Attacks. "

Security and Privacy in Wireless and Mobile Networks - WiSec 13, ACM, pp. 1-6, 2013.

This work has been supported by the European Union's Prevention of and Fight against Crime Programme "Illegal Use of Internet" - ISEC 2010 Action Grants, grant ref. HOME/2010/ISEC/AG/INT-002

Sources

- Aviv, et al. Smudge attacks on smartphone touch screens. In Proceedings of the 4th USENIX conference on Offensive technologies, pages 1–7. USENIX Association, August 2010.
- Mowery et al. Heat of the moment: characterizing the efficacy of thermal camera-based attacks. In Proceedings of the 5th USENIX conference on Offensive technologies, pages 6–6. USENIX Association, August 2011.
- Oorschot et al. On predictive models and user-drawn graphical passwords. ACM Trans. Inf. Syst. Secur., 10(4):5:1–5:33, January 2008
- Thorpe et al. Human-seeded attacks and exploiting hot-spots in graphical passwords. In USENIX Assosiation Proceedings of the 16th USENIX Security Symposium, pages 103–118. USENIX Association, August 2007.

Human in the loop: Exploring human vulnerabilities of authentication

Session 3. Practical session on the design of a useable lock mechanism for a mobile device

Dr Theo Tryfonas, MBCS CITP, CISA

@theotryfonas

HCI Intl 2013 tutorial
Tue., 23rd July 2013

Practical Session Brief

Objective:

- Given what you now know about potential attacks on the pattern lock mechanism, devise a graphical method for authenticating a user to a mobile device that would be easy to use and perceivably secure

Requirements:

- Work in a group of four (or two groups of three if applicable)
- Use rich pictures, story-boarding, UML, GUI sketches, simple conceptual diagrams or whatever else method you are familiar with for capturing requirements
- Demonstrate explicitly where and how you take into account specific usability and security concerns
- You may refer to Andriotis et al. and Il Shin et al.