The Quest for Usable Security

N. Asokan
Aalto University and University of Helsinki

Slides available at
http://asokan.org/asokan/research/talks.php

Aalto University

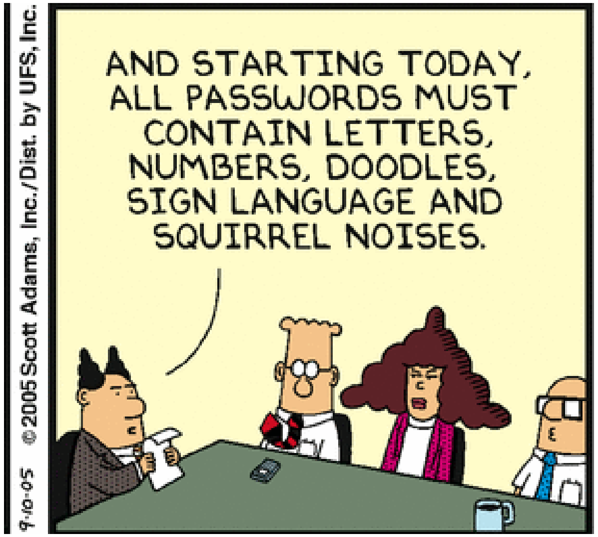
UNIVERSITY OF HELSINKI

http://dilbert.com/strip/2007-11-16

Why worry about usability?

Lack of security usability

- Harms security, eventually
- Lowers overall attractiveness of the device/service, eventually
- Costs money!



http://dilbert.com/strip/2005-09-10

Outline

- Two case studies
  - Secure First Connect
  - Pitfalls in designing zero-effort deauthentication
- Examples of other usable security problems

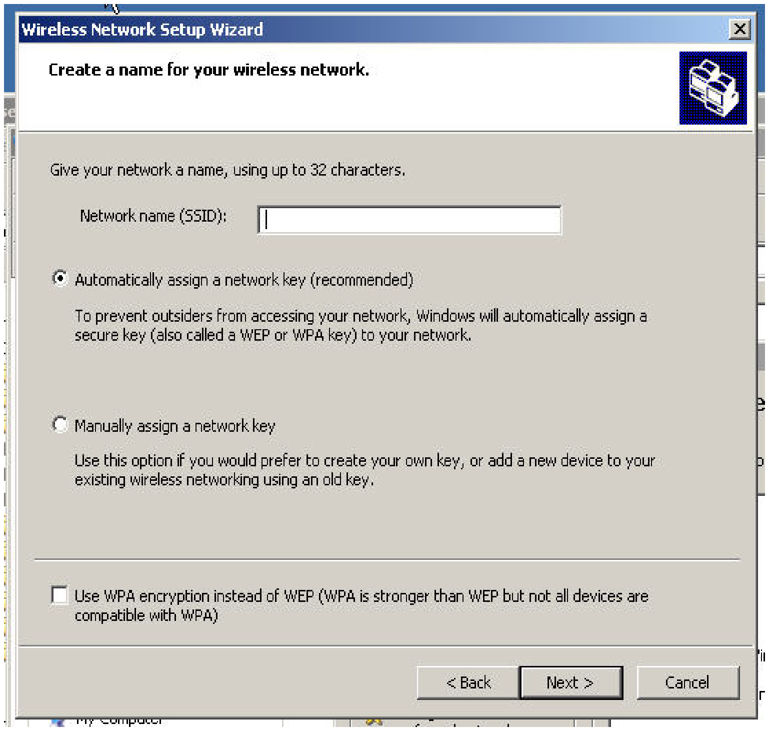
(focusing on mobile devices/users)

Secure First Connect

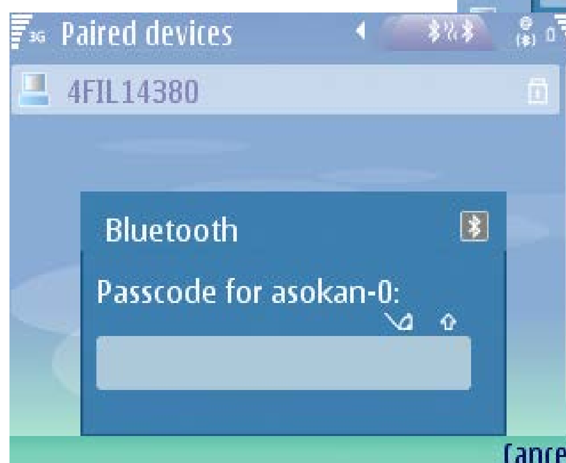
Setting up the first connection

- **First Connect**: setting up contexts for subsequent communication.
  - Typically for proximity communications between personal devices, e.g.:
    - Pairing a Bluetooth phone and headset
    - Enrolling a Phone or PC to a home WiFi network
- **Problem (circa 2006)**: Secure First Connect for personal devices
  - Initializing security associations (as securely as possible)
  - No security infrastructure (no PKI, key servers etc.)
  - Ordinary non-expert users
  - Cost-sensitive commodity devices

Prevalent mechanisms were not intuitive



SSID? WPA?
Passcode?



… and not very secure

Cracking the Bluetooth PIN*

Yaniv Shaked and Avishai Wool

School of Electrical E
Tel Aviv University, Ram
shakedy@eng.tau.ac.il,

Abstract

This paper describes the implementation of an attack on the Bluetooth security mechanism. Specifically, we de-

Security Weaknesses in Bluetooth

Markus Jakobsson and Susanne Wetzel

Lucent Technologies - Bell Labs
Information Sciences Research Center
Murray Hill, NJ 07974
USA
{markusj,sgwetzel}@research.bell-labs.com

Abstract. We point to three types of potential vulnerabilities in the Bluetooth standard, version 1.0B. The first vulnerability opens up the system to an attack in which an adversary under certain circumstances is able to determine the key exchanged by two victim devices, making

Naïve usability measures damage security

http://www.helsinki-hs.net/news.asp?id=20030930IE16

HELSINGIN SANOMAT
INTERNATIONAL EDITION

TODAY | THIS WEEK | WEBORTAGE | THIS IS

Consumer - Tuesday 30.9.2003

Pictures taken with mobile phone showed up on neighbour's TV

▶ Default password must be changed when starting to use Bluetooth-equipped devices; read the manual!

elsewhere as well. It is, therefore, absolutely essential that the password is changed immediately when the device is first installed."

"This is clearly printed in the user's manual", Rosenberg points out. How often have we heard *that* before?

"Once the digital receiver's password has been changed, the new password also has to be entered in the transmitting device, in this

Naïve security erodes usability

Pairing

To create a connection using Bluetooth wireless technology, you must exchange Bluetooth passcodes with the device you are connecting to for the first time for reasons of security. This operation is called pairing. The Bluetooth passcode is a 1- to 16-character numeric code, which you must enter in both devices. You only need this passcode once.

SIM access mode

In SIM access mode, if the car kit finds a compatible mobile phone that supports the Bluetooth SIM access profile standard, the car kit shows a randomly chosen, 16-character numeric code on the display, which you must enter on the compatible mobile phone to be paired with the car kit. Note that you must be prepared to do this quickly within 30 seconds. Follow the instructions on the display of your mobile phone.

If pairing is successful, Paired with, followed by the name of your mobile phone is displayed. Then Create connection is displayed. Press to establish the Bluetooth wireless connection.

Note
When pairing a mobile phone in SIM access mode, a 16-character numeric passcode is generated in the car kit. You can delete this passcode if desired: within 3 seconds, press to delete the Bluetooth passcode. Then enter an arbitrary 16-character numeric code into the car kit using the Navi wheel number editor.

Car kits

- Allow hands-free phone usage in cars
- Retrieve/use session keys from phone SIM
- require higher level of security

➢ users must enter 16-character passcodes

More secure = Harder to use?

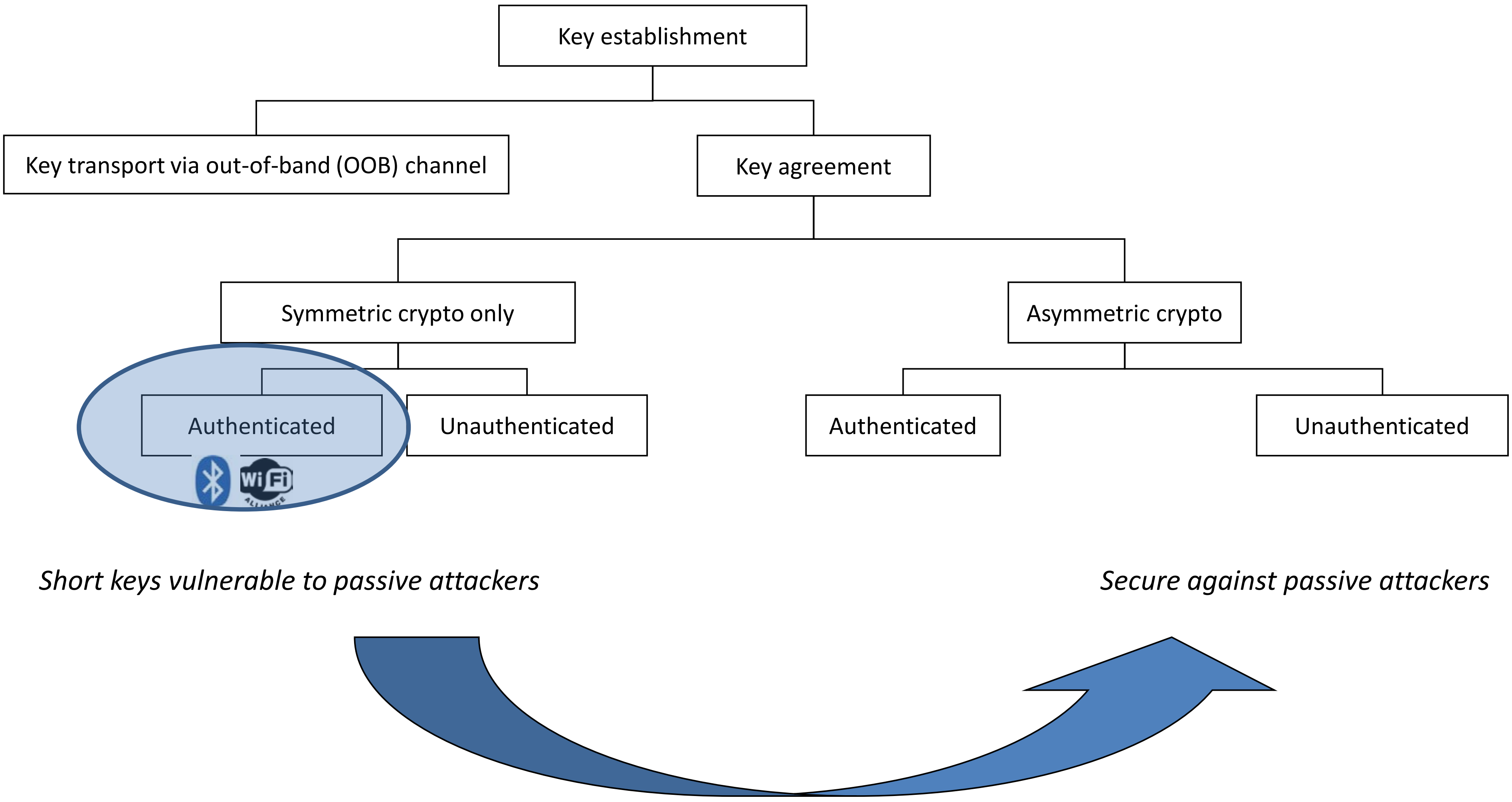
Cost:
Calls to Customer Support

Wanted: intuitive, inexpensive, secure first connect

- Two (initial) problems to solve
  - Peer discovery: finding the other device
  - **Authenticated key establishment**: setting up a security association

- Assumption: Peer devices are physically identifiable

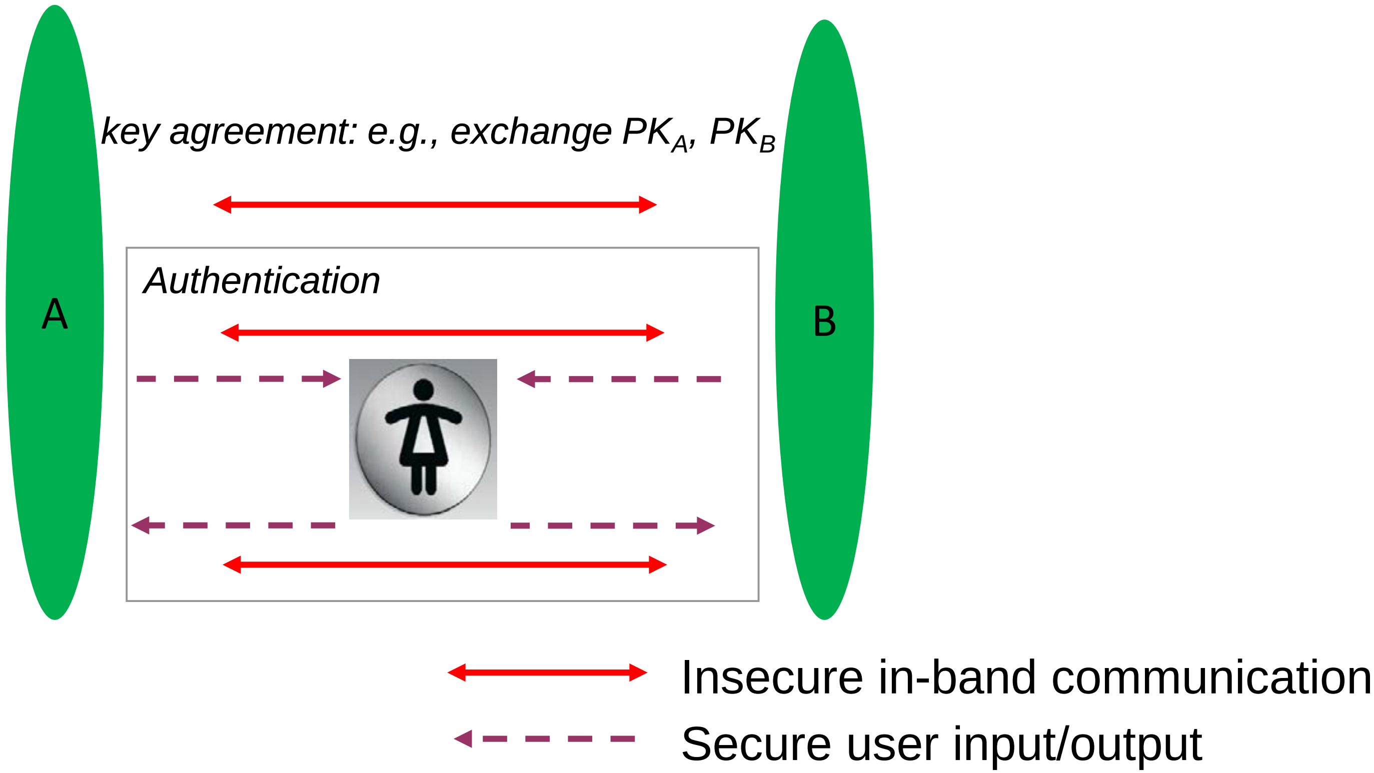
Key establishment for first connect ~2006



Authenticating key agreement

- Use an auxiliary (OOB) channel to transfer information needed for authentication
- Two possibilities for secure auxiliary channel
  - User assistance
  - Other OOB secure communication channels
    - E.g., Near Field Communication, infrared, …

Authenticating key agreement: user-assisted



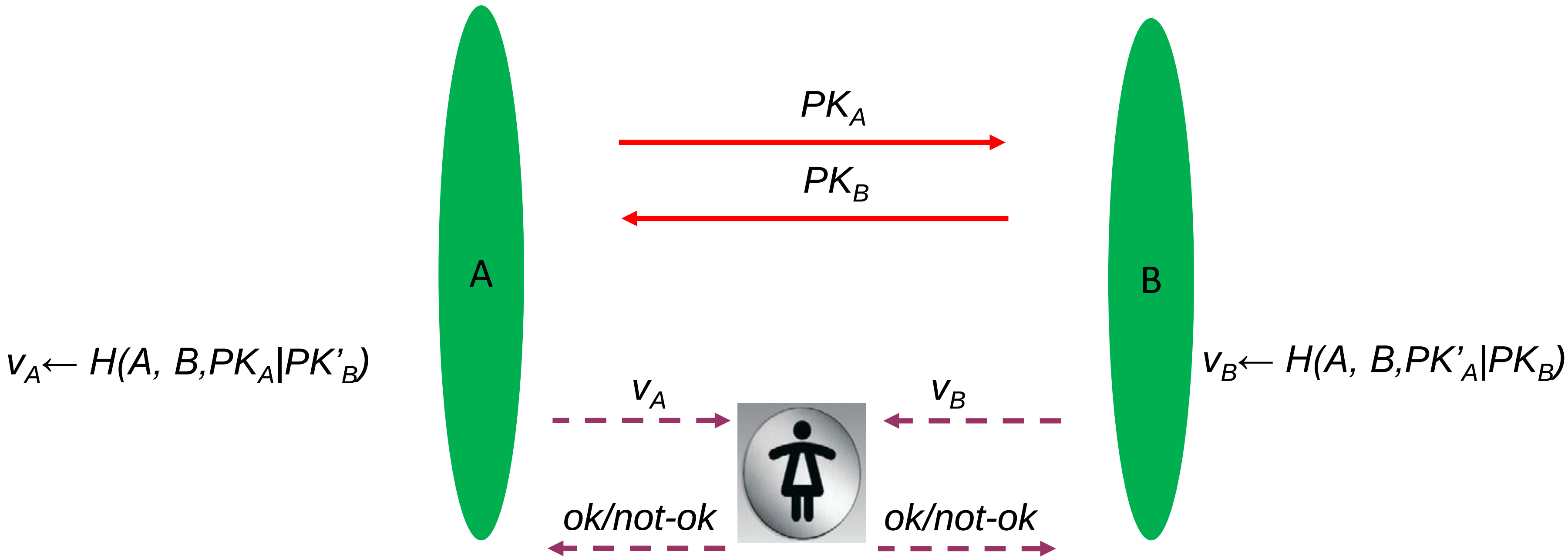
- User "bandwidth" is low (4 to 6 digits)
- Directionality depends on available hardware (1-way or 2-way)
- Security properties (integrity-only, or integrity+secrecy)

User as the secure channel

- Authentication of key agreement by
  - Comparing **short non-secret check codes** (aka "short authentication string"), or
  - entering a **short secret** "passkey"

- Short key/code should not hamper security
  - Standard security against offline attacks
  - Good enough security against active man-in-the-middle

Authentication by comparing short strings

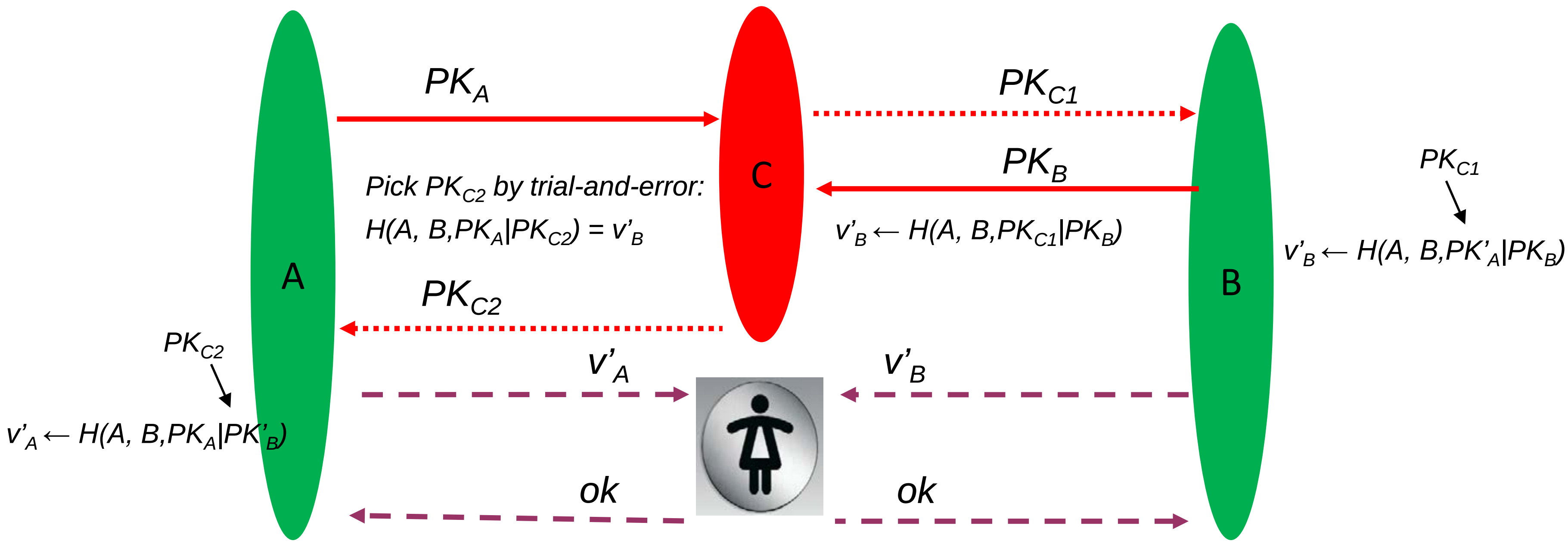


v_A and v_B are short strings (e.g., 4 digits),

User approves acceptance if v_A and v_B match

A man-in-the-middle can easily defeat this protocol

MitM in comparing short strings



Guess a value SK_{C2}/PK_{C2} until $H(A, B, PK_A|PK_{C2}) = v'_B$

If v'_B is n digits, attacker needs at most 10^n guesses; Each guess costs one hash calculation

A typical modern PC can calculate 100000 MACs in 1 second

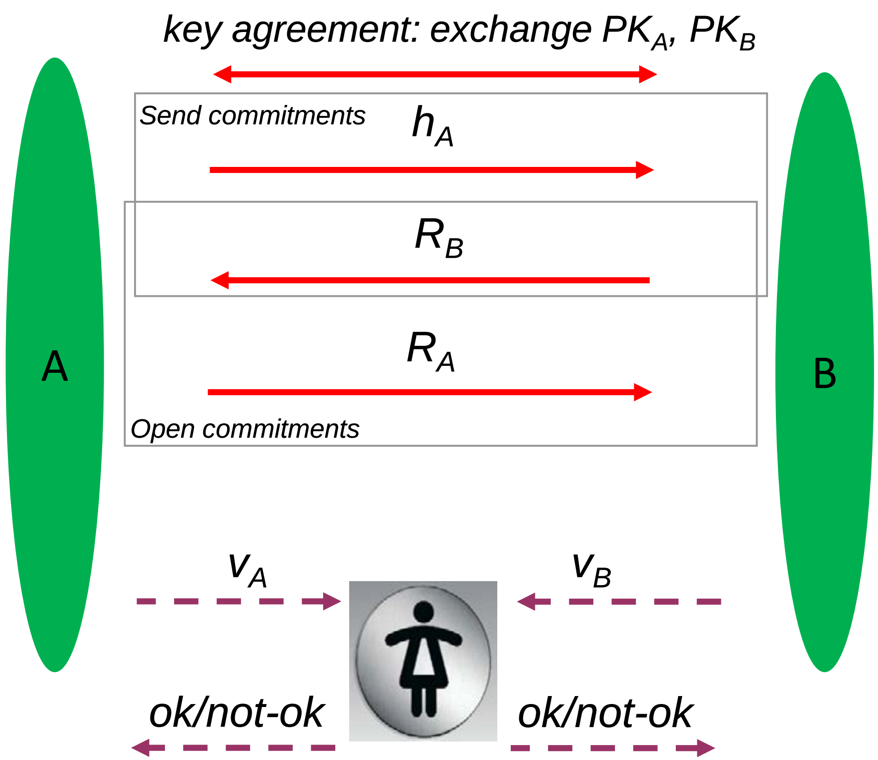
Authentication by comparing short strings

Choose long random R_A

Calculate commitment

$h_A \leftarrow h(A, R_A)$

$v_A \leftarrow H(A,B,PK_A|PK'_B,R_A,R'_B)$



Choose long random R_B

Verify commitment

$h'_A \stackrel{?}{=} h(A, R'_A)$

Abort on mismatch

$v_B \leftarrow H(A,B,PK'_A|PK_B,R'_A,R_B)$

User approves acceptance if v_A and v_B match

2^{-l} ("unconditional") security against man-in-the-middle (l is the length of v_A and v_B)

h() is a hiding commitment; in practice SHA-256

H*()* is a mixing function; in practice SHA-256 output truncated

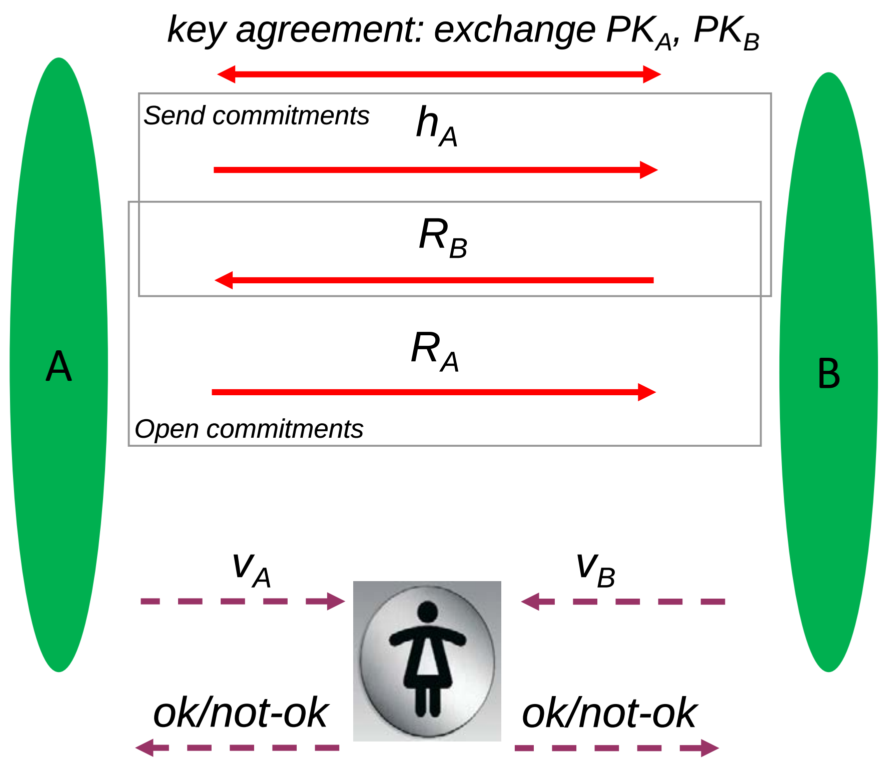
Authentication by comparing short strings

Choose long random R_A

Calculate commitment

$h_A \leftarrow h(A, R_A)$

$v_A \leftarrow H(A,B,PK_A|PK'_B,R_A,R'_B)$



Choose long random R_B

Verify commitment

$h'_A \stackrel{?}{=} h(A, R'_A)$

Abort on mismatch

$v_B \leftarrow H(A,B,PK'_A|PK_B,R'_A,R_B)$

User approves acceptance if v_A and v_B match

2^{-l} ("unconditional") security against man-in-the-middle (l is the length of v_A and v_B)

h() is a hiding commitment; in practice SHA-256

MANA IV by Laur, Asokan, Nyberg [IACR report] Laur, Nyberg [CANS 2006]

Authentication using interlocking short passkeys

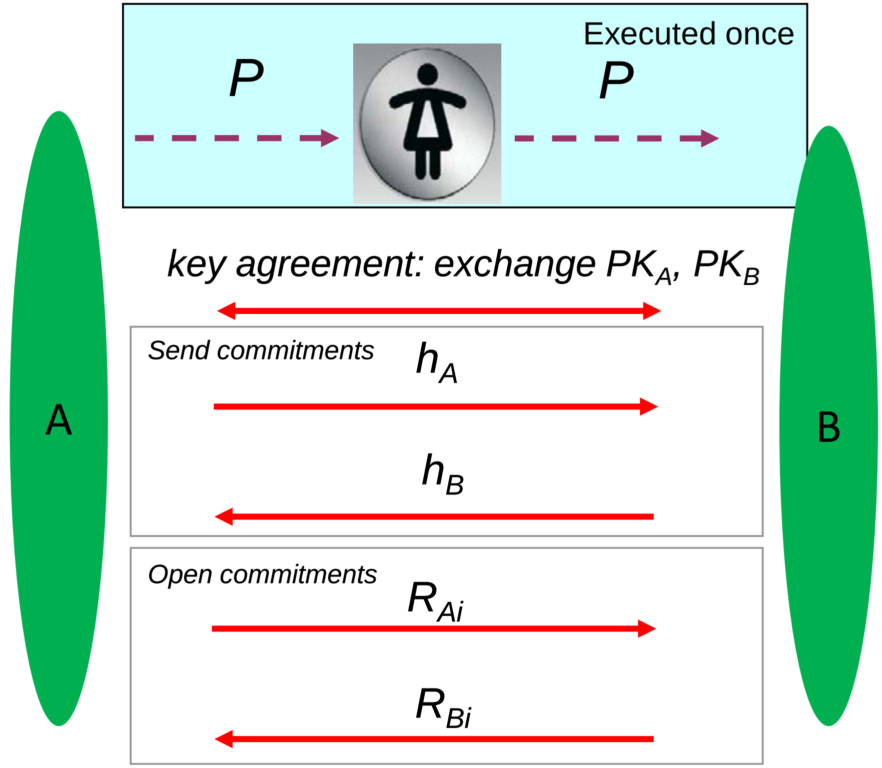
Choose long random R_{Ai}

Calculate commitment

$h_A \leftarrow h(A, PK_A|PK'_B, Pi, R_{Ai})$

Verify commitment

$h'_B \stackrel{?}{=} h(B, PK_A|PK'_B, Pi, R'_{Bi})$



Choose long random R_{Bi}

Calculate commitment

$h_B \leftarrow h(B, PK'_A|PK_B, Pi, R_{Bi})$

Verify commitment

$h'_A \stackrel{?}{=} h(A, PK'_A|PK_B, Pi, R'_{Ai})$

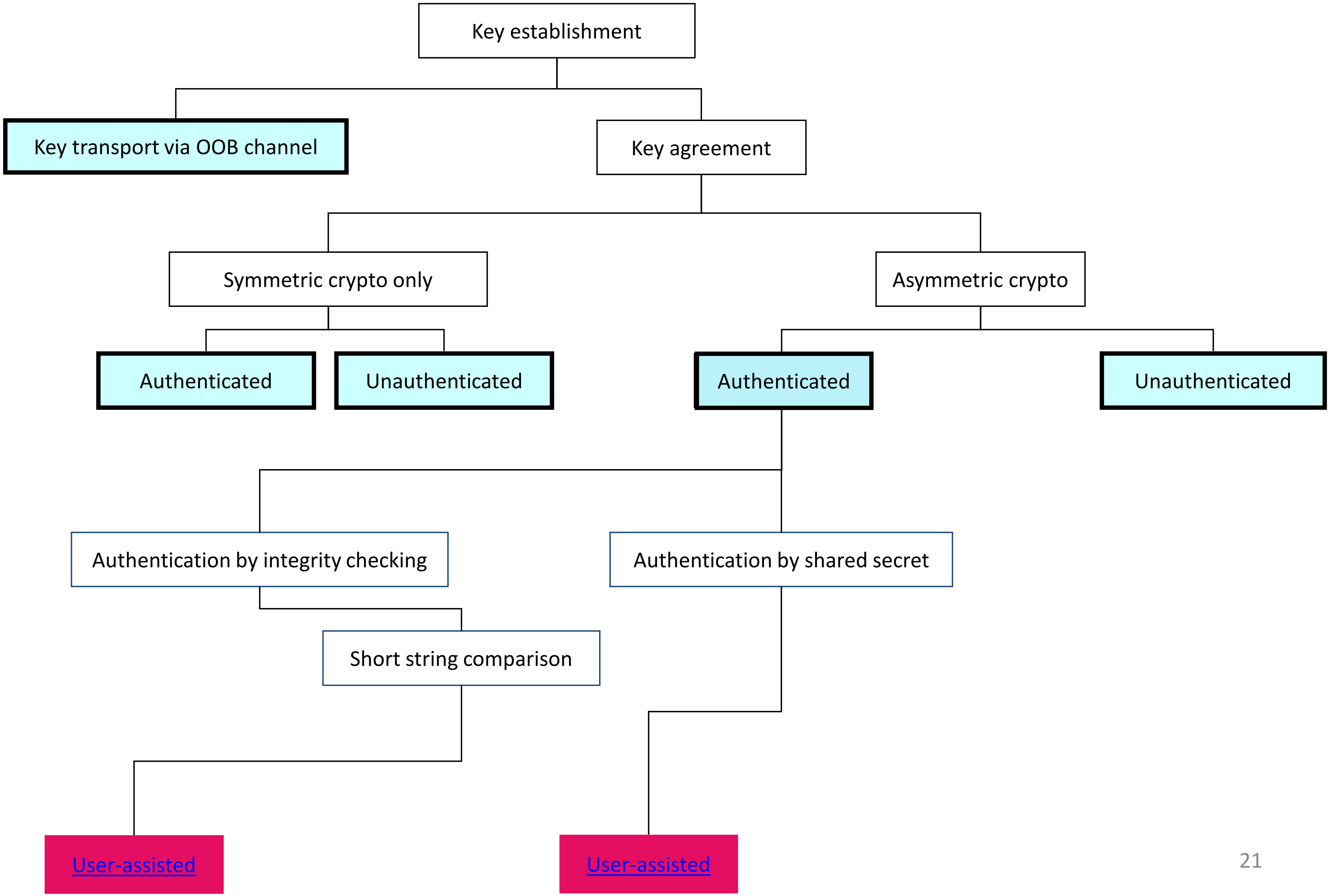
One-time passkey *P* is split into *k* parts ($l \geq k > 1$): next 4-round exchange repeated *k* times

h() is a hiding commitment; in practice SHA-256

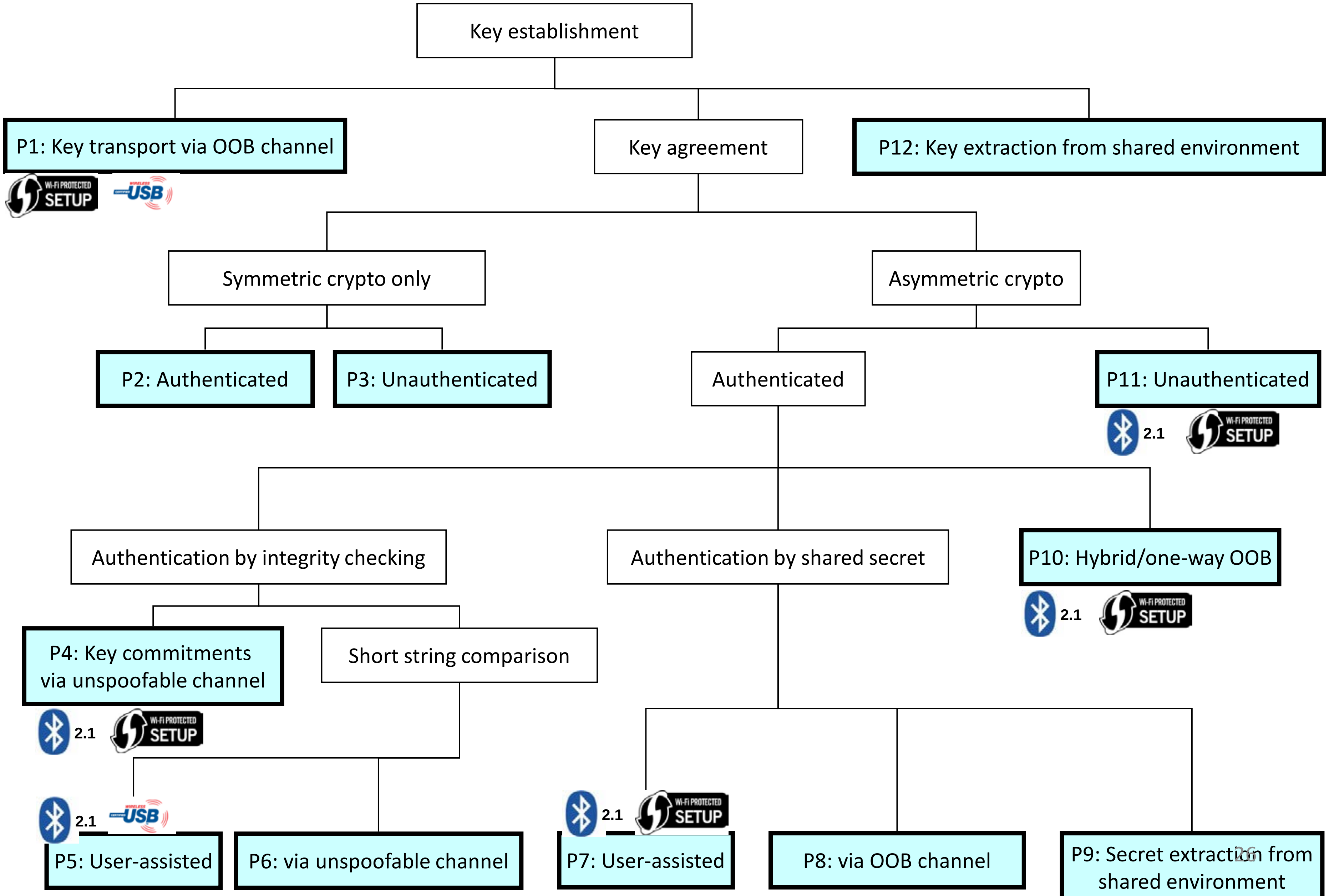
Up to $2^{-(l-1)}$ ("unconditional") security against man-in-the-middle (l is the length of *P*)

Originally proposed by Jan-Ove Larsson [2001]: essentially multi-round MANA III

Key establishment for first connect



Key establishment for first connect



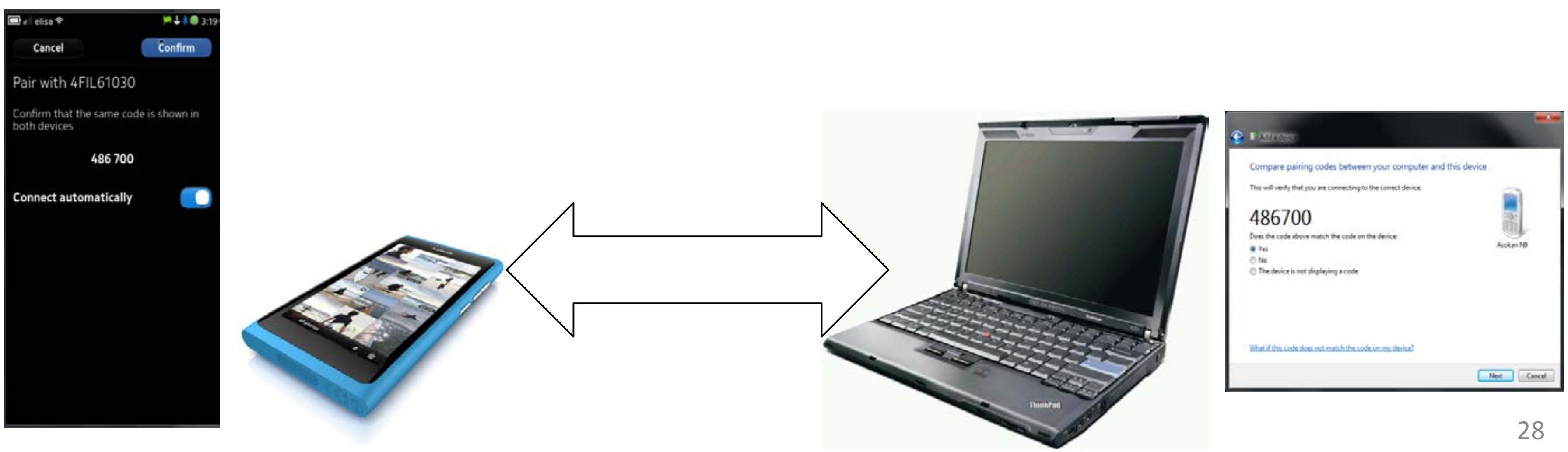
Key establishment for first connect ~2008

	Unauthenticated Diffie-Hellman	Authenticated Diffie-Hellman		
		short-string comparison	short PIN	Out-of-band channel
WiFi Protected Setup	"Push-button"		✓	NFC
Bluetooth 2.1	"Just-works"	✓	✓	NFC
Wireless USB		✓		USB Cable

"Security associations for wireless devices" (Overview, book chapter)
"Standards for security associations in personal networks: a comparative analysis" IJSN 4(1/2):87-100 (survey of standards)

First Connect: today

- Widely deployed (Bluetooth SSP, WiFi Protected Setup)
- **Improving usability/security → fundamental protocol changes**
  - Did it really help? (Usability Analysis of Secure Pairing Methods, USEC '07)
- Subsequent research exploiting properties of radio communication looks promising
  - Čapkun et al/TDSC 2008:5(4), Gollakota et al/Usenix Security '11

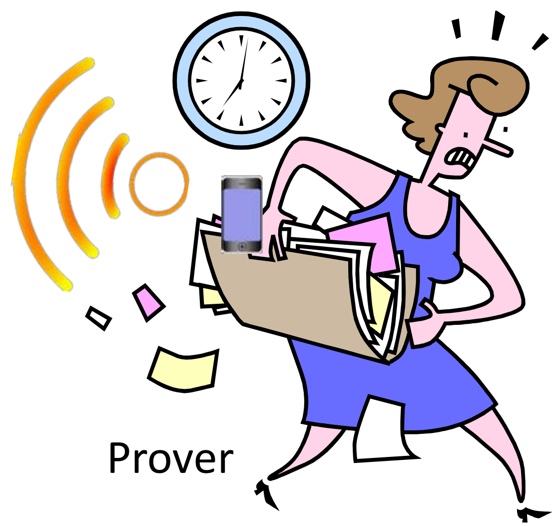


Pitfalls in Designing Zero-Effort Deauthentication

The deauthentication problem

Threat:

- **Unauthorized access** to a “terminal” after legitimate user has walked away
- Both **“innocent”** and **“malicious”**

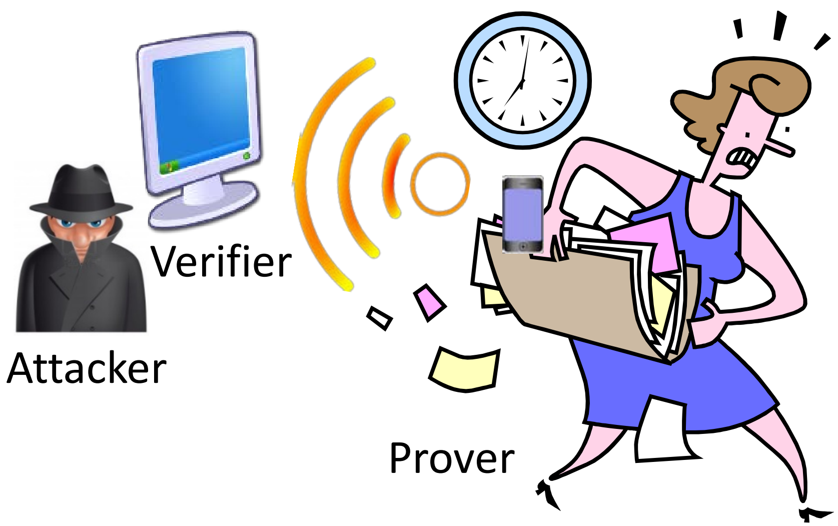


Cost:
False aggressive deauthentication → frustration

Deauthentication when “Not Far Enough”

Prover may be away from Verifier but still close

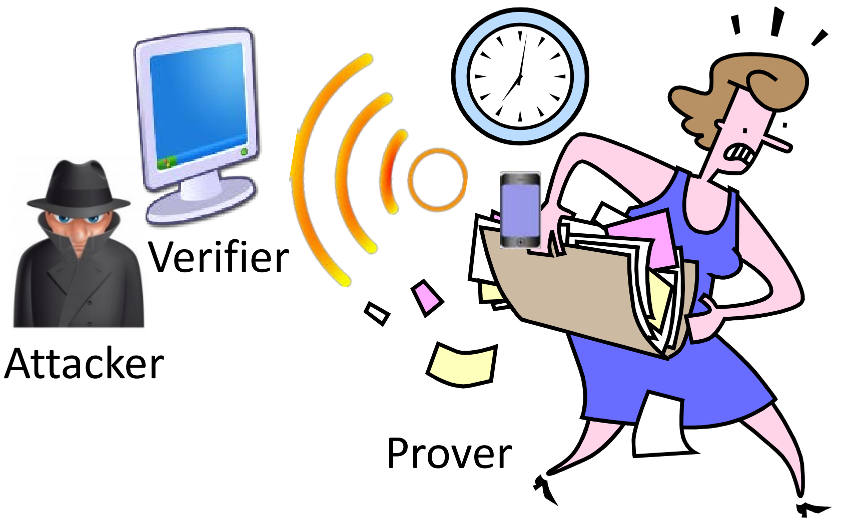
- Hospital wards, shop floors



ZEBRA

ZEro-effort Bilateral Recurring Authentication

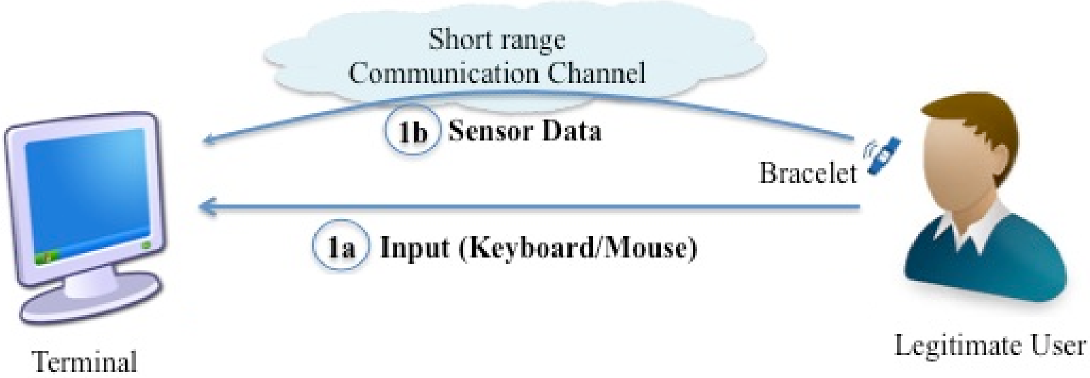
- Quickly and automatically deauthenticate (log out) user
- **...even with legitimate user is nearby**



[1] Mare, et al., "ZEBRA: Zero-effort bilateral recurring authentication." ***IEEE Symposium on Security and Privacy (SP) 2014*** **http://dx.doi.org/10.1109/SP.2014.51**

ZEBRA idea

- Each user has a bracelet: accelerometer/gyro
- Terminal compares bracelet data with its own
  - "bilateral recurring authentication"
- Transparent to user
  - "zero effort"



ZEBRA architecture

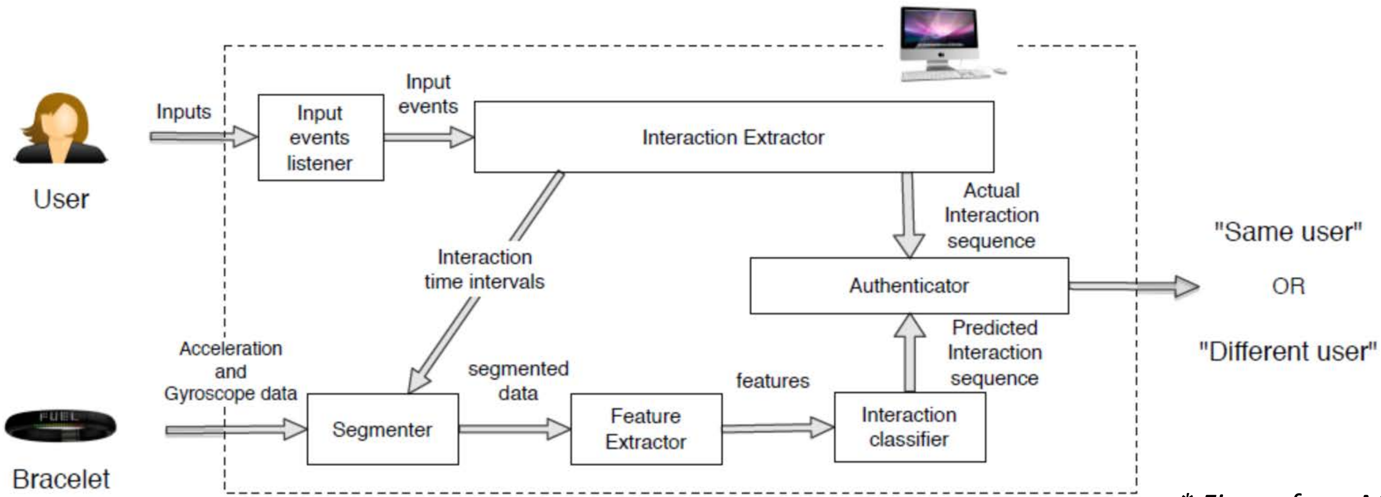


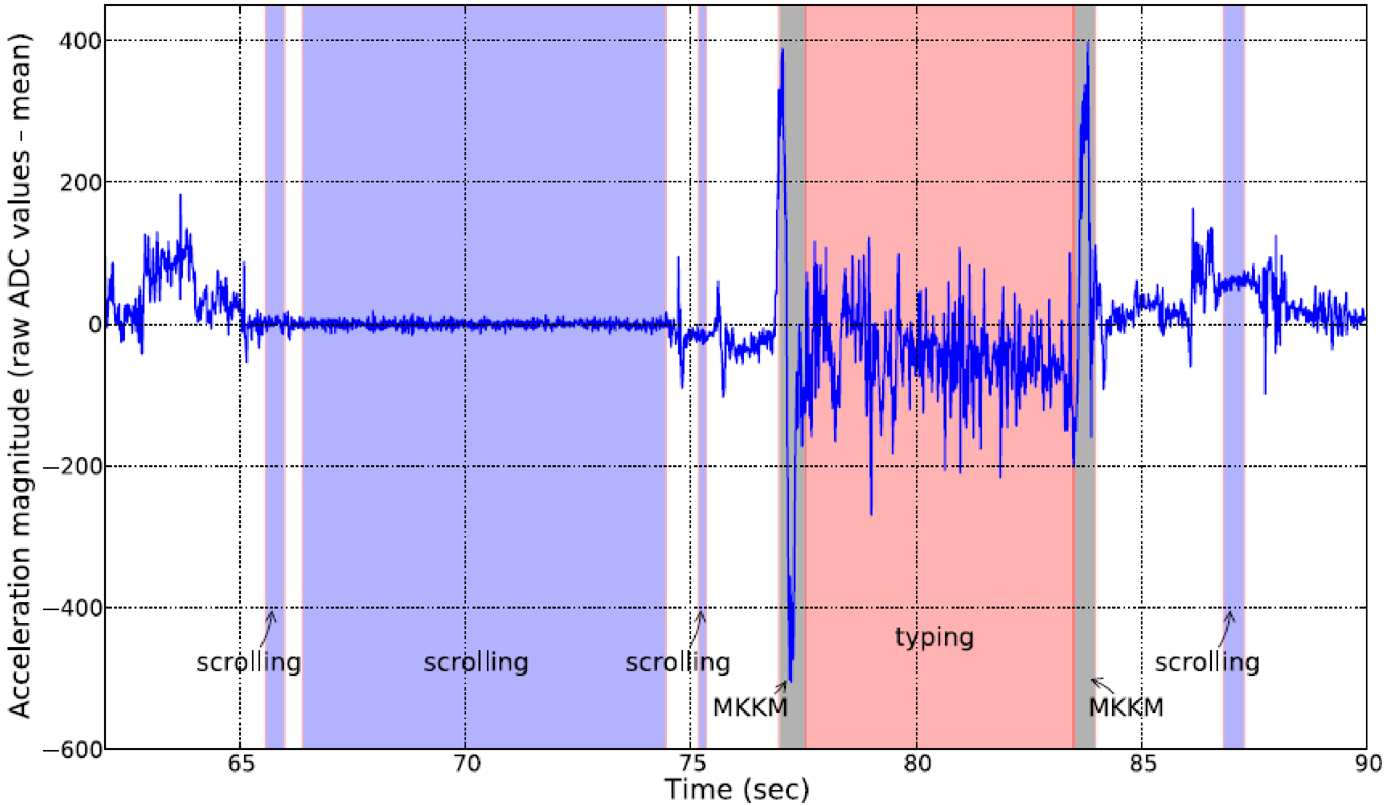
Figure 2: ZEBRA architecture.

** Figure from Mare et al. [1]*

- Interaction sequences: three types of events
  - Typing
  - Scrolling
  - MKKM: Mouse-to-KB or KB-to-Mouse

ZEBRA sensor data

Closer look at accelerometer measurements:

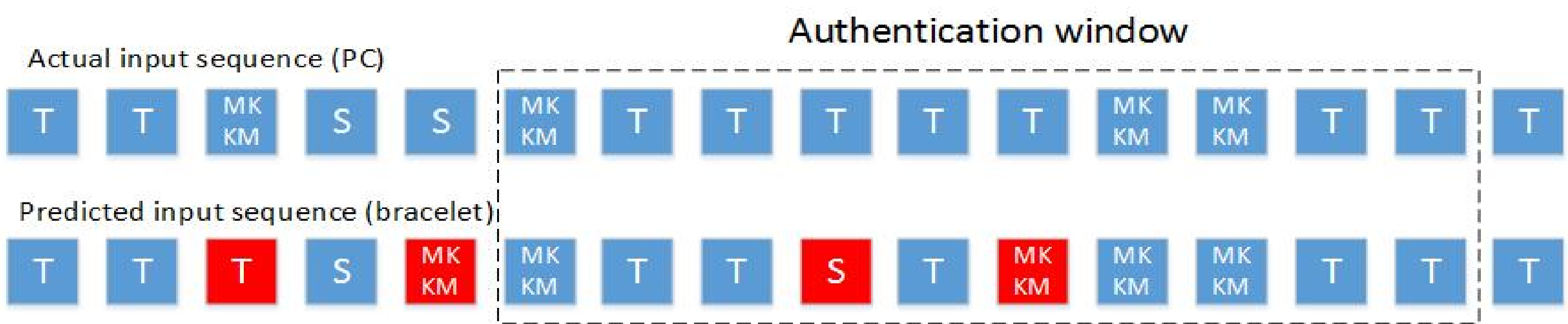


** Figure from Mare et al. [1]*

ZEBRA authentication

- Considers a **window** of interactions
  - robustness in the face of misclassifications
- Sets min. **threshold** for matching interactions in a window
  - When users fall below threshold, log them out

    Example:
    - Window size 10, Threshold 70%
    - 8/10 matches = 80% => User remains logged in



Bracelet data classified selectively

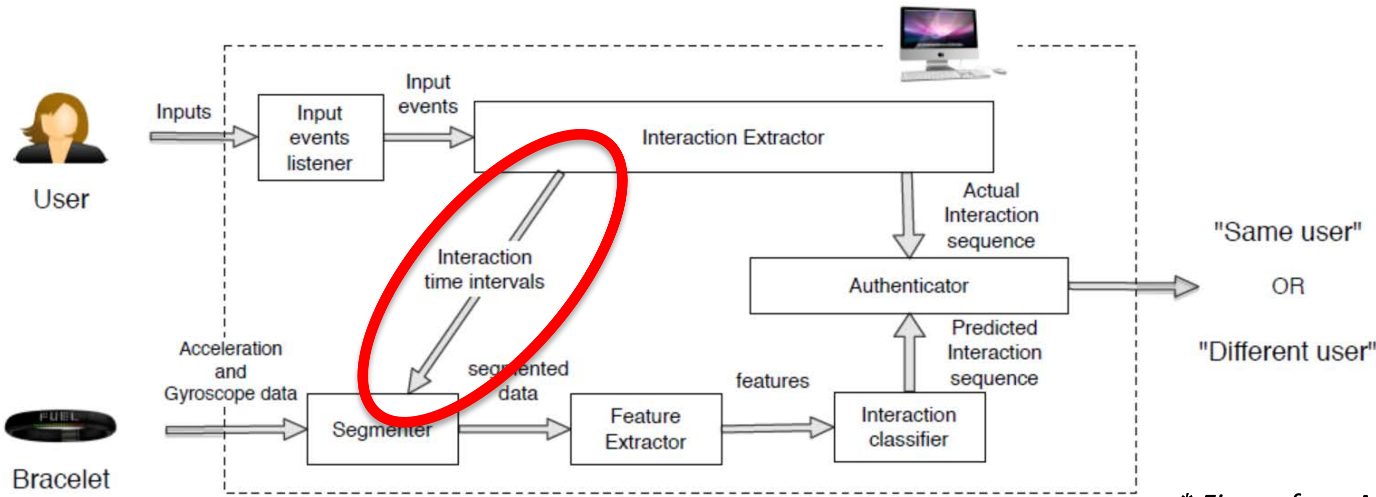


Figure 2: ZEBRA architecture.

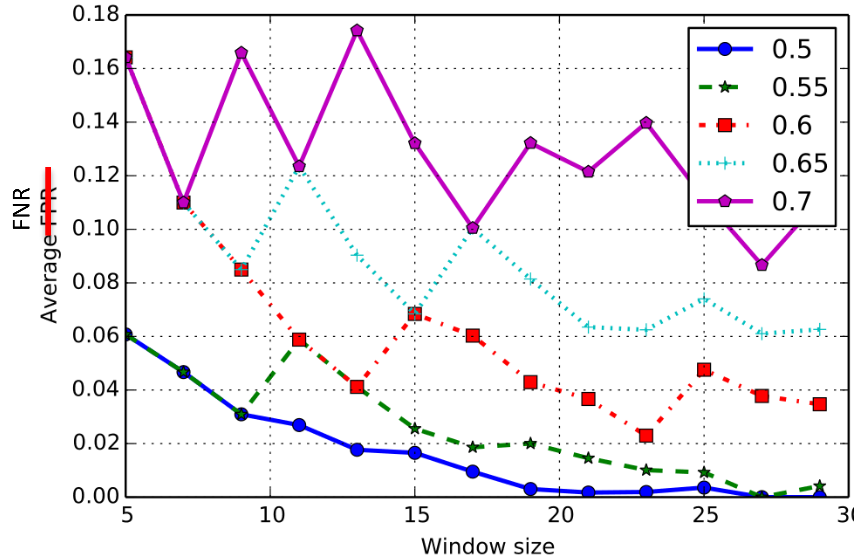
** Figure from Mare et al. [1]*

- Bracelet data classified only when Terminal sees input events
  – Why? User privacy [1], accuracy of classifier?
  – No activity → no predicted interaction sequence

ZEBRA Performance: Mare et al [1]

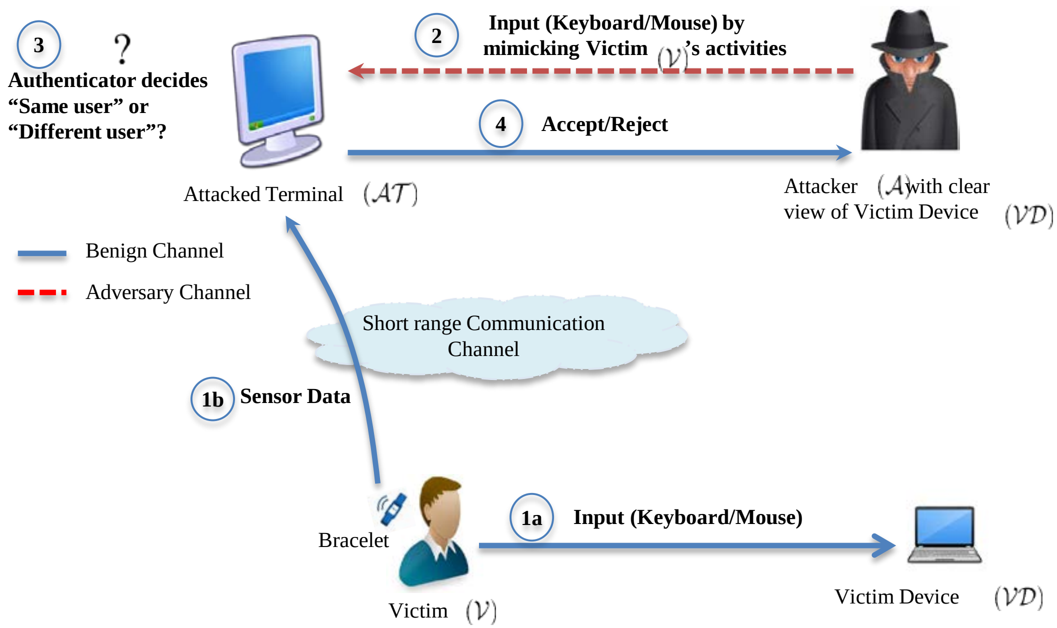
- Varies a lot depending on chosen parameters
  - **Window size** => Time it takes to detect attacker (5 – 30 different interactions)
  - **Threshold** => How many false interactions within one window (50 – 70%)

- Normal usage
  - **Usability**: False-negatives 0 – 17%
    - e.g. FNR 3%, FPR 13%
      (window size 10, threshold 50%)
  - **Security**: False-positives 0 – 17%
    - e.g. FNR 14%, FPR 2%
      (window size 10, threshold 70%)



** Figure from Mare et al. [1]*

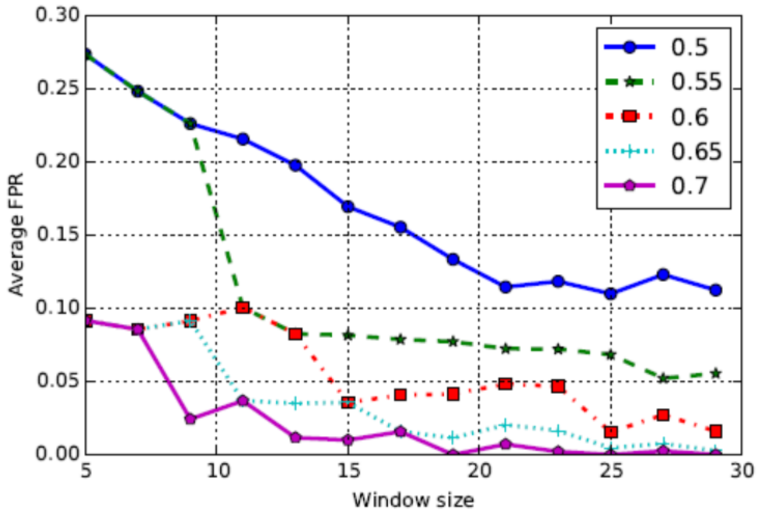
Modeling "malicious attacker" [1]



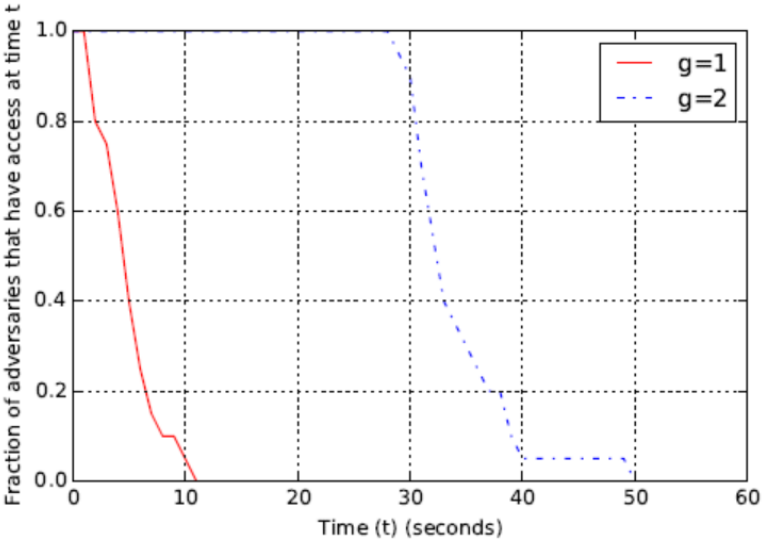
- Experiment with 20 participants
- Participant is attacker; researcher is victim
- Victims verbally announce their interactions
- Attacker asked to mimic all of victim's interactions

Security against malicious attackers

** Figures from Mare et al. [1]*



Average FPR for different window sizes and thresholds



Fraction of adversaries remaining logged in (window size = 21, threshold=60%)

ZEBRA performs well against such attackers [1]

Breaking ZEBRA

Attacking ZEBRA

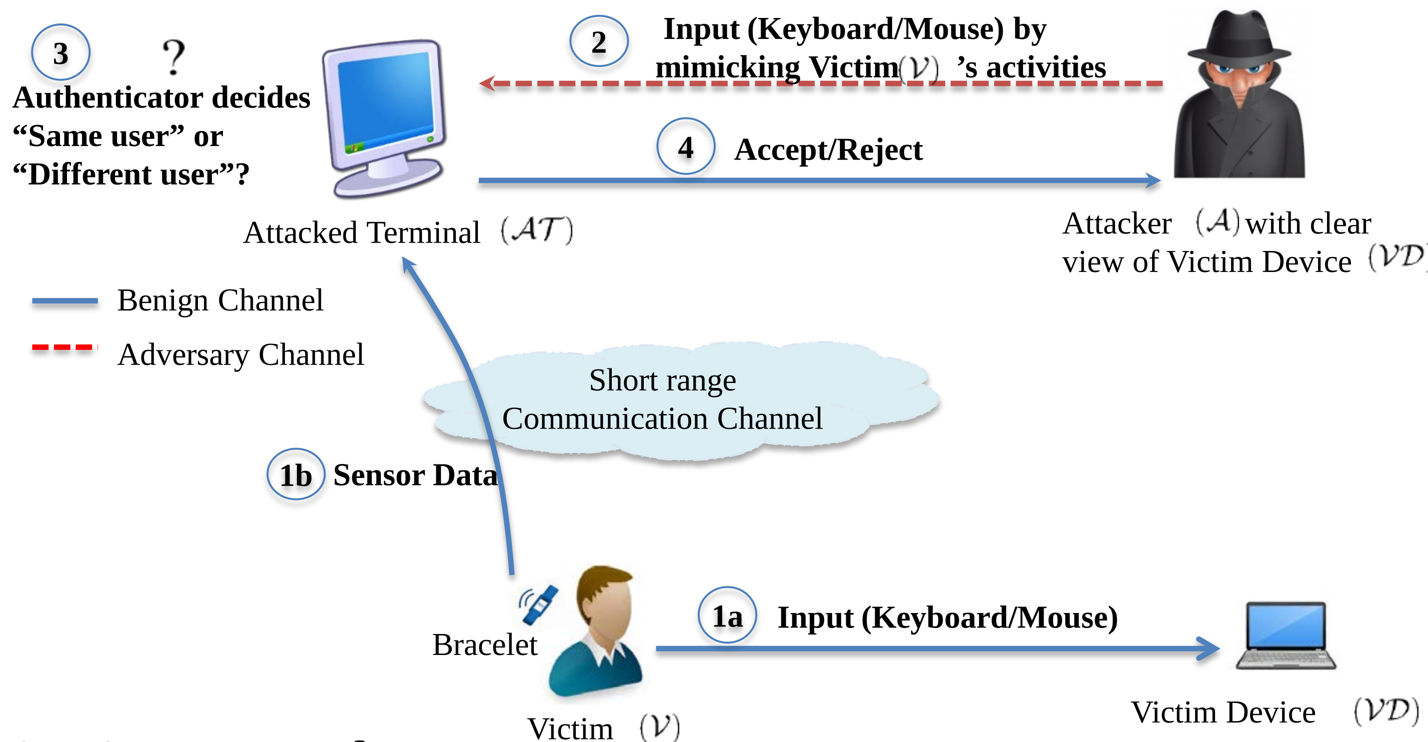
Weaknesses:

- **Deauthentication dependent on PC activity**
  → No activity, no deauthentication!
  → Attacker controls when and what interactions are compared!

- **Sequence of interactions needed for decision**
  → Can take long to deauthenticate (5 – 30 interactions)

- **Low decision threshold allows many 'false' interactions**
  → Trade-off between usability and security (50 – 70% threshold)

Opportunistic attacker



Opportunistic attacker

- Observe user interactions
- Mimic interactions **selectively:** e.g., focus only on mimicking typing interactions

Possible attack scenarios

1. Näive all-activity
   – As in Mare et al [1]: mimic all activities
2. Opportunistic KB-only
   – Mimic only selected typing activity
3. Opportunistic all-activity
   – Mimic all *types* of activities, but selectively
4. Audio-only opportunistic KB-only
   – Same as Opportunistic KB-only, but assuming that attacker can only hear, but not see, the victim

Skip to attacks

Our implementation of ZEBRA

- Implemented end-to-end ZEBRA from scratch

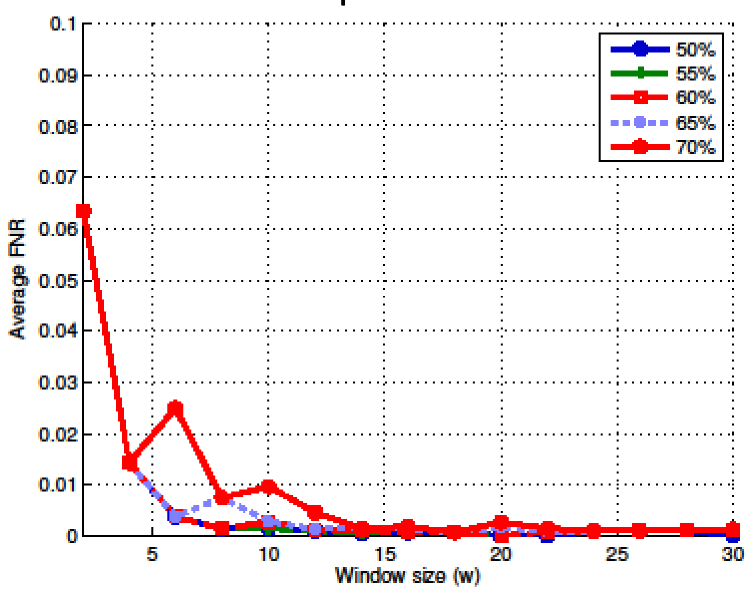
- Using off-the-shelf Android Wear smartwatch
  - Wider applicability: existing affordable models
  - Original Shimmer device expensive/single-purpose

- Re-use ZEBRA parameters wherever possible
  - ZEBRA paper did not list all parameter choices

Performance analysis: closer look

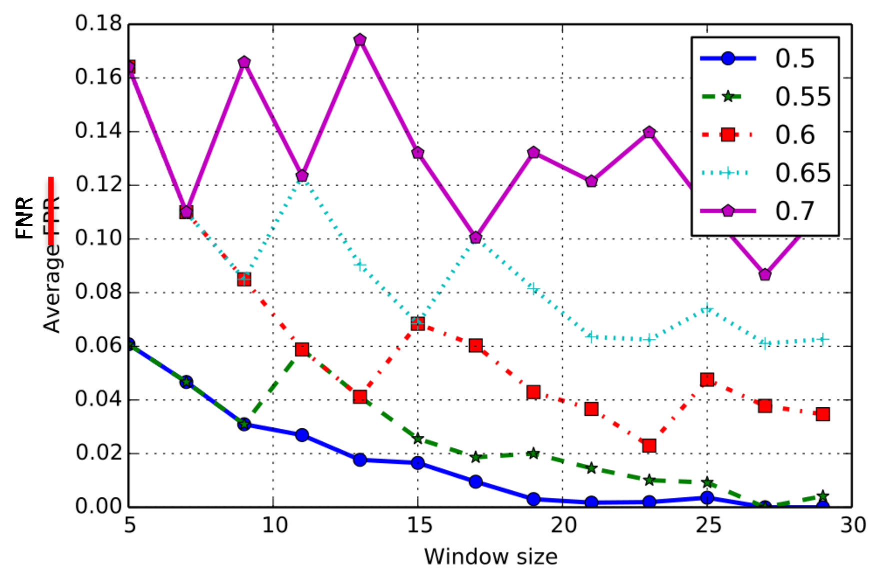
- Good usability: low FNR for legitimate users
  - (No legitimate users logged out)

Our Implementation



Average FNR for different window sizes and thresholds

Mare et al [1] implementation



** Figure from Mare et al. [1]*

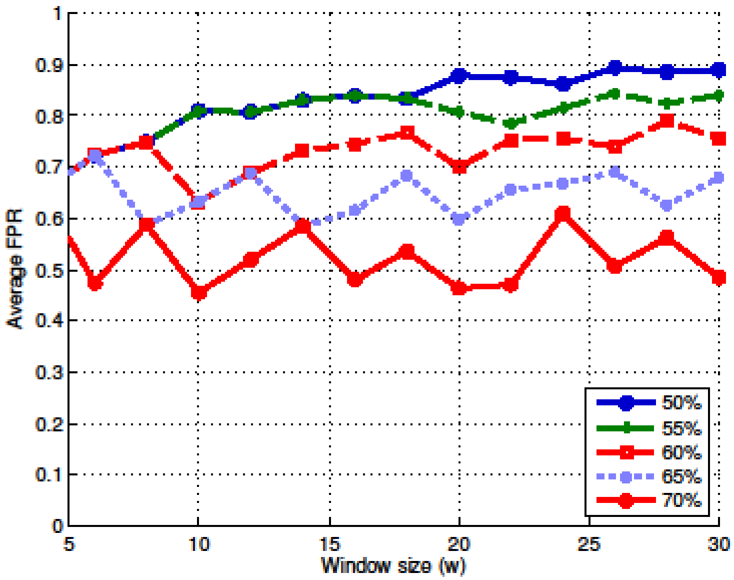
Opportunistic attack experiments

- ZEBRA is susceptible to Opportunistic Attacker
  - 40% of opportunistic attackers not detected at all (up to 10 mins)
  - 80% remain logged in after one minute
- Participant is victim; researcher is attacker

Attack analysis: closer look 1/2

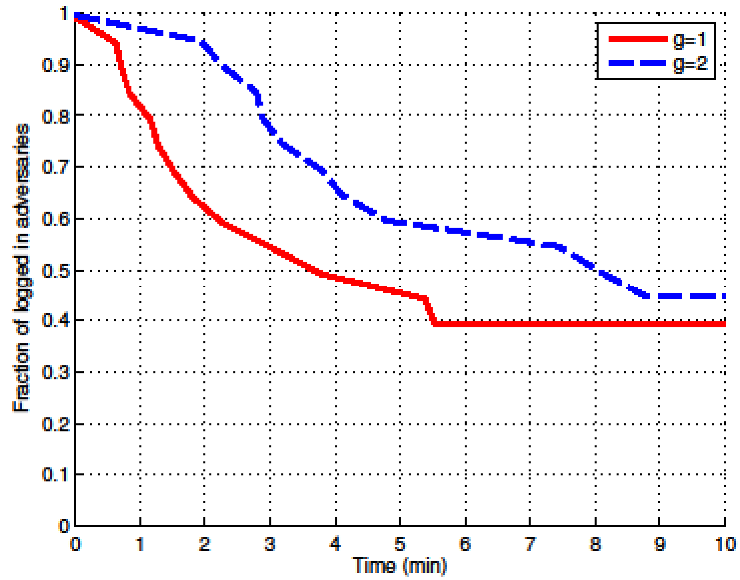
- Vulnerable to Opportunistic KB-only Attacker
  - Attacker opportunistically mimics only typing

False Positives very high



Average FPR for different window sizes and thresholds

40% of attackers not detected

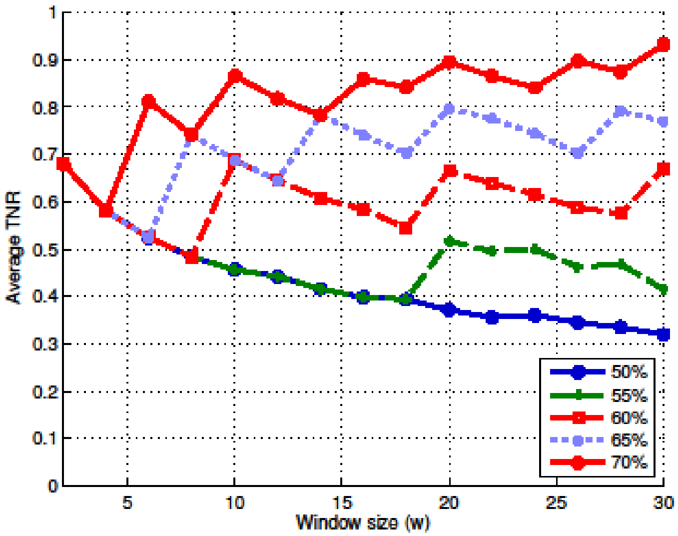


Fraction of attackers remaining logged in
(window size = 20, threshold=60%)

Attack analysis: closer look 2/2

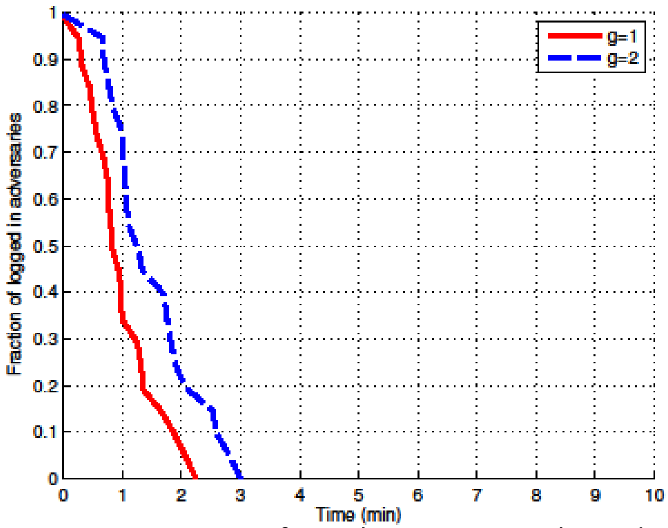
- Can still protect against accidental misuse
  - All users eventually logged out
- Performance for mismatched traces

True Negatives now high



Average TNR for different window sizes and thresholds

All "attackers" logged out

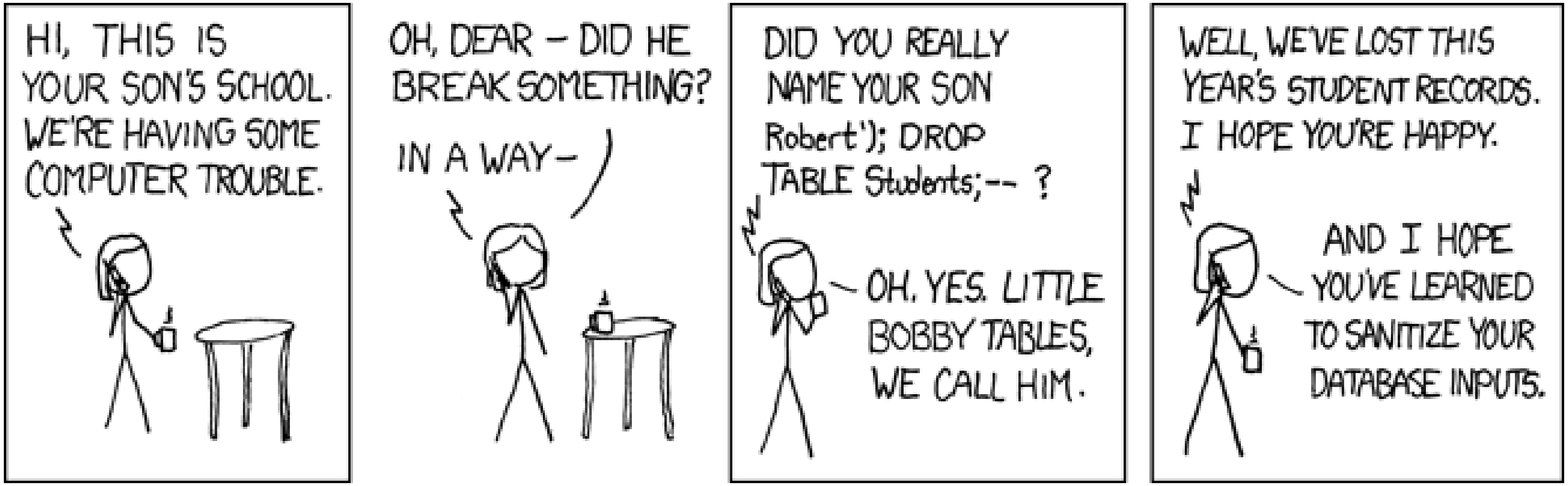


Fraction of attackers remaining logged in (window size = 20, threshold=60%)

Improving ZEBRA

What is wrong with ZEBRA?

- Fundamental design flaw:
  "Authentication based on input source controlled by adversary"
- A case of **tainted input**
  – Attacker controls Terminal
  – Can choose type/timing of interactions



https://xkcd.com/327/

What is wrong with ZEBRA?

- Fundamental design flaw:

  "Authentication based on input source controlled by adversary"

- A case of **tainted input**
  - Attacker controls Terminal
  - Can choose type/timing of interactions

- Fixes:
  - Trigger authentication based on sensor data
  - Sanitize untrusted input (PC interactions)
    - Blacklist known bad interaction sequences
    - Whitelist only interaction sequences known to be good

https://xkcd.com/327/

ZEBRA summary

- Designing usable secure systems correctly is hard
  - Balance between usability and security
  - Care in defining threat model

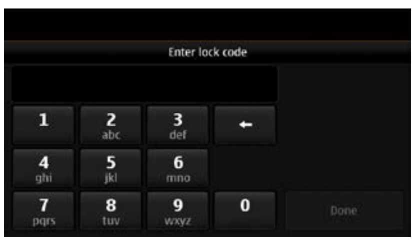
- ZEBRA susceptible to **opportunistic attackers** still usable for preventing **accidental misuse**

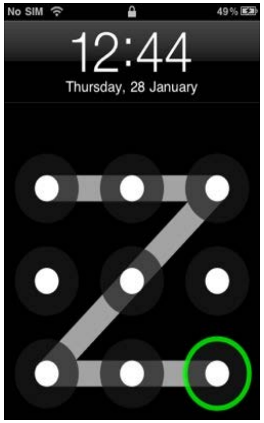
- Paper to be presented at NDSS '16
  - Pitfalls in Designing Zero-Effort Deauthentication: Opportunistic Human Observation Attacks http://arxiv.org/abs/1505.05779

Other usable security problems

Local user authentication

[Shoulder-surfing resistance of authentication based on image recognition (SOUPS '10)]





Cost: users avoid using apps that mandate local authentication (work e-mail!)

Cost: weak PINs

Need alternatives that are:

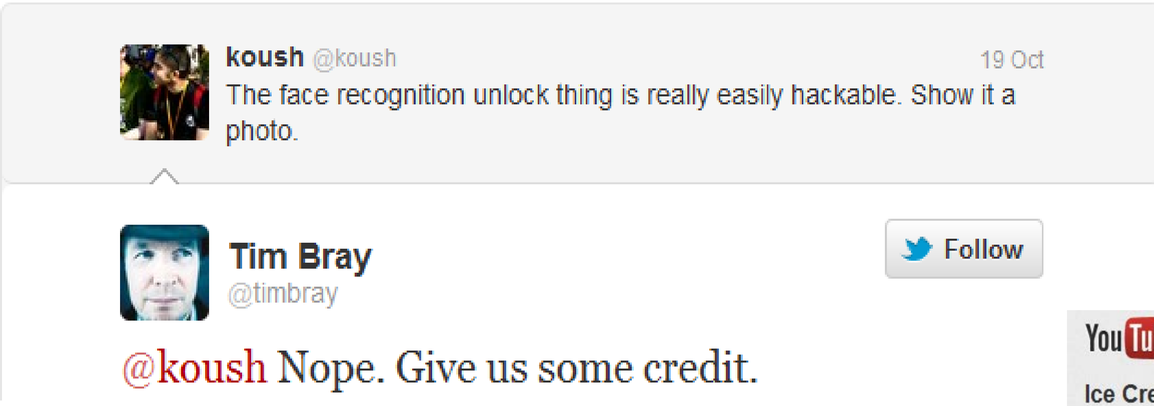
- Faster
- More enjoyable
- Secure enough

Biometrics

Wearables

?

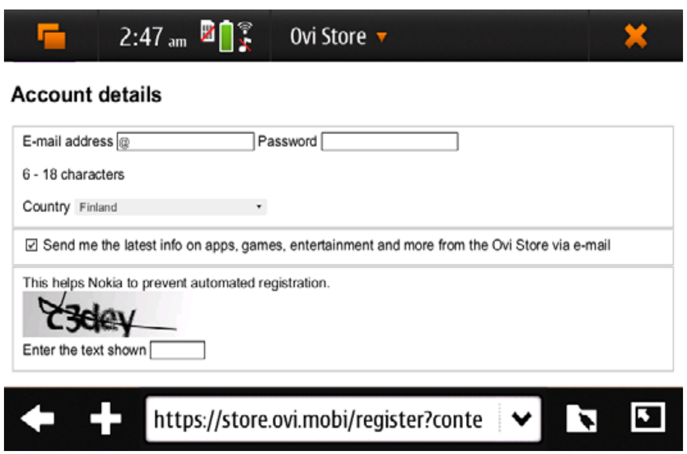
Local user auth.: a cautionary tale

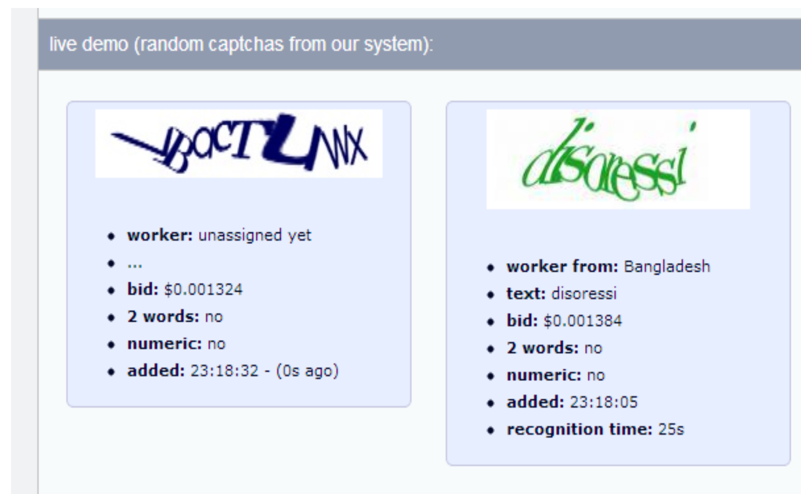


http://youtu.be/BwfYSR7HttA

CAPTCHA on mobile devices

Cost:
Estimated 15% drop-off rate when encountering a CAPTCHA on mobile devices





http://antigate.com

Alternatives to standard CAPTCHA?

- The problem is real
- Can it be solved without CAPTCHA?
  - Device authentication
- Mobile-friendly CAPTCHA variants?

Usable security problems on mobile devices

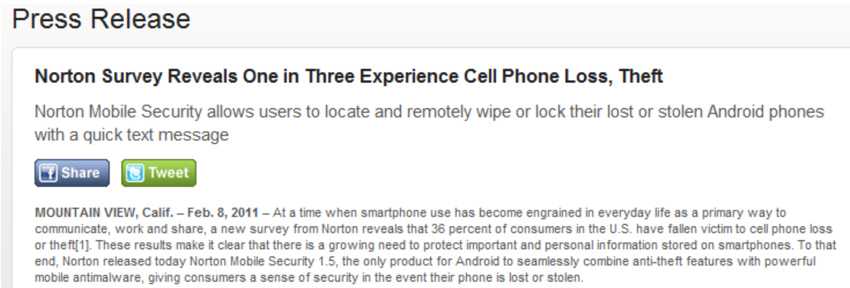
- Secure First Connect
- Continuous user authentication
  - (and deauthentication)
- Local user authentication
- CAPTCHA
- Permission granting to apps
- …?

Mobility helps security/privacy

- Mobility/portability can help in surprising ways: e.g.,
  - PayPal Bump
  - ”Mobility helps security in ad hoc networks”, Čapkun et al, MobiHoc ’03
  - …
- Mobiles sense location, motion, light/sound, …
  - Use cues from context/history to set sensible access control policies ? (“Contextual Security”)

Skip to Summary

An example: device lock


Press Release

Norton Survey Reveals One in Three Experience Cell Phone Loss, Theft

Norton Mobile Security allows users to locate and remotely wipe or lock their lost or stolen Android phones with a quick text message

Share Tweet

MOUNTAIN VIEW, Calif. – Feb. 8, 2011 – At a time when smartphone use has become engrained in everyday life as a primary way to communicate, work and share, a new survey from Norton reveals that 36 percent of consumers in the U.S. have fallen victim to cell phone loss or theft[1]. These results make it clear that there is a growing need to protect important and personal information stored on smartphones. To that end, Norton released today Norton Mobile Security 1.5, the only product for Android to seamlessly combine anti-theft features with powerful mobile antimalware, giving consumers a sense of security in the event their phone is lost or stolen.

http://www.symantec.com/about/news/release/article.jsp?prid=20110208_01


nakedsecurity IT Security Blog of the Year

News. Opinion. Advice. Research

malware | spam | social networks | data loss | law & order | apple | podcast | vic

FLAMING RETORT: Hacktivism, hacking and hackers - what do these words really mean?

Hacking gang breaks into Norwegian killer's email accounts

Survey says 70% don't password-protect mobiles: download free Mobile Toolkit

Join thousands of others, and sign-up for Naked Security's newsletter

you@example.com Do it!

Don't show me this again

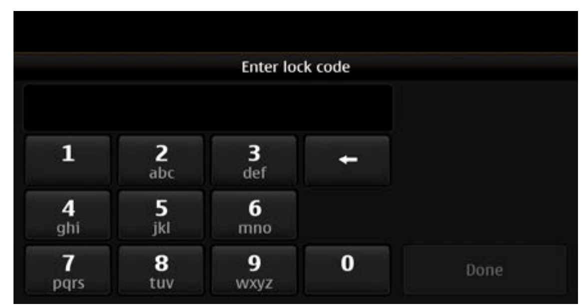
by Carole Theriault on August 9, 2011 | Comments (5)

FILED UNDER: Data loss, Featured, Malware, Mobile, Social networks, Video

Have you ever lost your mobile phone? I have. Four times last year.

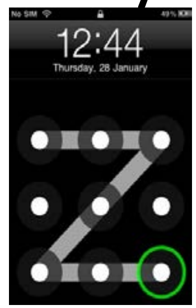
http://nakedsecurity.sophos.com/2011/08/09/free-sophos-mobile-security-toolkit/

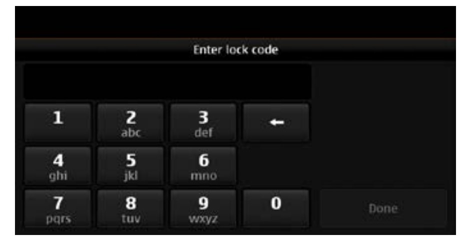
- Intended for theft protection
- Example of one-size-fits-all
  - Device lock always kicks in
- Can be annoying in
  - Freezing weather
  - Groggy mornings
  - …



Better device lock via context profiling

- Timeout and unlocking method adjusted based on estimated familiarity/safety of current context

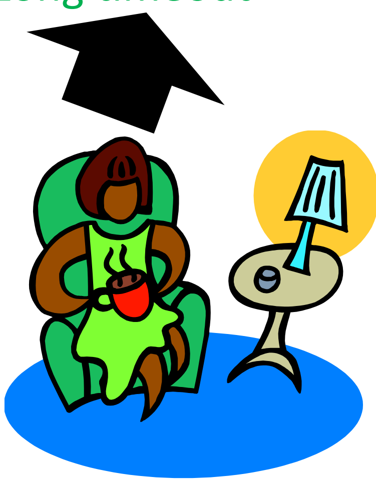




Long timeout

Medium timeout

Short timeout



Home

Work Cafeteria

Unknown

Estimating familiarity of people & places

[Aditi Gupta et al, SocialCom '12](#)
[Markus Miettinen et al, ACM ASIACCS '14](#)

Devices are proxies for people

Detect nearby devices & keep track of encounters

Identify places ("contexts") meaningful to user

Estimate context familiarity based on who is nearby

How to estimate safety?

Other contextual security solutions

Access control based on implicit user gestures

Mind How You Answer Me!

(Transparently Authenticating the User of a Smartphone when Answering or Placing a Call)

Mauro Conti Irina Zachia-Zlatea Bruno Crispo

http://dx.doi.org/10.1145/1966913.1966945

Tap-Wave-Rub: Lightweight Malware Prevention for Smartphones using Intuitive Human Gestures

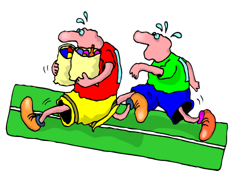
Haoyu Li[1], Di Ma[1], Nitesh Saxena[2], Babins Shrestha[2], and Yan Zhu[1]

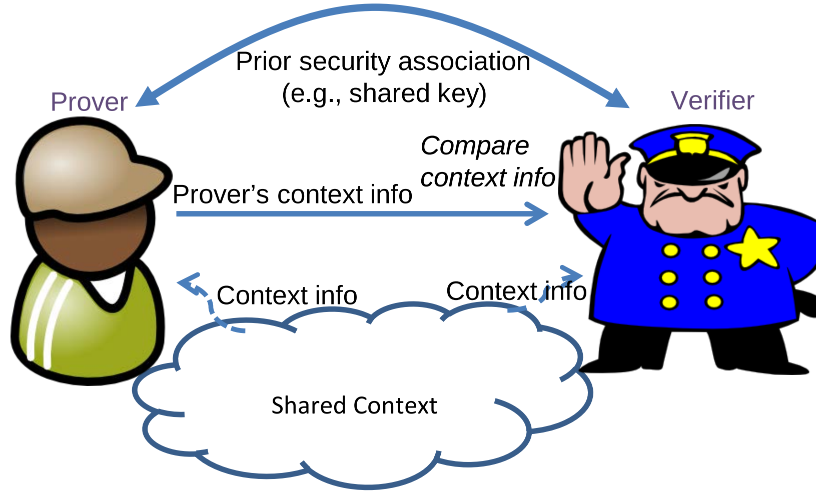
http://dx.doi.org/10.1145/2462096.2462101

Other contextual security solutions

Comparing contexts for zero-interaction auth.

http://sourceforge.net/projects/blueproximity/





But naive zero-interaciton auth is vulnerable to relay attacks!

Comparing and Fusing Different Sensor Modalities for Relay Attack Resistance in Zero-Interaction Authentication

Hien Thi Thu Truong*, Xiang Gao*, Babins Shrestha†, Nitesh Saxena†, N.Asokan‡ and Petteri Nurmi*

http://se-sy.org/projects/coco

Other contextual security solutions

Key agreement based on shared context

Amigo: Proximity-Based Authentication of Mobile Devices

Alex Varshavsky[1], Adin Scannell[1], Anthony LaMarca[2], and Eyal de Lara[1]

http://link.springer.com/chapter/10.1007%2F978-3-540-74853-3_15

Secure Communication Based on Ambient Audio

Dominik Schürmann and Stephan Sigg, *Member, IEEE Computer Society*

http://dx.doi.org/10.1109/TMC.2011.271

http://dx.doi.org/10.1109/TMC.2011.271

ACM CCS 2014: "Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices"

http://doi.acm.org/10.1145/2660267.2660334

Challenges in contextual security

- What is the right adversary model?
  - Can guess context information?
  - Can manipulate integrity of context sensing?
- Ensuring user privacy

Summary

- Usable security is challenging but worthy
  - Lack thereof results in surprising costs
  - Needs changes under-the-hood
    - protocols, algorithms, …
  - Calls for careful design
- No satisfactory solutions yet for several instances
- Contextual cues can help

Slides available at
http://asokan.org/asokan/research/talks.php