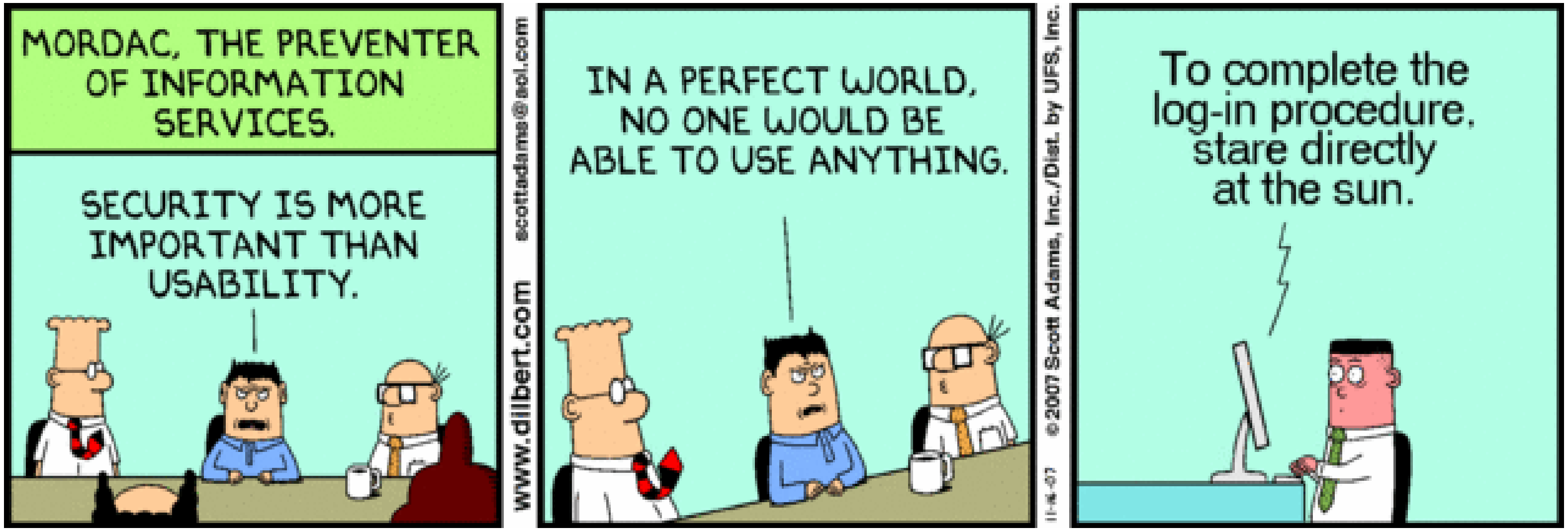
Usability of Mobile Security

TCE Summer School, 2014

N. Asokan

Aalto University and University of Helsinki


MORDAC, THE PREVENTER OF INFORMATION SERVICES.
SECURITY IS MORE IMPORTANT THAN USABILITY.
IN A PERFECT WORLD, NO ONE WOULD BE ABLE TO USE ANYTHING.
To complete the log-in procedure, stare directly at the sun.
www.dilbert.com scottadams@aol.com
© 2007 Scott Adams, Inc./Dist. by UFS, Inc.
11-16-07

http://www.dilbert.com (11/16/2007)

Why worry about usability?

Lack of security usability

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

Outline

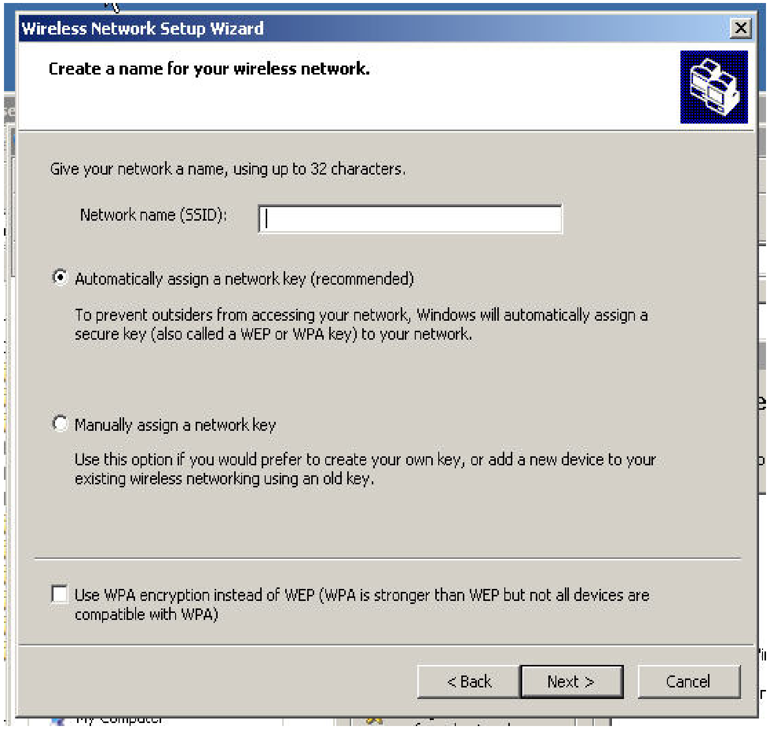
- Two case studies
  - Secure First Connect
  - Granting permission to apps
- Why usable mobile security is different
- Examples of usable mobile security problems

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 in the home WLAN
- **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



Wireless Network Setup Wizard

Create a name for your wireless network.

Give your network a name, using up to 32 characters.

Network name (SSID):

● Automatically assign a network key (recommended)

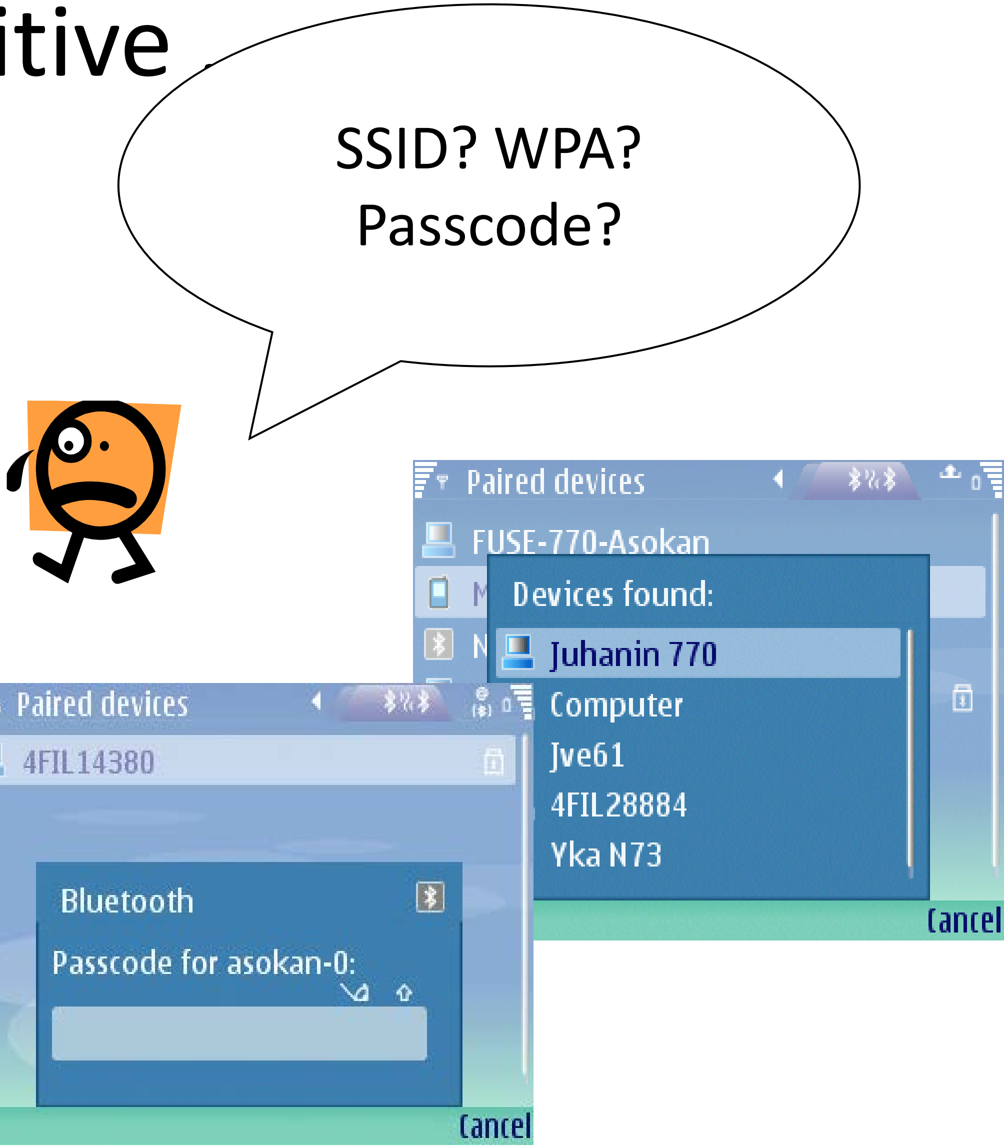
To prevent outsiders from accessing your network, Windows will automatically assign a secure key (also called a WEP or WPA key) to your network.

○ Manually assign a network key

Use this option if you would prefer to create your own key, or add a new device to your existing wireless networking using an old key.

□ Use WPA encryption instead of WEP (WPA is stronger than WEP but not all devices are compatible with WPA)

< Back | Next > | Cancel



Paired devices

FUSE-770-Asokan

Devices found:

Juhanin 770

Computer

Jve61

4FIL28884

Yka N73

Cancel

Paired devices

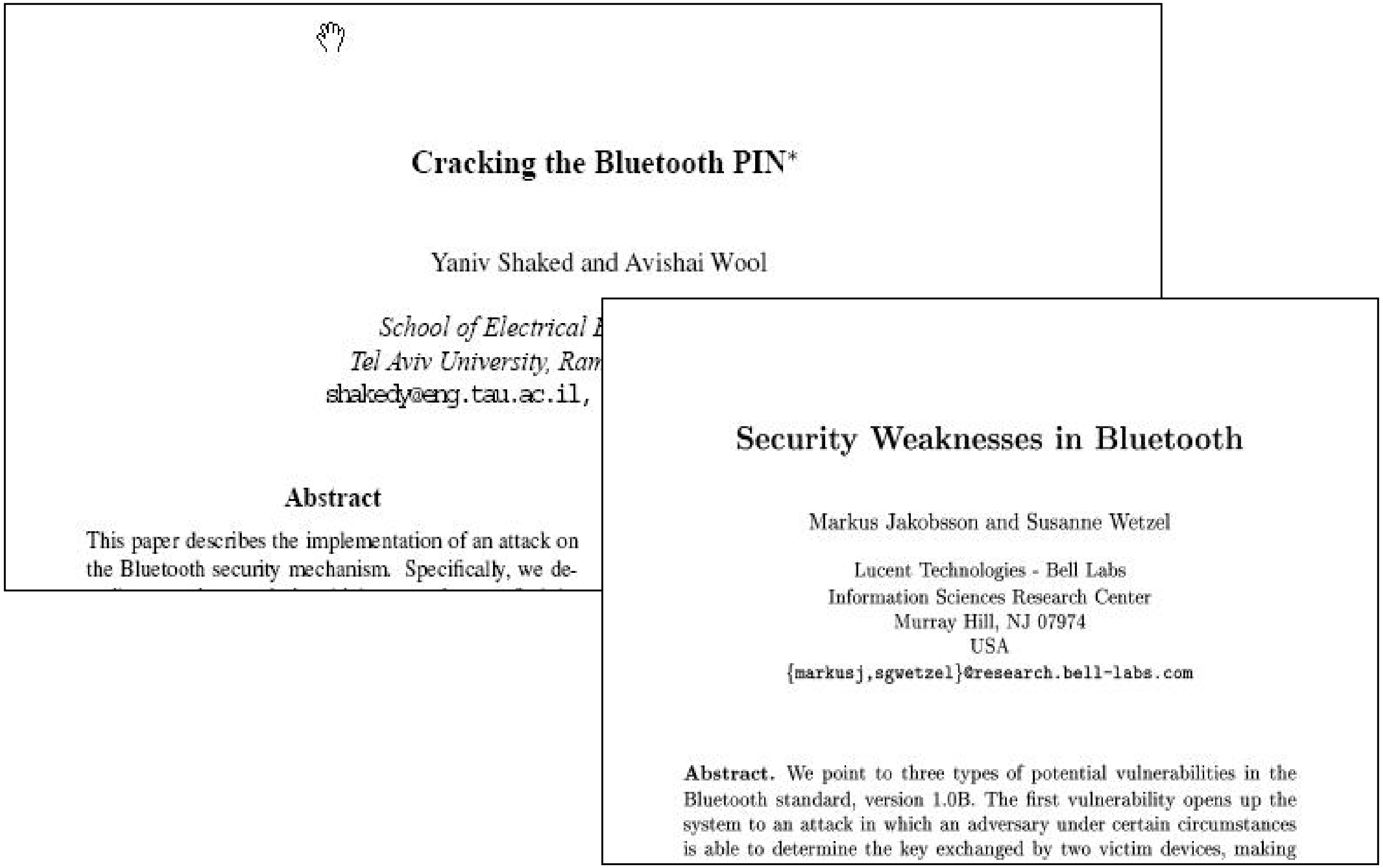
4FIL14380

Bluetooth

Passcode for asokan-0:

Cancel

… 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-

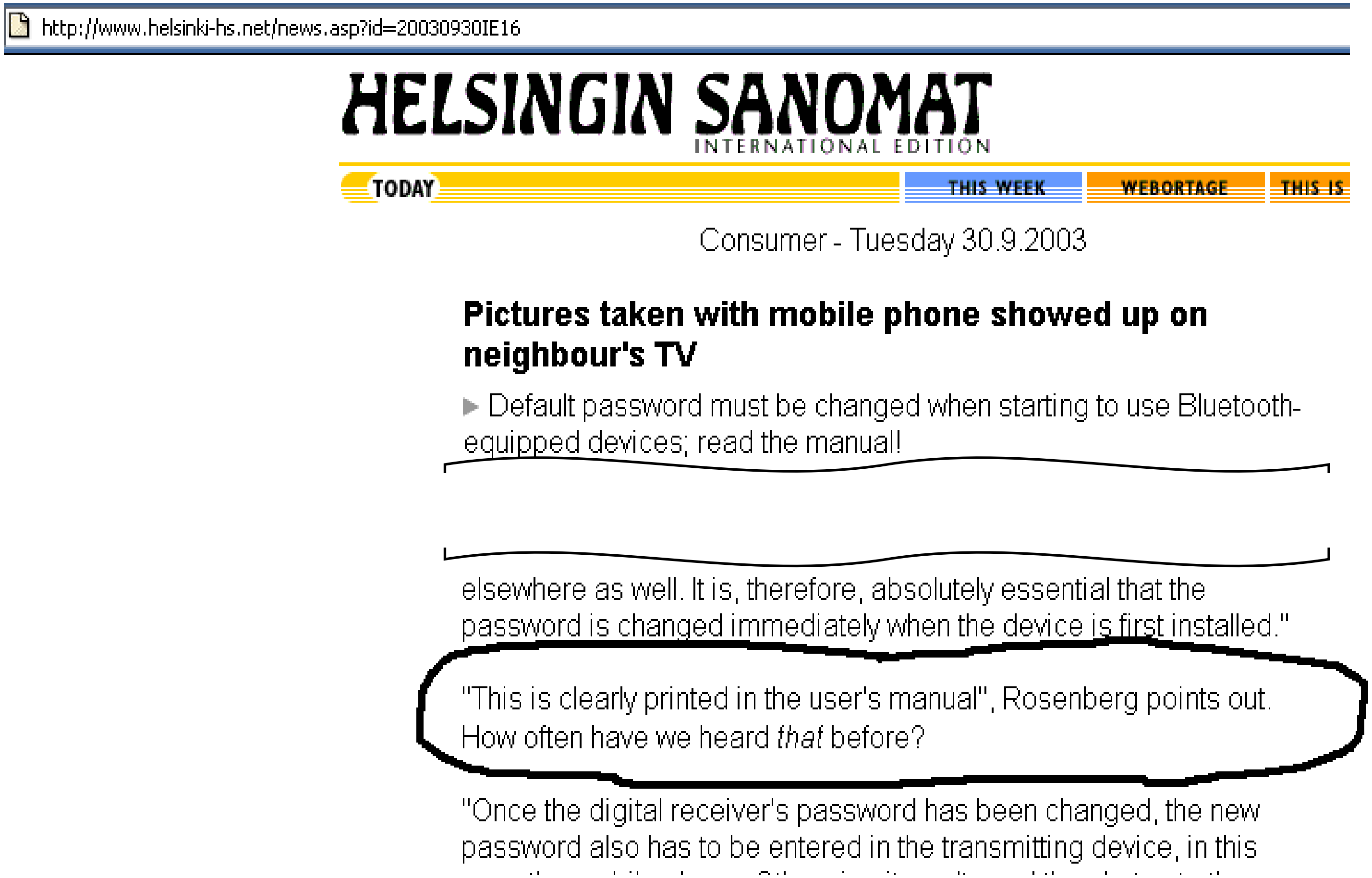
Security Weaknesses in Bluetooth

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

Consumer - Tuesday 30.9.2003

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

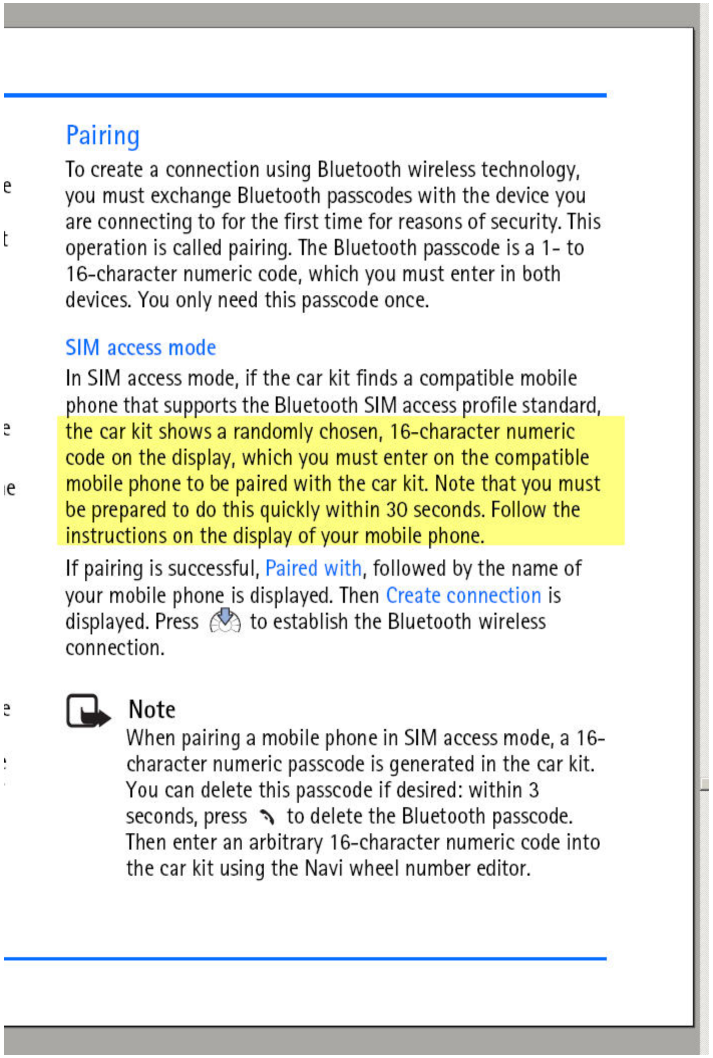
▶ Default password must be changed when starting to use Bluetooth-equipped devices; read the manual!

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 a car phone to retrieve and use session keys from a mobile phone smartcard

- Car kit requires higher level of security
  - users have to enter 16-character passcodes

More secure = Harder to use?

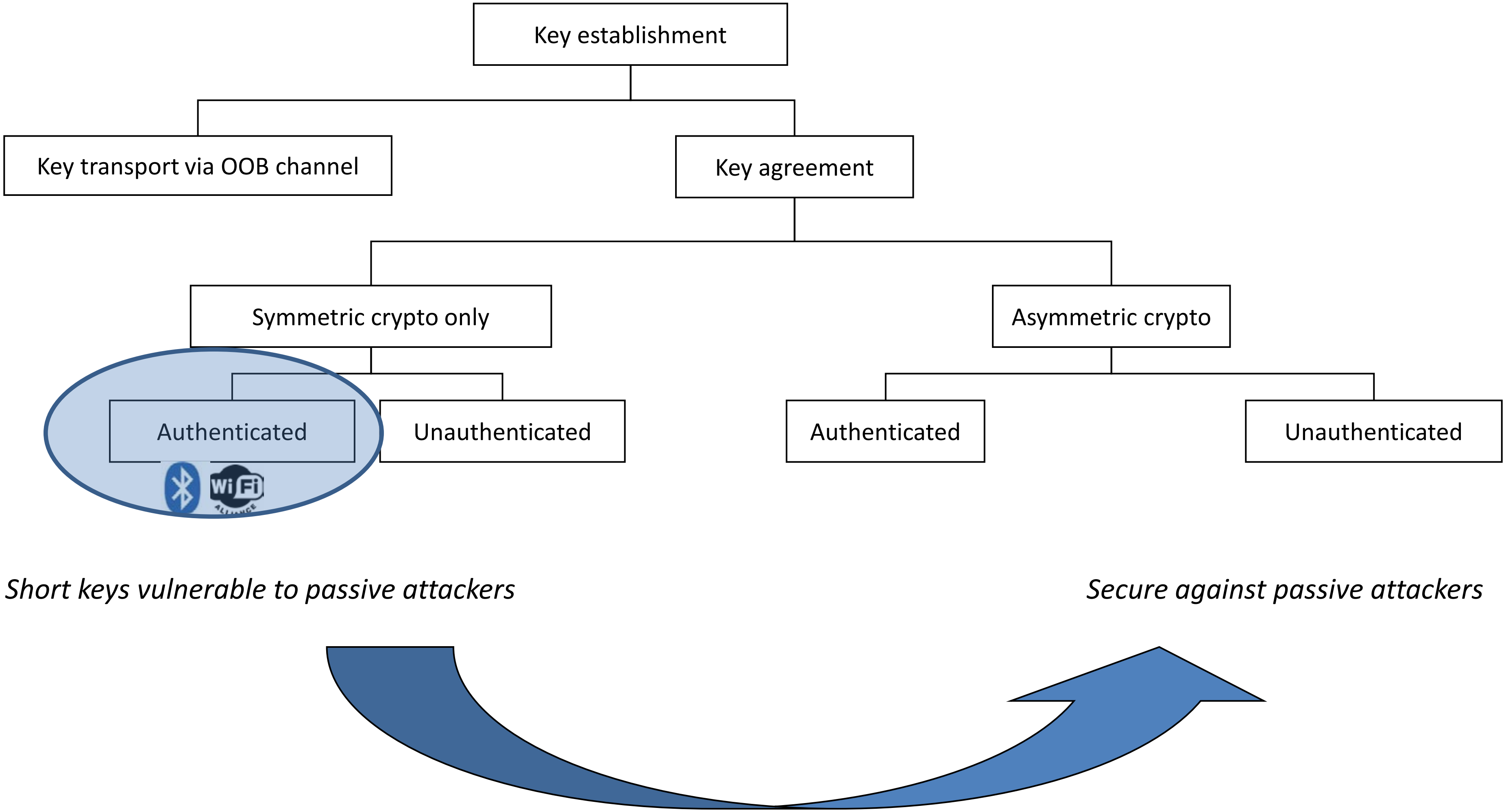
Cost:
Calls to Customer Support

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



Key establishment

Key transport via OOB channel

Key agreement

Symmetric crypto only

Asymmetric crypto

Authenticated

Unauthenticated

Authenticated

Unauthenticated

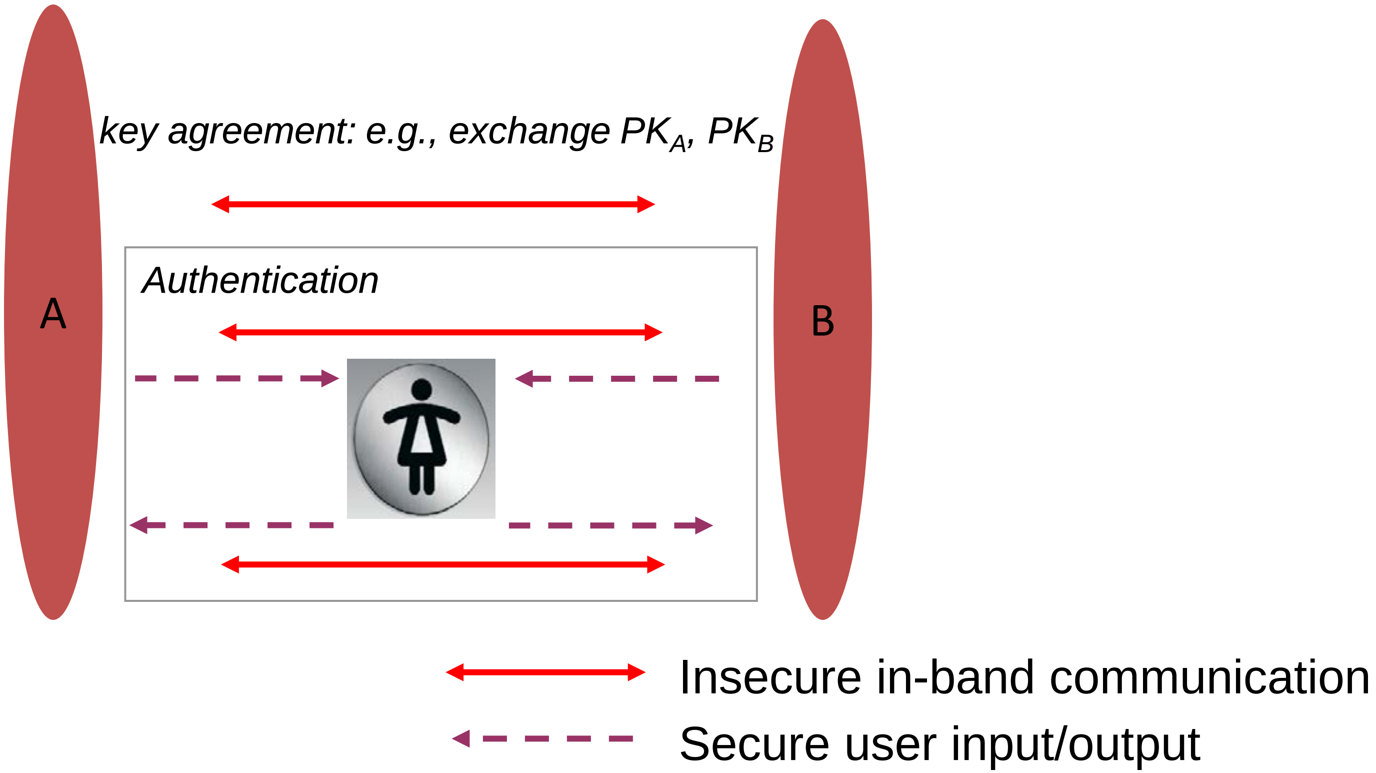
Short keys vulnerable to passive attackers

Secure against passive attackers

Authenticating key agreement

- Use an auxiliary channel to transfer information needed for authentication
- Two possibilities for realizing secure auxiliary channel
  - User assistance
  - Other out-of-band secure communication channels:
    - E.g., Near Field Communication, infrared, …

Authenticating key agreement: user-assisted



A

B

key agreement: e.g., exchange PK_A, PK_B

Authentication

Insecure in-band communication

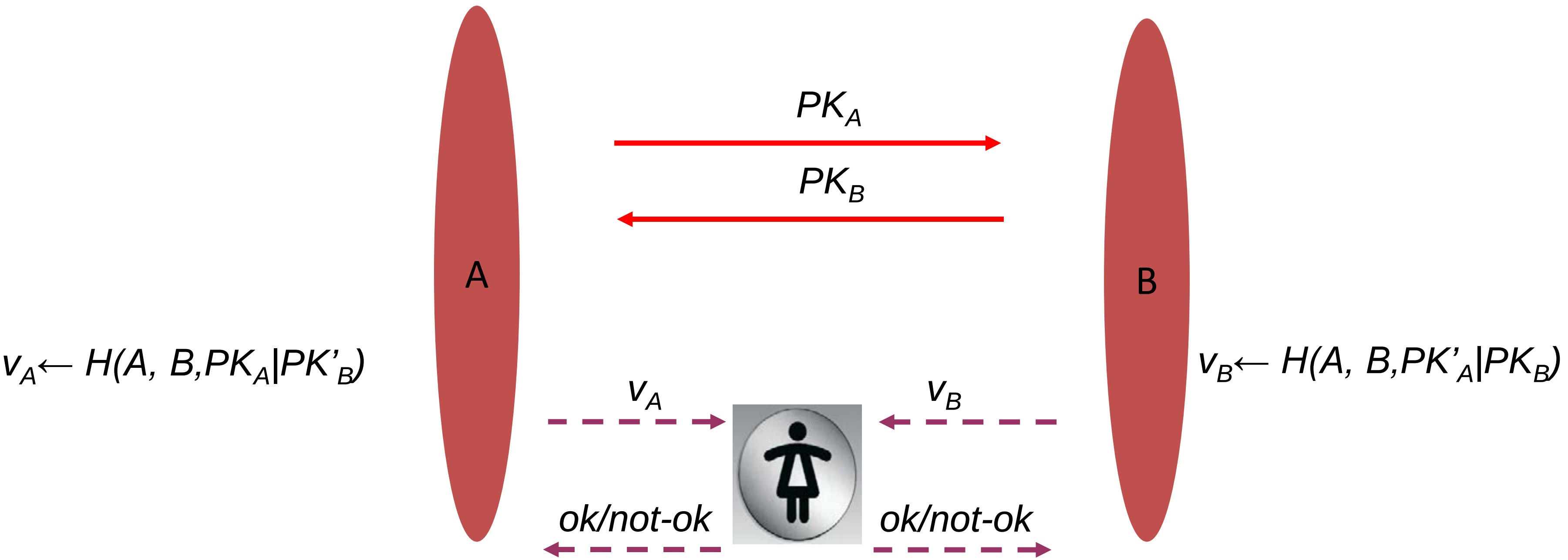
Secure user input/output

- 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

- Peer discovery by “user conditioning”: introduce a special first connect mode
  - E.g., Press a button to put device into the special mode
  - Demonstrative/indexical identification
- 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



PK_A

PK_B

A

B

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

v_A

v_B

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

ok/not-ok

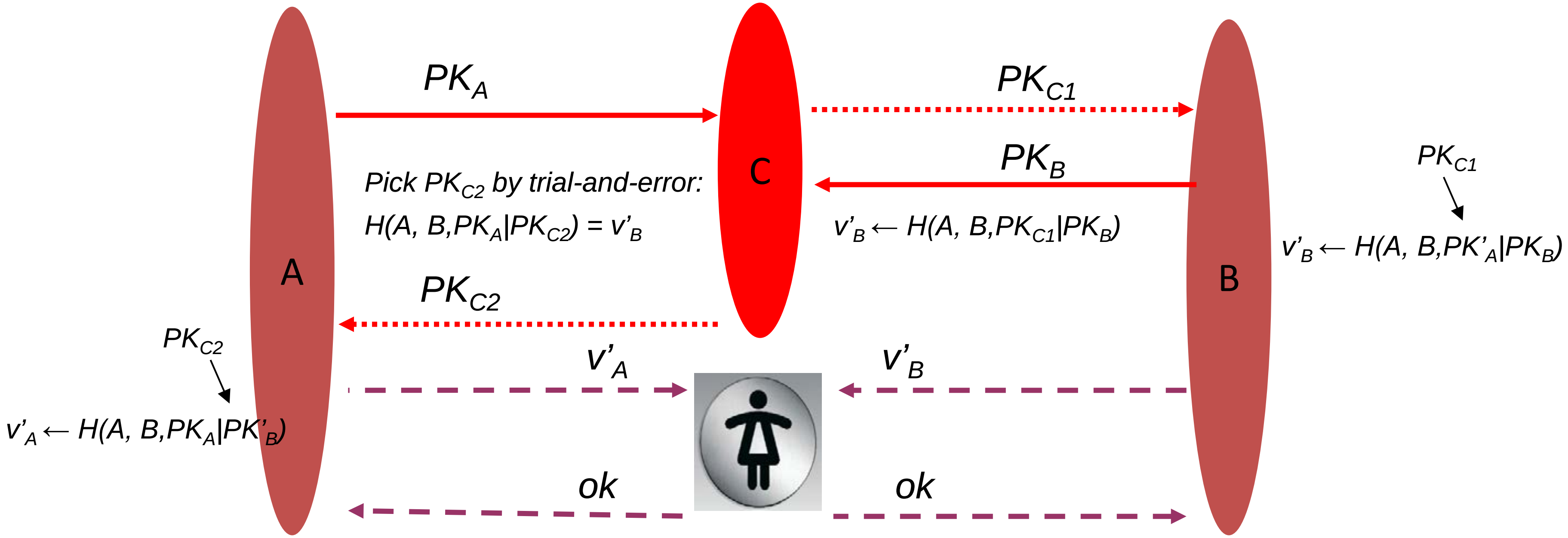
ok/not-ok

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

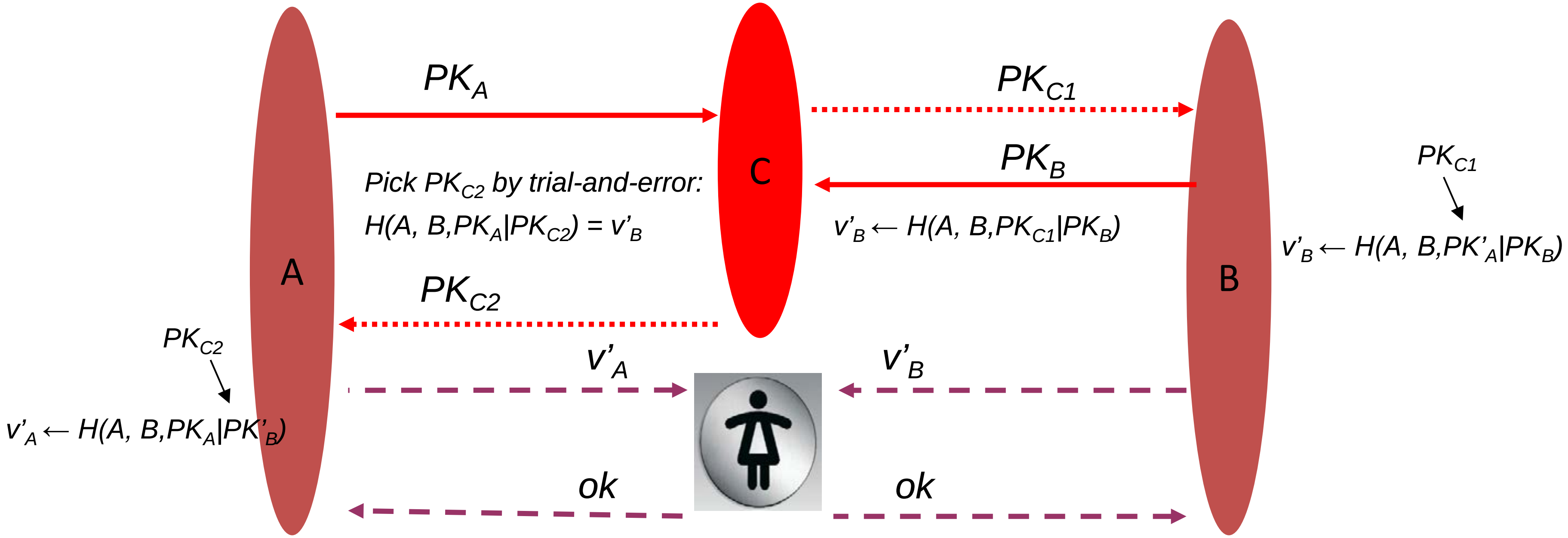
MitM in comparing short strings



PK_A
PK_{C1}
C
PK_B
PK_{C1}
Pick PK_{C2} by trial-and-error:
$H(A, B, PK_A|PK_{C2}) = v'_B$
$v'_B \leftarrow H(A, B, PK_{C1}|PK_B)$
$v'_B \leftarrow H(A, B, PK'_A|PK_B)$
A
B
PK_{C2}
PK_{C2}
v'_A
v'_B
$v'_A \leftarrow H(A, B, PK_A|PK'_B)$
ok
ok

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

MitM in comparing short strings


PK_A
PK_{C1}
PK_B
C
A
B
Pick PK_{C2} by trial-and-error:
$H(A, B, PK_A|PK_{C2}) = v'_B$
$v'_B \leftarrow H(A, B, PK_{C1}|PK_B)$
PK_{C1}
$v'_B \leftarrow H(A, B, PK'_A|PK_B)$
PK_{C2}
PK_{C2}
$v'_A \leftarrow H(A, B, PK_A|PK'_B)$
v'_A
v'_B
ok
ok

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

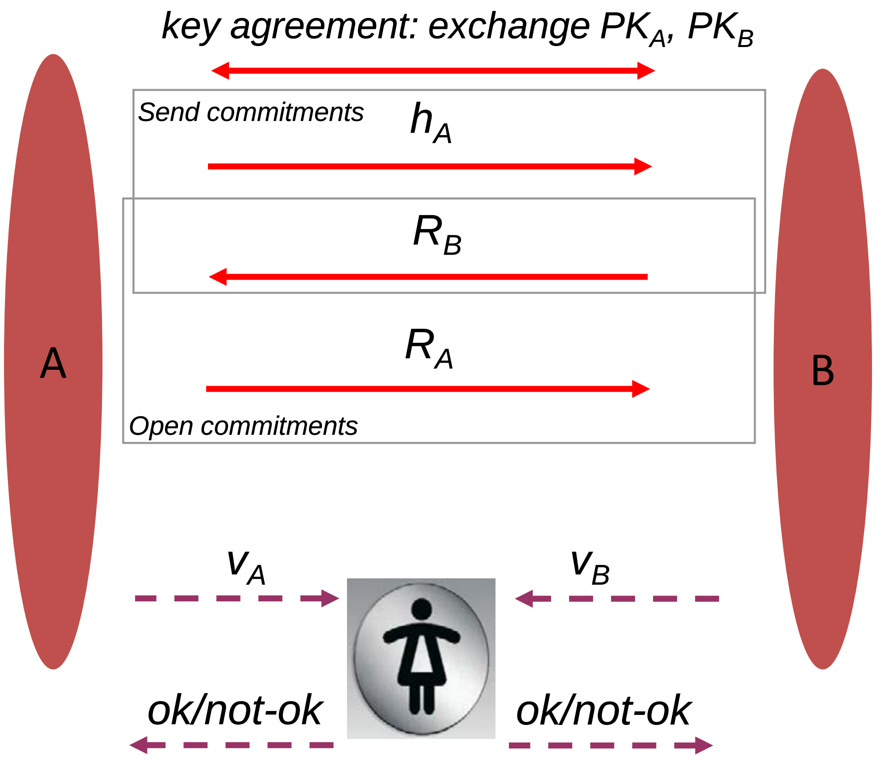
Authentication by comparing short strings

Choose long random R_A

Calculate commitment

$h_A \leftarrow h(A, R_A)$

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



key agreement: exchange PK_A, PK_B

Send commitments h_A

R_B

R_A

Open commitments

A

B

v_A

v_B

ok/not-ok

ok/not-ok

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

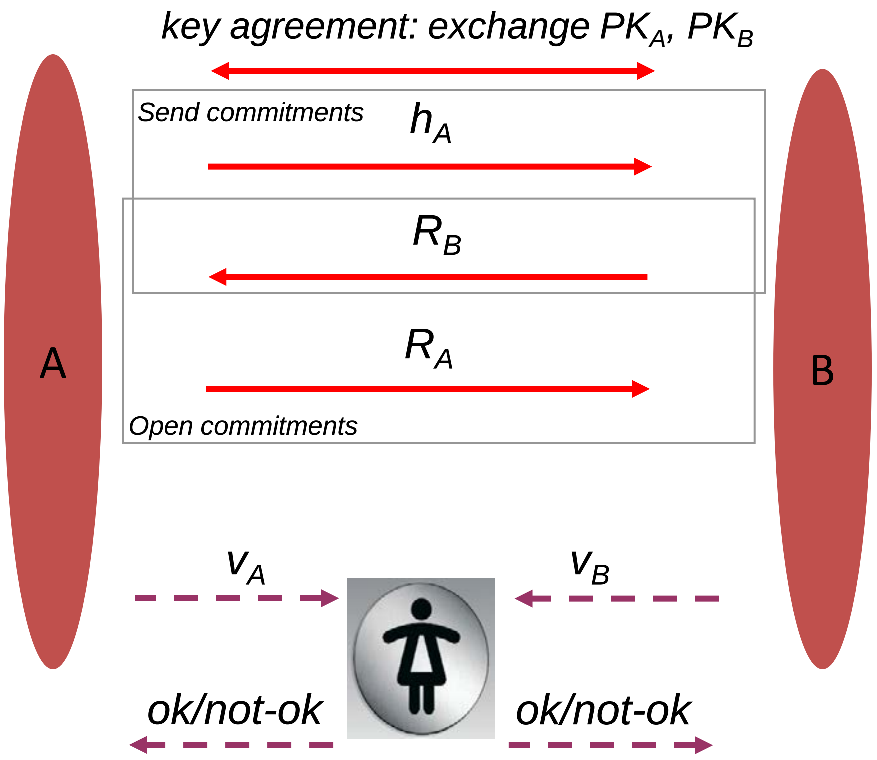
H*()* is a mixing function; in practice SHA-256 output truncated

Authentication by comparing short strings

Choose long random R_A

Calculate commitment

$h_A \leftarrow h(A, R_A)$



key agreement: exchange PK_A, PK_B

Send commitments h_A

R_B

R_A

Open commitments

A

B

v_A

v_B

ok/not-ok

ok/not-ok

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

Choose long random R_B

Verify commitment

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

Abort on mismatch

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

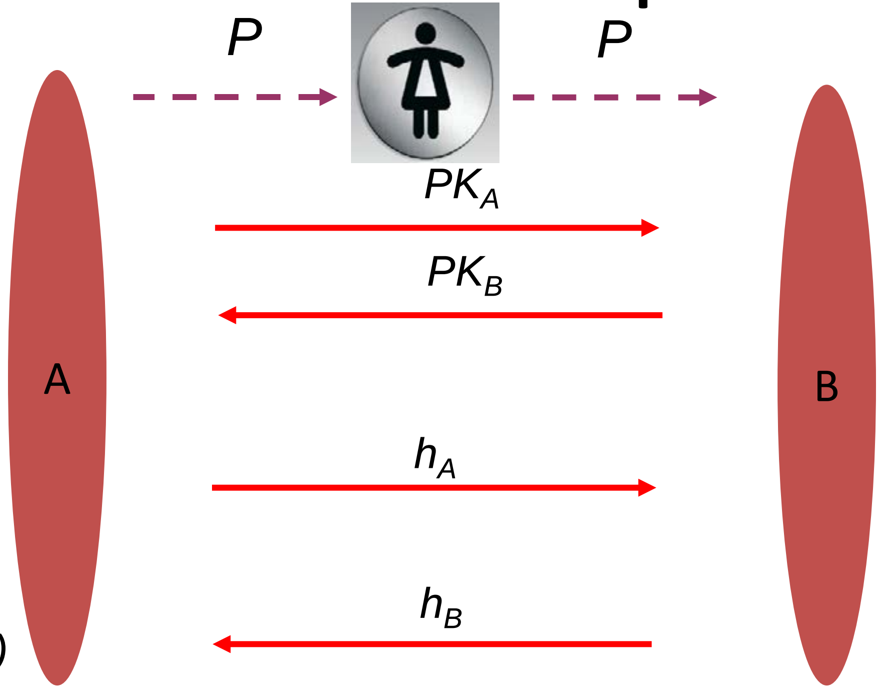
User approves acceptance if v_A and v_B match

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 a short passkey: a first attempt



$h_A \leftarrow MAC(A|PK_A|PK'_B, P)$

$h'_B \overset{?}{=} MAC(B|PK_A|PK'_B, P)$

$h'_A \overset{?}{=} MAC(A|PK'_A|PK_B, P)$

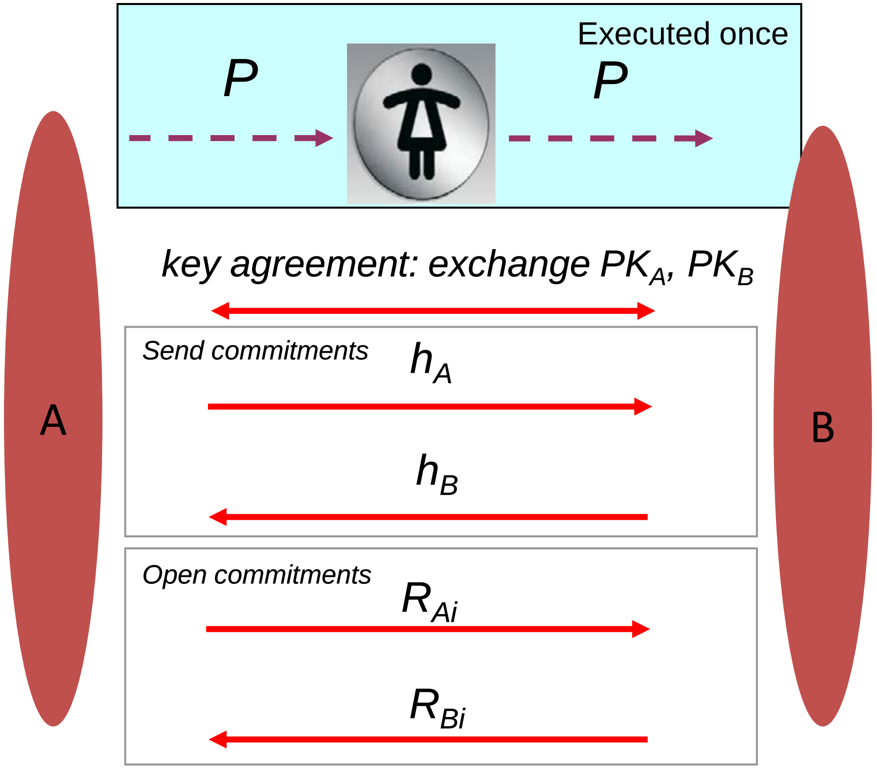
$h_B \leftarrow MAC(B|PK'_A|PK_B, P)$

P is a short passkey (e.g., 4 digits)

MAC() is a message authentication code: e.g., HMAC-SHA1

But a man-in-the-middle can easily defeat this protocol!

Authentication using interlocking short passkeys



Executed once

P P

Choose long random R_{Ai}

Choose long random R_{Bi}

key agreement: exchange PK_A, PK_B

Calculate commitment

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

Calculate commitment

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

Send commitments h_A

h_B

A

B

Open commitments R_{Ai}

R_{Bi}

Verify commitment

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

Verify commitment

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

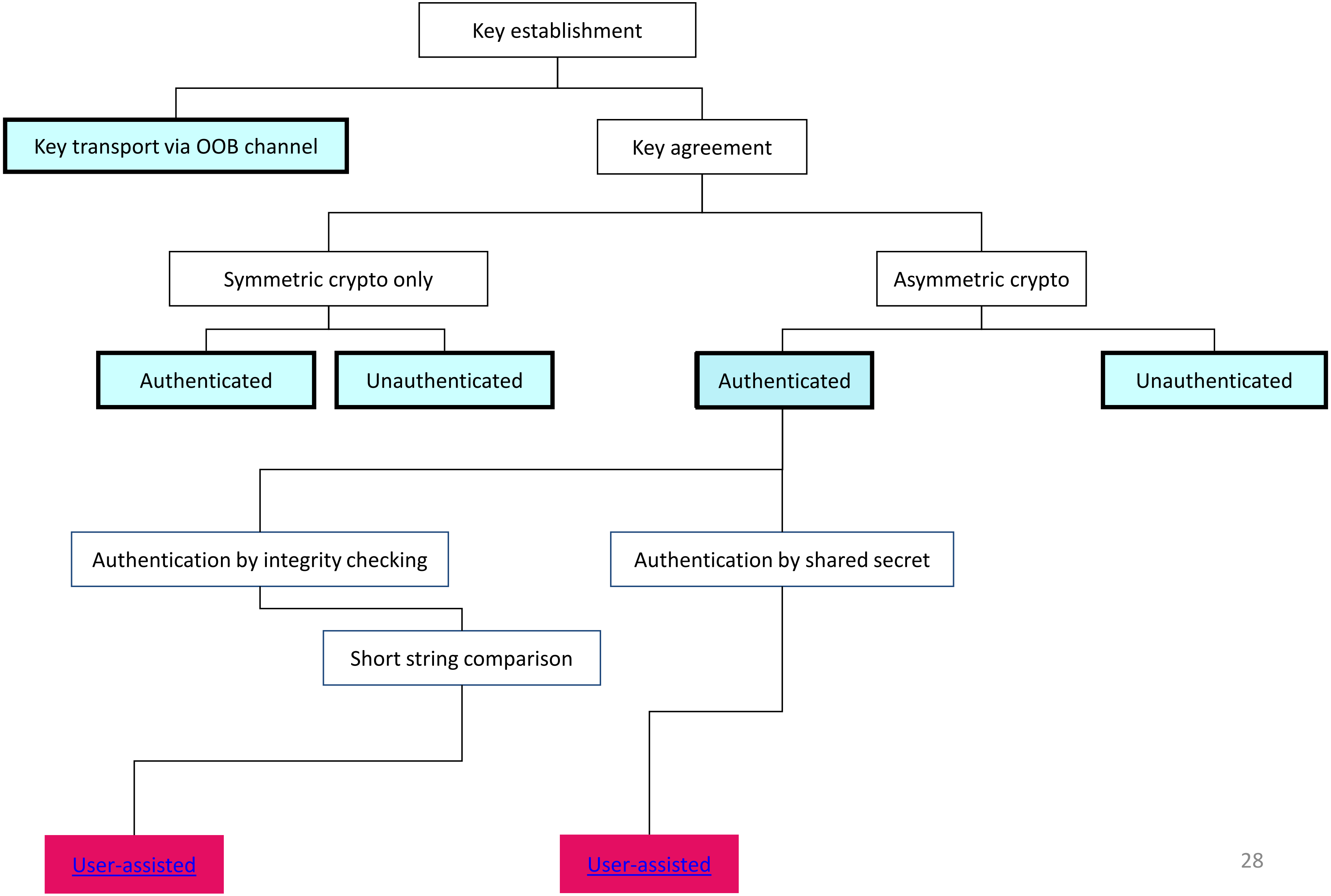
One-time passkey *P* is split into *k* parts ($l \geq k > 1$): next 4-round exchange repeated *k* times

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



Problems with user-as-secure-channel

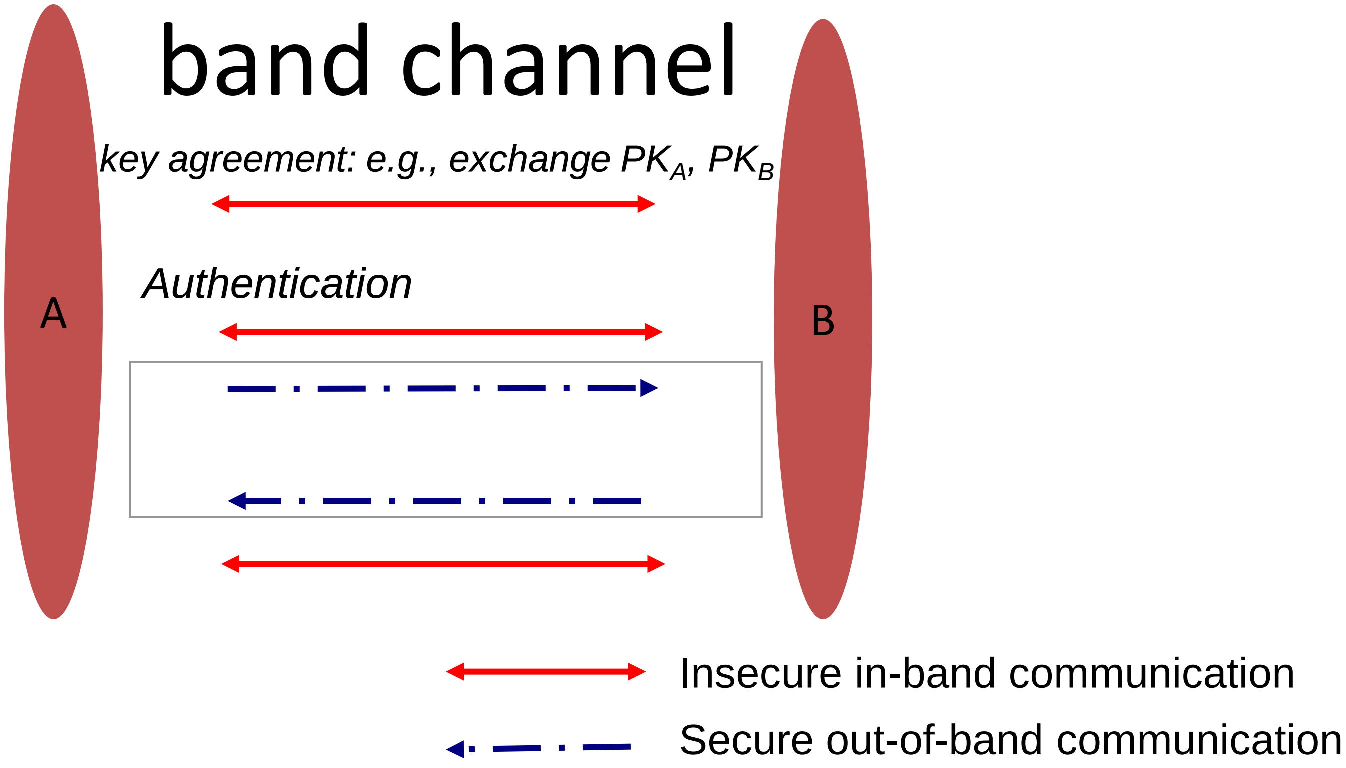
- Relies on availability of specific hardware (display, keypad, buttons, …)
- What about usability?

Skip to “problems with OOB channels”

Out-of-band secure channel

- Idea: use a physically secure channel to transfer security critical information
  - Minimize user involvement → better usability, … and security
- Peer discovery is intuitive
  - Demonstrative/indexical identification
- Channel must have certain security properties
  - integrity (tampering with messages can be detected)
  - Sometimes secrecy as well

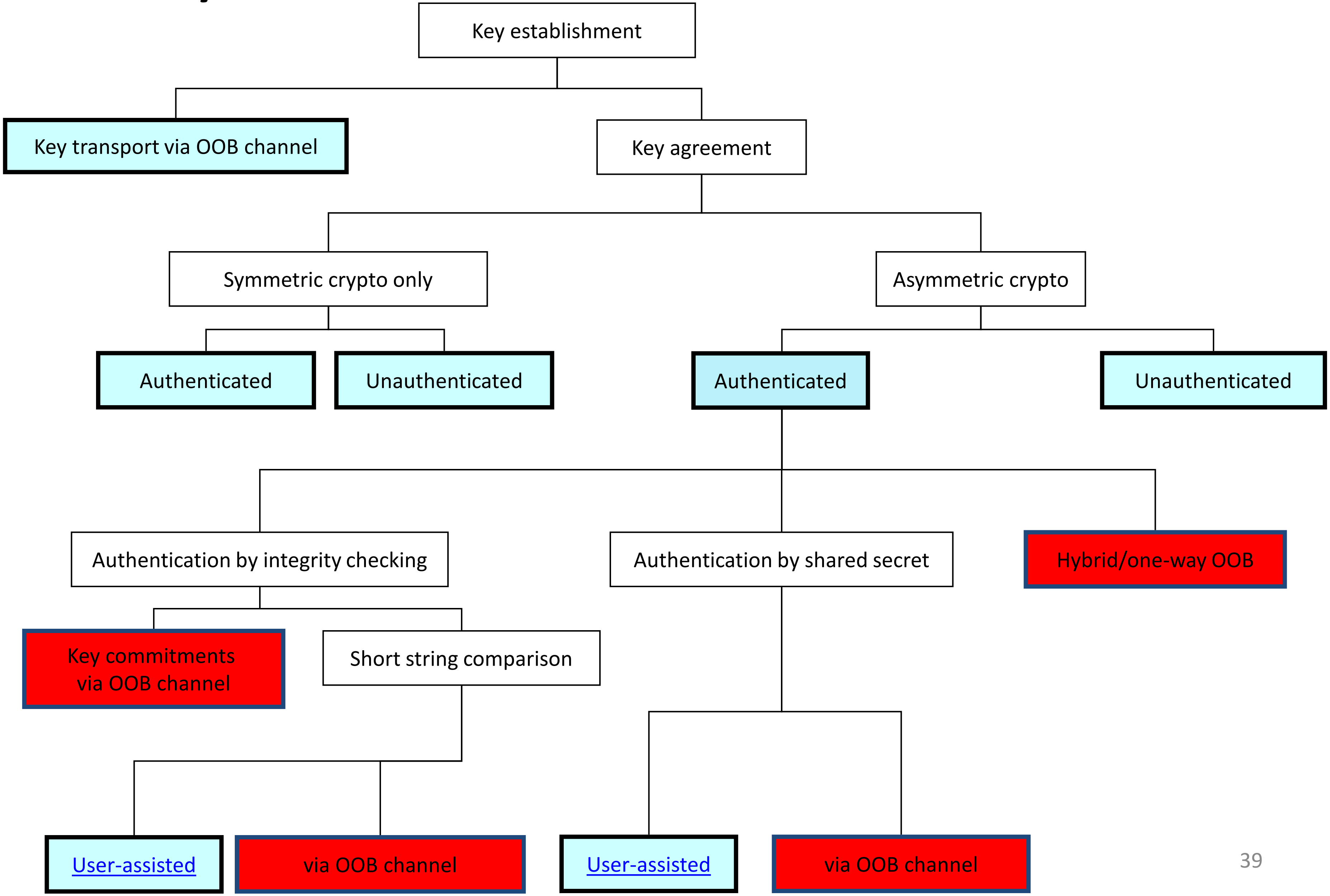
Authenticating key agreement: out-of-band channel


A
B
key agreement: e.g., exchange PK_A, PK_B
Authentication
Insecure in-band communication
Secure out-of-band communication

Different out-of-band channels have different

- Bandwidth
- Directionality (1-way or 2-way)
- Security properties (integrity-only, or integrity+secrecy)

Key establishment for first connect



Key establishment

Key transport via OOB channel

Key agreement

Symmetric crypto only

Asymmetric crypto

Authenticated

Unauthenticated

Authenticated

Unauthenticated

Authentication by integrity checking

Authentication by shared secret

Hybrid/one-way OOB

Key commitments via OOB channel

Short string comparison

User-assisted

via OOB channel

User-assisted

via OOB channel

Problems with out-of-band channels

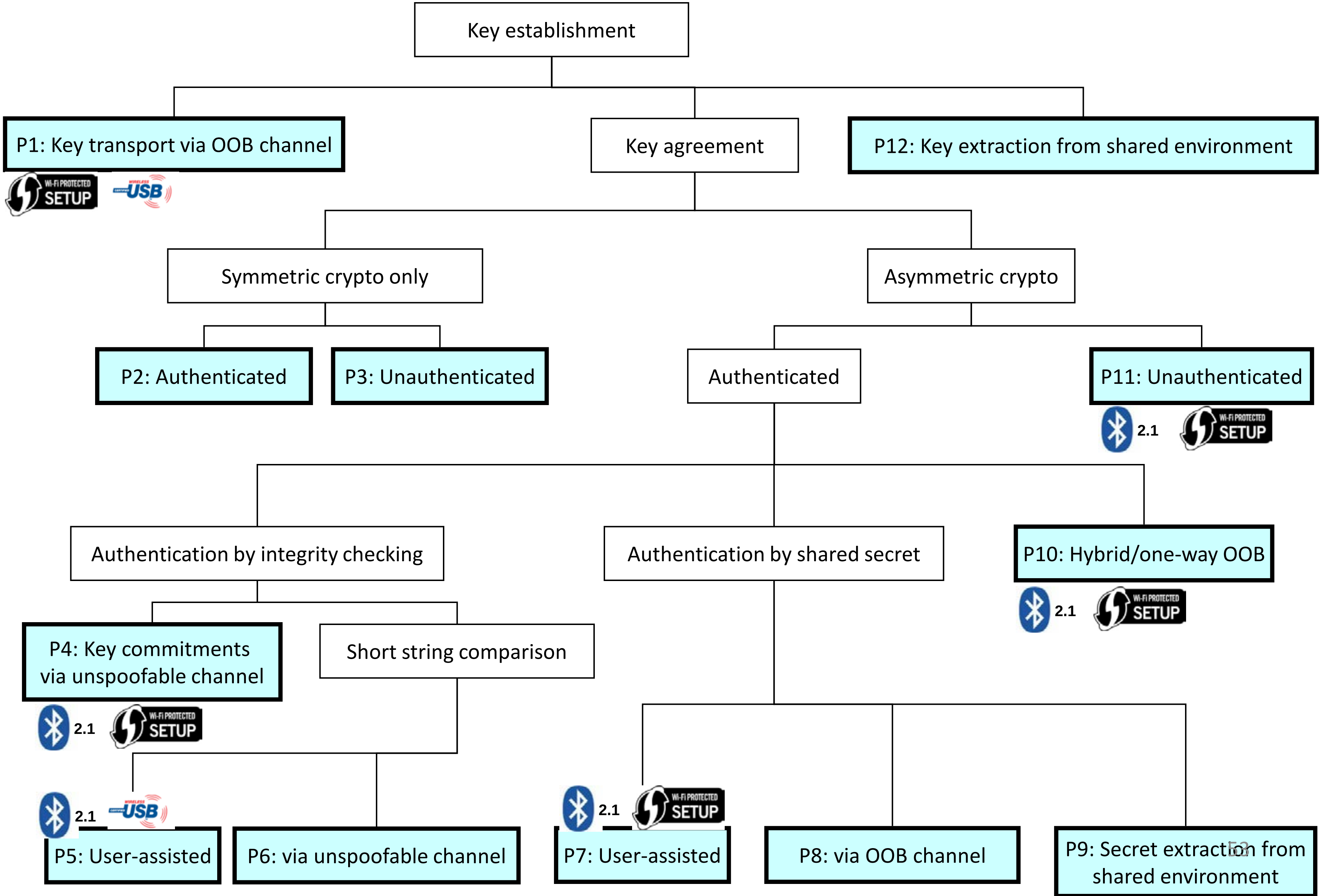
- Cost
  - Availability of specific (possibly new) hardware interfaces

- Deployability
  - Universally deployed auxiliary channel needed
  - Else how to discover common aux. channels between devices?
    - Leave-it-to-the-user: visible well-known logos
    - Negotiation protocol

Can we use the radio interface itself for authentication?

- In-band integrity checking
  - Assumption: genuine device emits energy during transmission; a distant attacker cannot easily drown this out
  - I-codes by Čagalj et al
- Common radio environment
  - Assumption: genuine devices hear the same radio signals; a distant attacker likely hears something different
  - Amigo by Varshavsky et al
- Spatial indistinguishability
  - Assumption: a distant attacker cannot tell which device is transmitting
  - Shake-them-up by Castelluccia et al

Key establishment for first connect



Key establishment

P1: Key transport via OOB channel

Key agreement

P12: Key extraction from shared environment

Symmetric crypto only

Asymmetric crypto

P2: Authenticated

P3: Unauthenticated

Authenticated

P11: Unauthenticated

Authentication by integrity checking

Authentication by shared secret

P10: Hybrid/one-way OOB

P4: Key commitments via unspoofable channel

Short string comparison

P5: User-assisted

P6: via unspoofable channel

P7: User-assisted

P8: via OOB channel

P9: Secret extraction from shared environment

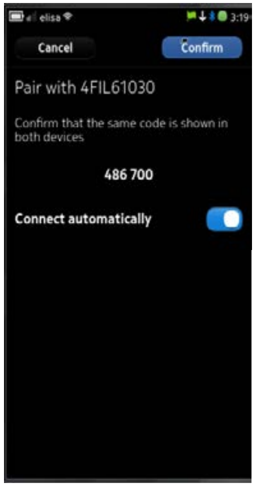
Key establishment for first connect ~2008

	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)
- Recent research exploiting properties of radio communication looks promising
  - Čapkun et al/TDSC 2008:5(4), Gollakota et al/Usenix Security ‘11

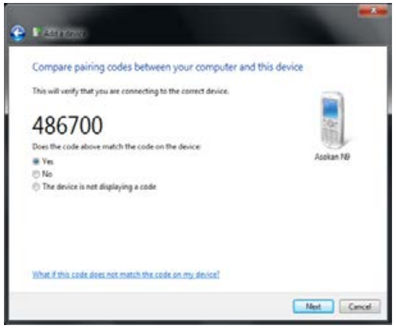


Pair with 4FIL61030

Confirm that the same code is shown in both devices

486 700

Connect automatically



Compare pairing codes between your computer and this device

This will verify that you are connecting to the correct device.

486700

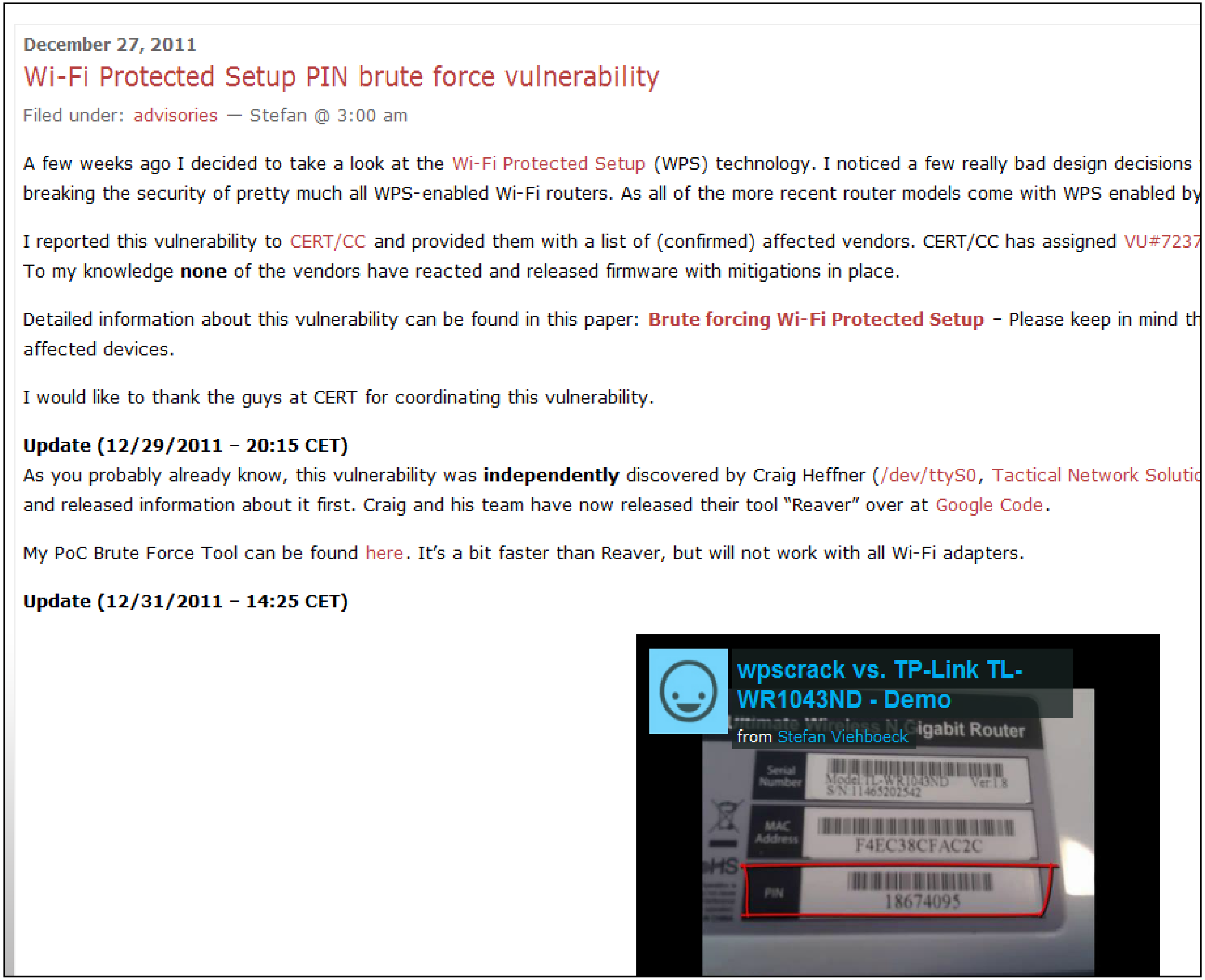
Does the code above match the code on the device:

● Yes

○ No

○ The device is not displaying a code

What if this code does not match the code on my device?



December 27, 2011

Wi-Fi Protected Setup PIN brute force vulnerability

Filed under: advisories — Stefan @ 3:00 am

A few weeks ago I decided to take a look at the Wi-Fi Protected Setup (WPS) technology. I noticed a few really bad design decisions breaking the security of pretty much all WPS-enabled Wi-Fi routers. As all of the more recent router models come with WPS enabled by

I reported this vulnerability to CERT/CC and provided them with a list of (confirmed) affected vendors. CERT/CC has assigned VU#7237 To my knowledge **none** of the vendors have reacted and released firmware with mitigations in place.

Detailed information about this vulnerability can be found in this paper: **Brute forcing Wi-Fi Protected Setup** – Please keep in mind th affected devices.

I would like to thank the guys at CERT for coordinating this vulnerability.

Update (12/29/2011 – 20:15 CET)
As you probably already know, this vulnerability was **independently** discovered by Craig Heffner (/dev/ttyS0, Tactical Network Solutio and released information about it first. Craig and his team have now released their tool "Reaver" over at Google Code.

My PoC Brute Force Tool can be found here. It's a bit faster than Reaver, but will not work with all Wi-Fi adapters.

Update (12/31/2011 – 14:25 CET)

wpscrack vs. TP-Link TL-WR1043ND - Demo
from Stefan Viehboeck
Ultimate Wireless N Gigabit Router
Serial Number
Model:TL-WR1043ND Ver:1.8
S/N:11465202542
MAC Address
F4EC38CFAC2C
PIN
18674095

http://sviehb.wordpress.com/2011/12/27/wi-fi-protected-setup-pin-brute-force-vulnerability/
http://www.kb.cert.org/vuls/id/723755

Granting Permissions to Apps

Apps and Permissions

- Access control: regulate what subjects can do
- On single-user systems (like mobile devices) subjects are programs
- Popular mobile software platform security architectures are permission-based
  - Assign permissions to programs (apps)
  - Check permissions at time of access

Granting permissions to apps

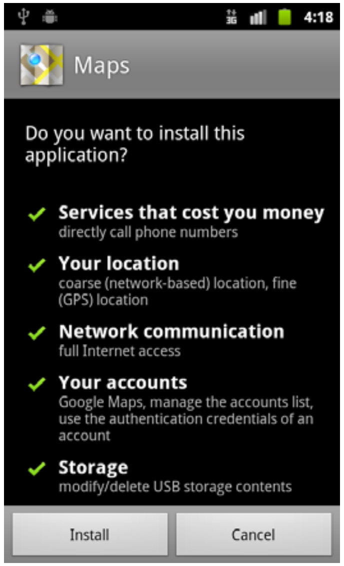
Punt to user

Decide centrally (mostly)

”Is this App Safe?”

A Large Scale Study on Application Permissions and Risk Signals

(WWW 2012)



4:18

Maps

Do you want to install this application?

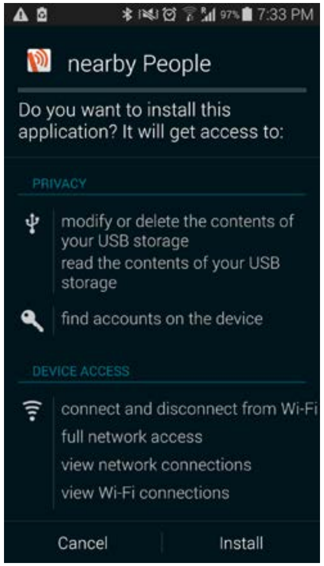
✓ **Services that cost you money**
directly call phone numbers

✓ **Your location**
coarse (network-based) location, fine (GPS) location

✓ **Network communication**
full Internet access

✓ **Your accounts**
Google Maps, manage the accounts list, use the authentication credentials of an account

✓ **Storage**
modify/delete USB storage contents



97% 7:33 PM

nearby People

Do you want to install this application? It will get access to:

PRIVACY

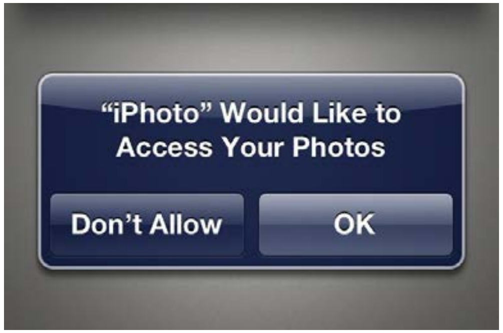
modify or delete the contents of your USB storage
read the contents of your USB storage

find accounts on the device

DEVICE ACCESS

connect and disconnect from Wi-Fi
full network access
view network connections
view Wi-Fi connections

Android



“iPhoto” Would Like to Access Your Photos

iOS, Windows Phone, (late) Symbian

Granting permissions to apps

Punt to user

- Personalized
- …
- Hard-to-use
- Ill-informed decisions
- Habituation
- …

Decide centrally

- Ease-of-use
- …
- Not personalized
- Potential liability
- …

Cost: user dissatisfaction

How to improve permission granting?

1. Provide more context in prompts
   - Annotations with additional information
2. Time of granting: Install time vs. Run time
3. Implicit granting via trusted UIs
4. Automatic granting + auditability

1. Annotations

- Show additional annotations to help user make more informed decisions
- Information obtained by
  - Analyzing app
  - Expert and crowdsourced rating
  - …

Annotations from analysis

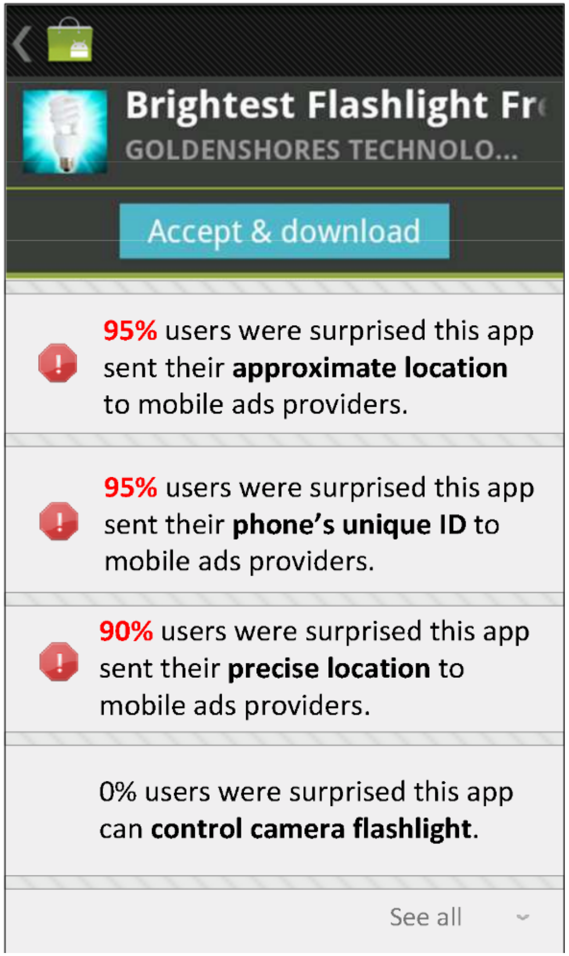
- Problem: privacy risk depends on context
  - E.g., “Location”: ok for maps, not for flashlight
  - Privacy at risk if user’s expectations not met
- Idea:
  - *Training*: Tell some users what app does and ask if that matches their expectations
  - *Use*: Annotate permission prompts (for other users) with results from training

Lin et al, “Expectation and Purpose: Understanding Users’ Mental Models of Mobile App Privacy through Crowdsourcing “

Where the info comes from

- Step #1: Get permissions from manifests
- Step #2: Figure out how data is used
  - Analyse using TaintDroid (tracks where data goes)
  - Categorize uses: core functionality / secondary (e.g. tagging, sharing) / targeted ads
- Step #3: Ask users about their reactions
  - Do you expect this app to use …
  - Are you uncomfortable with it using X to support Y
  - Participants recruited on Amazon Mechanical Turk

Showing cues to users

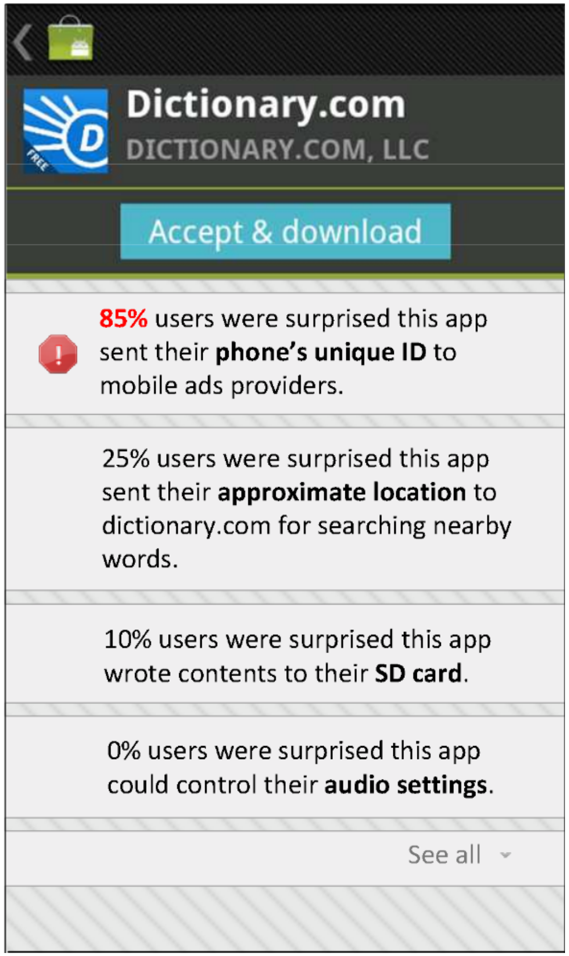


95% users were surprised this app sent their **approximate location** to mobile ads providers.

95% users were surprised this app sent their **phone's unique ID** to mobile ads providers.

90% users were surprised this app sent their **precise location** to mobile ads providers.

0% users were surprised this app can **control camera flashlight**.



85% users were surprised this app sent their **phone's unique ID** to mobile ads providers.

25% users were surprised this app sent their **approximate location** to dictionary.com for searching nearby words.

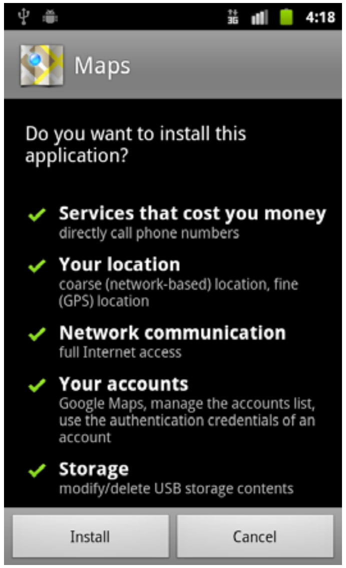
10% users were surprised this app wrote contents to their **SD card**.

0% users were surprised this app could control their **audio settings**.

Example permission UI from Lin et al, 2012

2. Time of granting

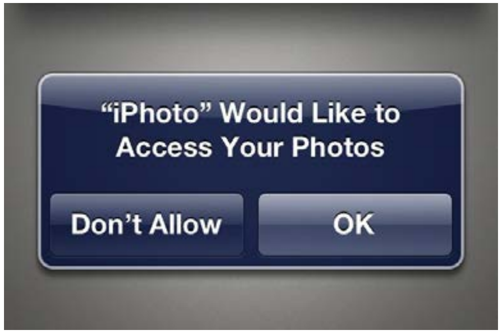
Install time *vs.* *Run time*



Do you want to install this application?

- ✓ **Services that cost you money**
  directly call phone numbers
- ✓ **Your location**
  coarse (network-based) location, fine (GPS) location
- ✓ **Network communication**
  full Internet access
- ✓ **Your accounts**
  Google Maps, manage the accounts list, use the authentication credentials of an account
- ✓ **Storage**
  modify/delete USB storage contents

- more time to think
- less disruptive
- no contextual info.



"iPhoto" Would Like to Access Your Photos

- more contextual info.
- more fine-grained
- more intrusive

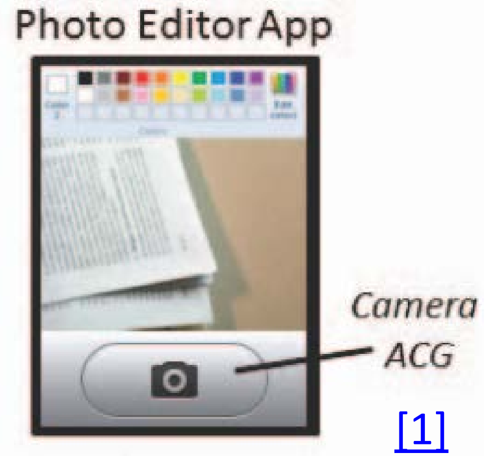
3. Implicit Permission Granting

Trusted UI

- Trusted path to user
  - Trusted widgets
  - E.g. PIN/login input screen
- Not forgeable nor obscurable by apps
  - Hardware support needed
- Other application areas:
  - User authentication
  - Transaction confirmation
  - Provisioning

Trusted permission widgets

- Goal: Permission requests should be
  - In context – informed decisions
  - Least-privilege – not ”take photos at any time”
  - Supporting user task – not interrupt it
- Idea: trusted widget for action + permission
  - ”Camera trigger”
  - ”Microphone record button”
  - *access control gadget*

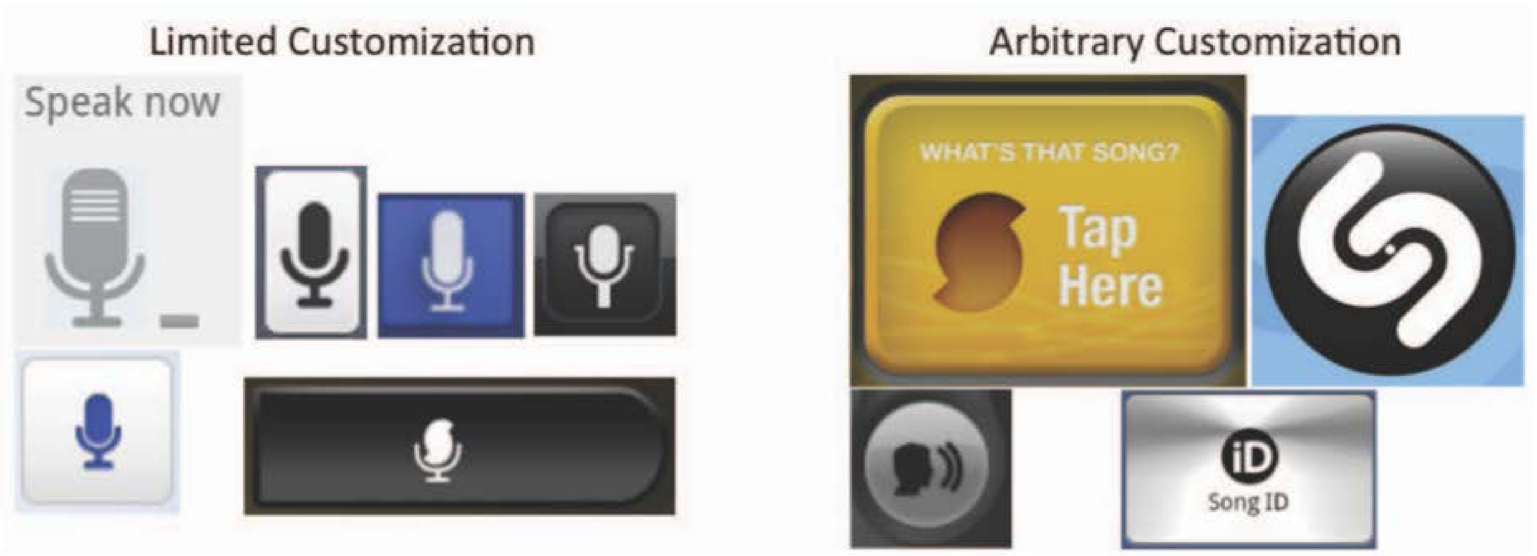


[1]

[1] *Roesner et al, “User-driven access control: Rethinking permission granting in modern operating systems”*

Permission widgets: visuals

- Grant: once, session, scheduled, permanent…
- Convey semantics clearly to user
- Must be identifiable – UI customization?



[1]

4. Automatic granting

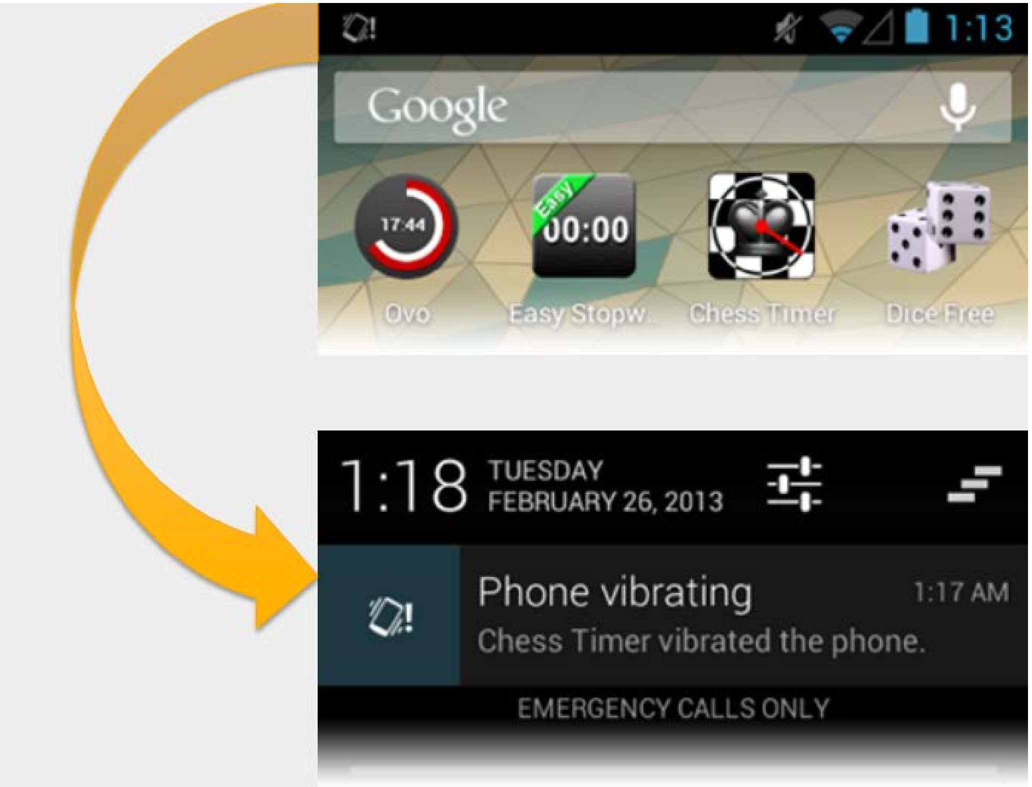
Grant requested permissions

- ... for low risk and reversible permissions
- ... but allow for **auditability**
  - Letting user figure out if app abuses permission

Thompson et al, “When it’s better to ask forgiveness than get permission: attribution mechanisms for smartphone resources”

Allowing for auditability

Show who was responsible for a change (e.g., notification)
e.g., notification shows which app is vibrating phone

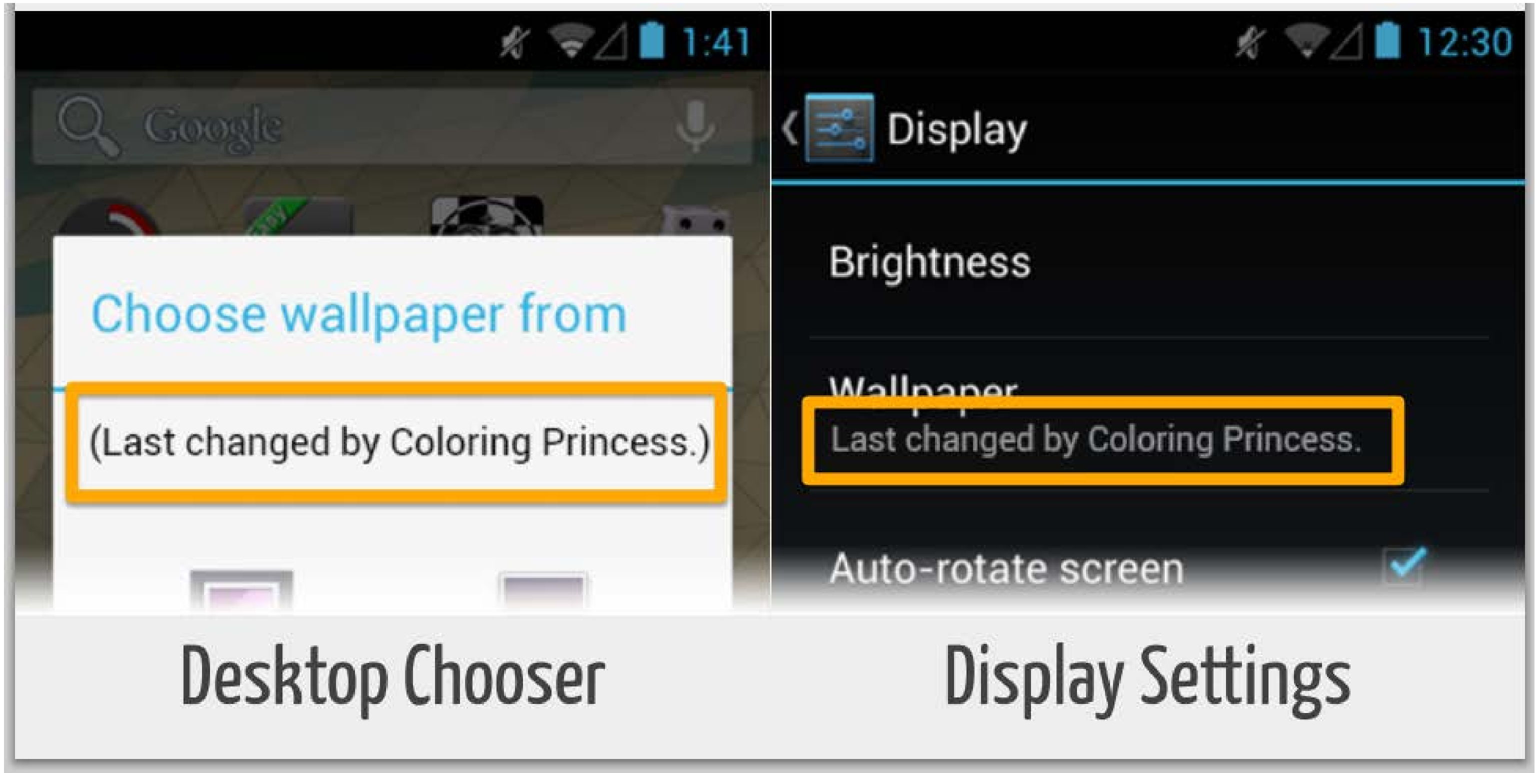


Phone vibrating 1:17 AM
Chess Timer vibrated the phone.

Allowing for auditability

Show who was responsible for a change (e.g., settings):

e.g., display settings shows which app changed wall paper



(Last changed by Coloring Princess.)

Last changed by Coloring Princess.

Desktop Chooser

Display Settings

Is attribution effective?

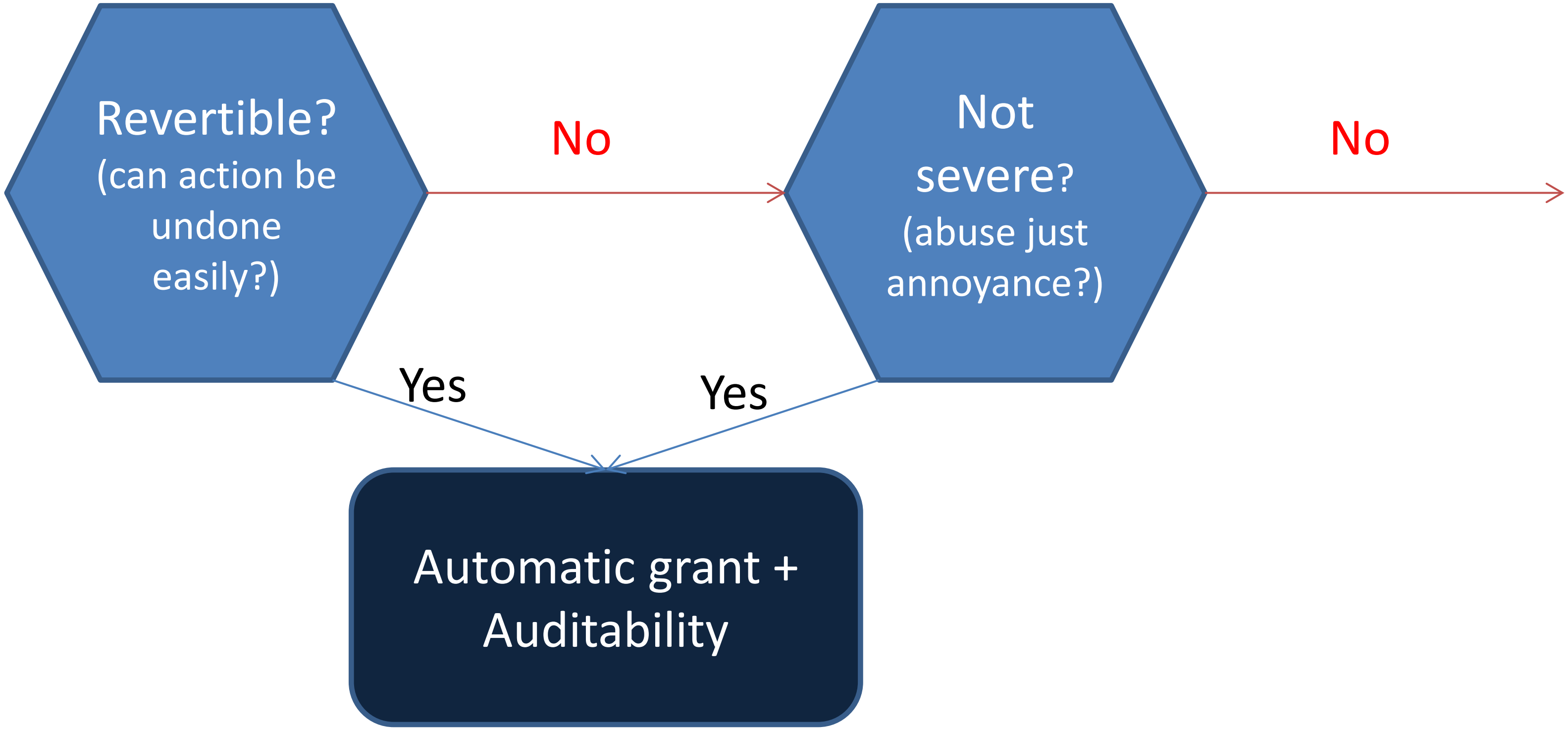
- Will users notice attribution indicators?
- Will they identify the apps responsible?

- Controlled laboratory study

Design Choices for Permission Granting

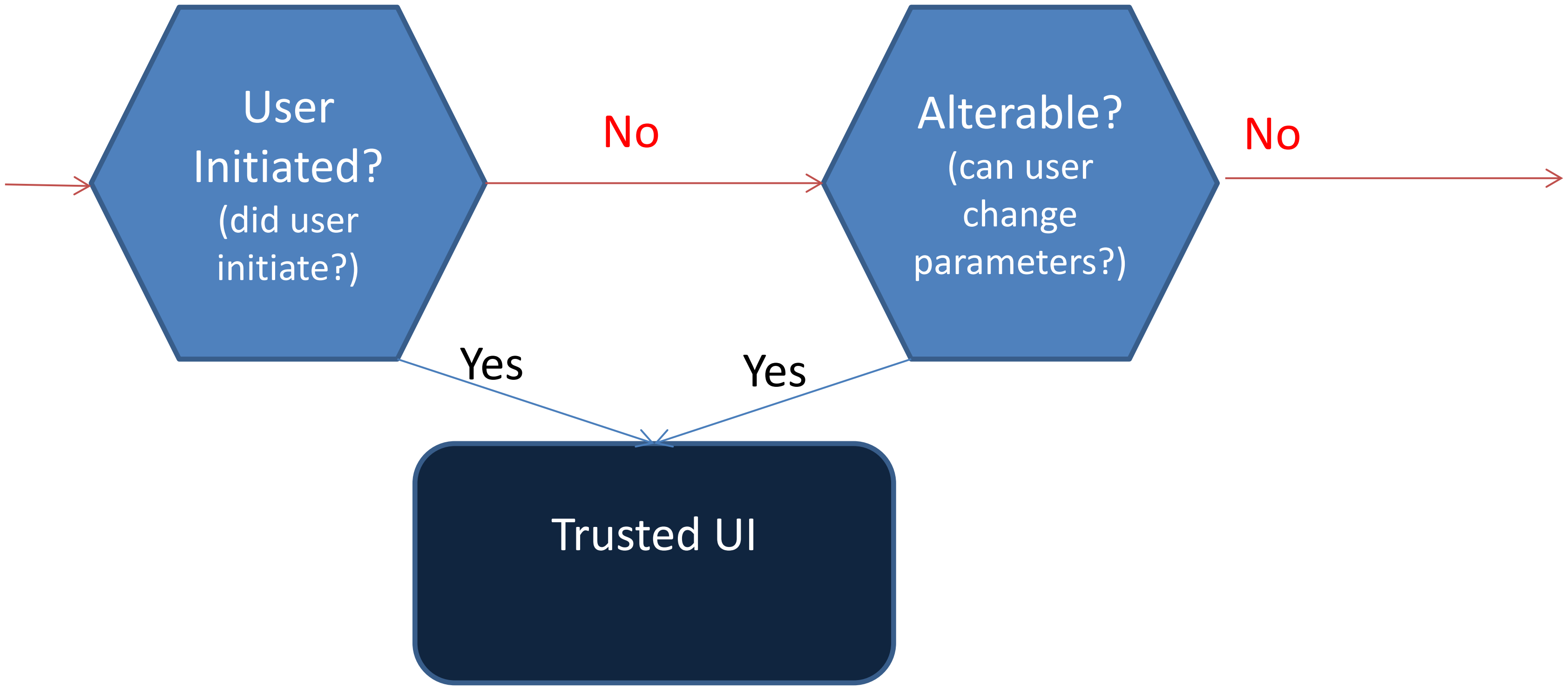
- Via user prompt
  - Install time
  - Run time
- Implicitly, via trusted UI interaction
- Automatically (with auditability)

Choosing granting mechanism (1/3)


Revertible?
(can action be undone easily?)
No
Not severe?
(abuse just annoyance?)
No
Yes
Yes
Automatic grant + Auditability

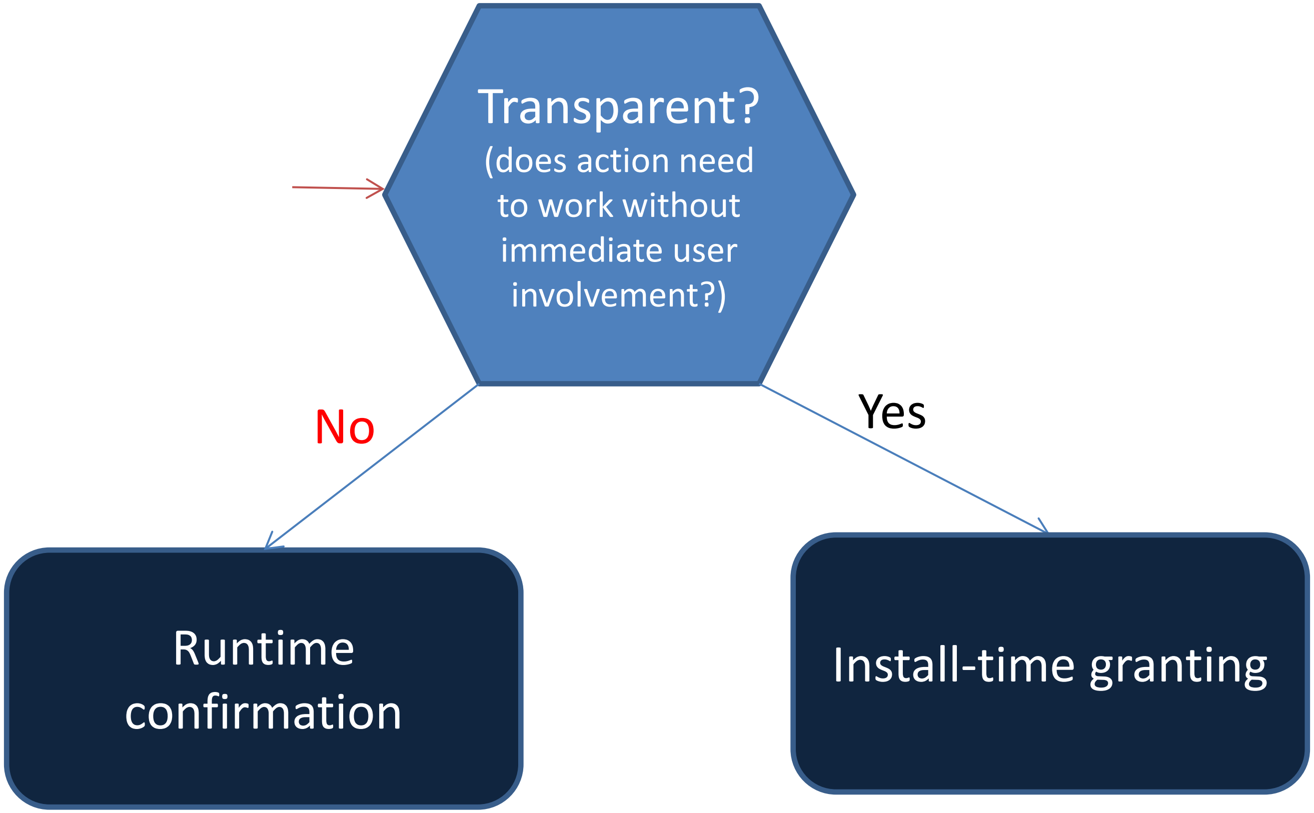
Adapted from “How to Ask for Permission” Porter Felt et al, HotSec ‘12

Choosing granting mechanism (2/3)


User Initiated? (did user initiate?)
No
Alterable? (can user change parameters?)
No
Yes
Yes
Trusted UI

Adapted from "How to Ask for Permission" Porter Felt et al, HotSec '12

Choosing granting mechanism (3/3)



Transparent?
(does action need to work without immediate user involvement?)

No

Yes

Runtime confirmation

Install-time granting

Adapted from “How to Ask for Permission” Porter Felt et al, HotSec ‘12

Permission Granting - Summary

- Essential component of mobile platform security
- Current methods are improving, but still fall short

Why is usable mobile security different?

Your mobile phone: Not a smaller version of your PC

!=

Your mobile phone: Not a smaller version of your PC

Mobile phone applications have different requirements due to

1. Smaller physical screen size

   → Less room for security indicators, notifications etc.

Your mobile phone: Not a smaller version of your PC

Mobile phone applications have different requirements due to

1. Smaller physical screen size
2. Different input mechanisms

Directional pad + keyboard

Touch screen

Keyboard + mouse + ...

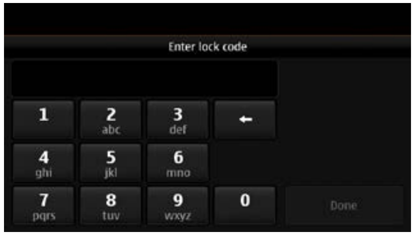
Your mobile phone: Not a smaller version of your PC

Mobile phone applications have different requirements due to

1. Smaller physical screen size
2. Different input mechanisms
3. Limited battery life
4. More prone to theft/loss
5. Slower and less reliable network connectivity
6. (Comparatively) limited computational power

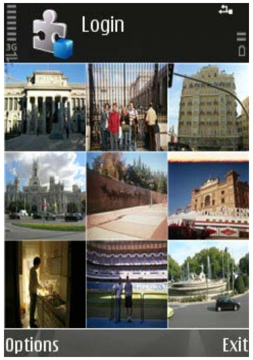
Other usable security problems

Local user authentication

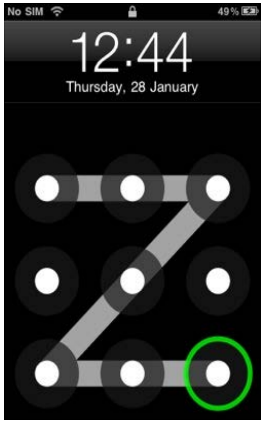


Need alternatives that are:

- Faster
- More enjoyable
- Secure enough



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



Biometrics

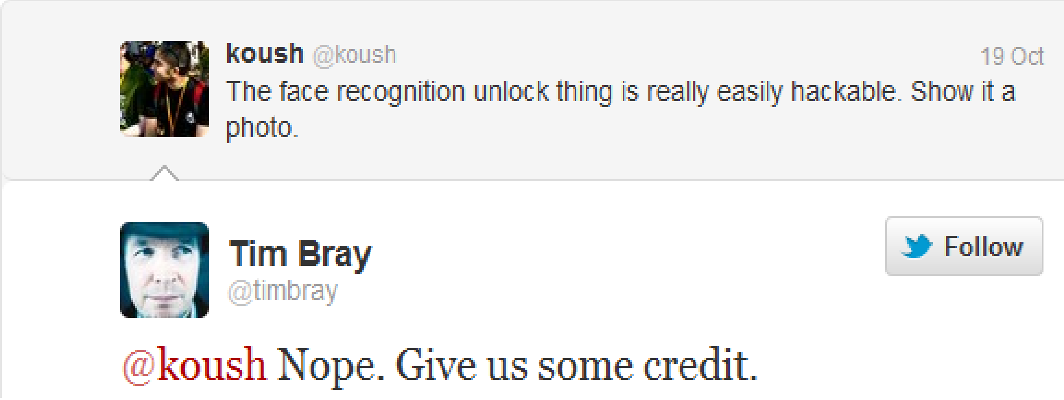
Wearables

?

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

Cost: weak PINs

Local user authentication: a cautionary tale

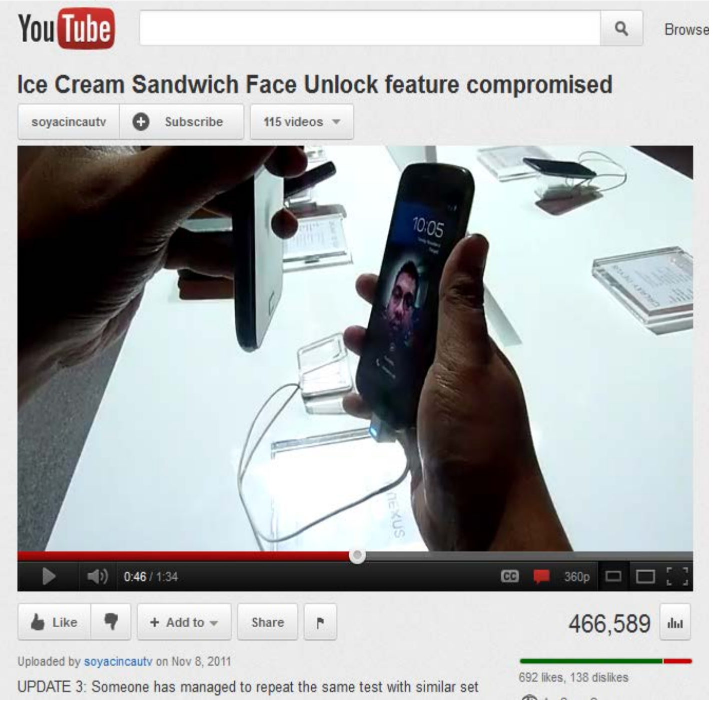


koush @koush 19 Oct

The face recognition unlock thing is really easily hackable. Show it a photo.

Tim Bray @timbray

@koush Nope. Give us some credit.



Ice Cream Sandwich Face Unlock feature compromised

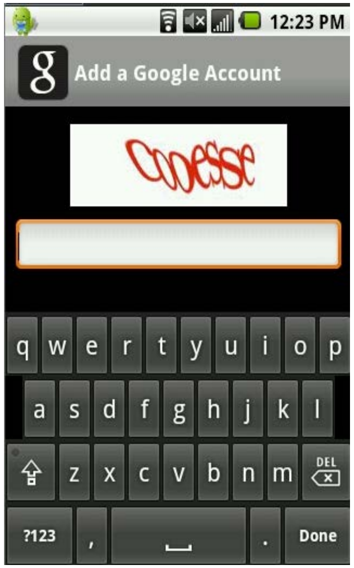
Uploaded by soyacincautv on Nov 8, 2011

692 likes, 138 dislikes

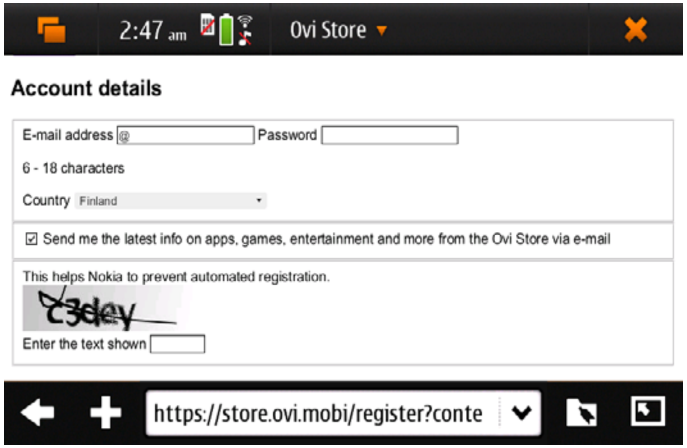
UPDATE 3: Someone has managed to repeat the same test with similar set

http://youtu.be/BwfYSR7HttA

CAPTCHA on mobile devices



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



Account details

E-mail address @ Password

6 - 18 characters

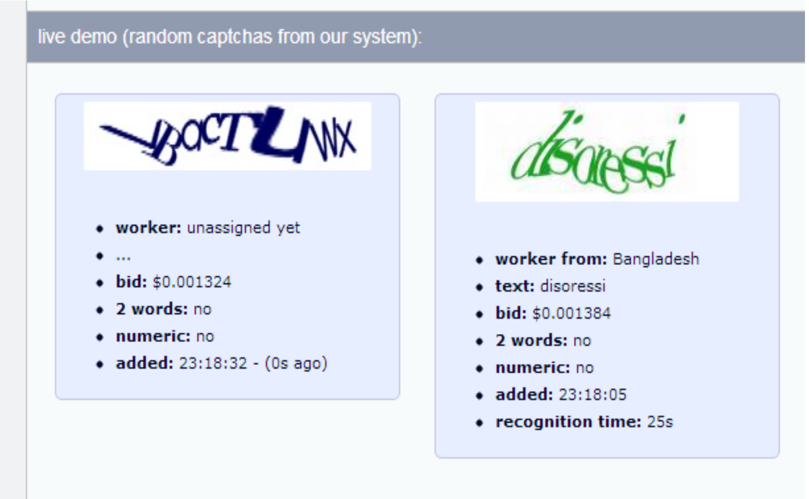
Country Finland

☑ Send me the latest info on apps, games, entertainment and more from the Ovi Store via e-mail

This helps Nokia to prevent automated registration.

Enter the text shown

https://store.ovi.mobi/register?conte



live demo (random captchas from our system):

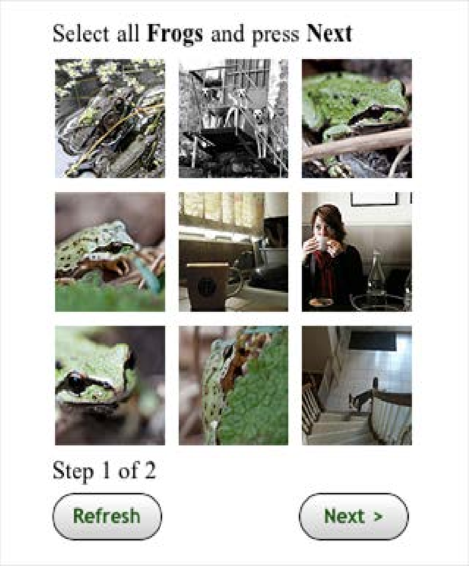
- **worker:** unassigned yet
- ...
- **bid:** $0.001324
- **2 words:** no
- **numeric:** no
- **added:** 23:18:32 - (0s ago)

- **worker from:** Bangladesh
- **text:** disoressi
- **bid:** $0.001384
- **2 words:** no
- **numeric:** no
- **added:** 23:18:05
- **recognition time:** 25s

http://antigate.com

Alternatives to standard CAPTCHA?

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



Select all **Frogs** and press **Next**

Step 1 of 2

Mobile CAPTCHA by Alex Smolen, Becky Hurwitz, Dhawal Mujumdar, UC Berkeley i213 Spring 2010

Usable security problems on mobile devices

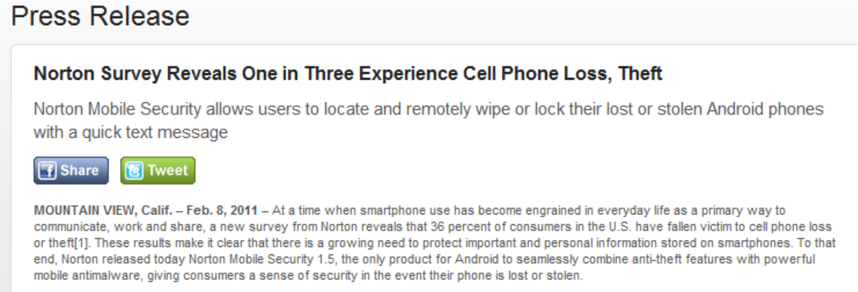
- Secure First Connect
- Permission granting to apps
- Local user authentication
- CAPTCHA
- …?

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

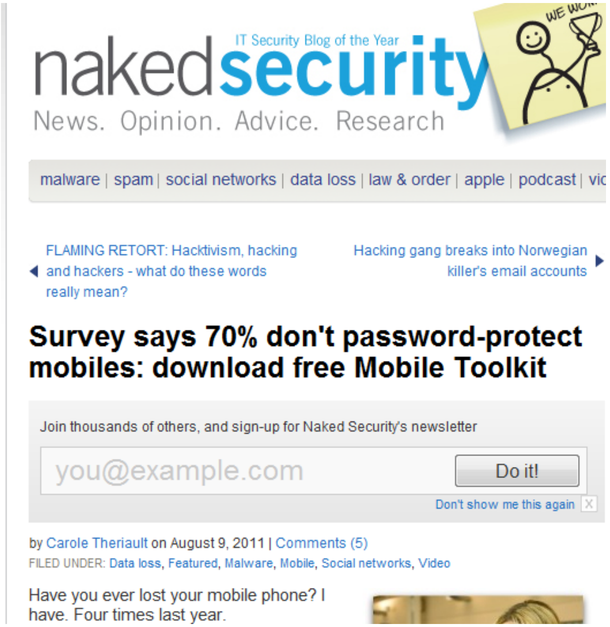

Press Release

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

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
WE WON
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

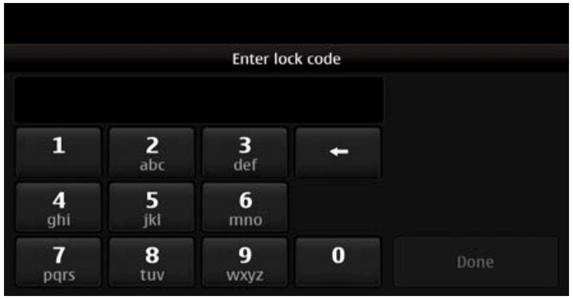
by Carole Theriault on August 9, 2011 | Comments (5)

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

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

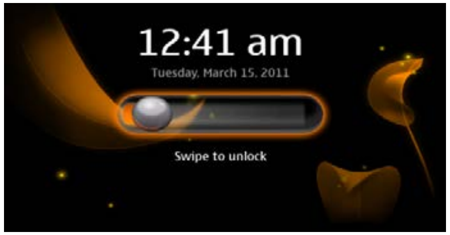
http://nakedsecurity.sophos.com/2011/08/09/free-sophos-mobile-security-toolkit/

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

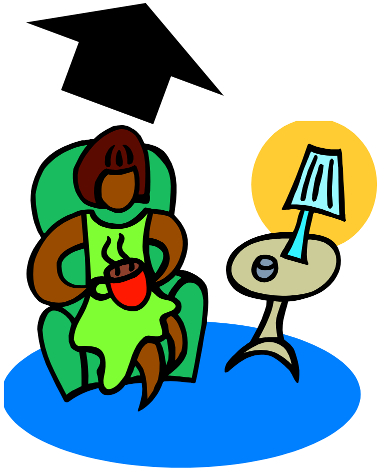

Enter lock code

Better Device Lock via Context Profiling

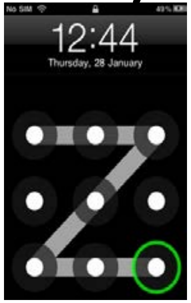
- Timeout and unlocking method adjusted based on estimated familiarity/safety of current context



Long timeout

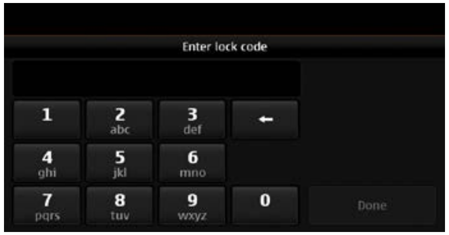


Home



Medium timeout

Work Cafeteria



Short timeout

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

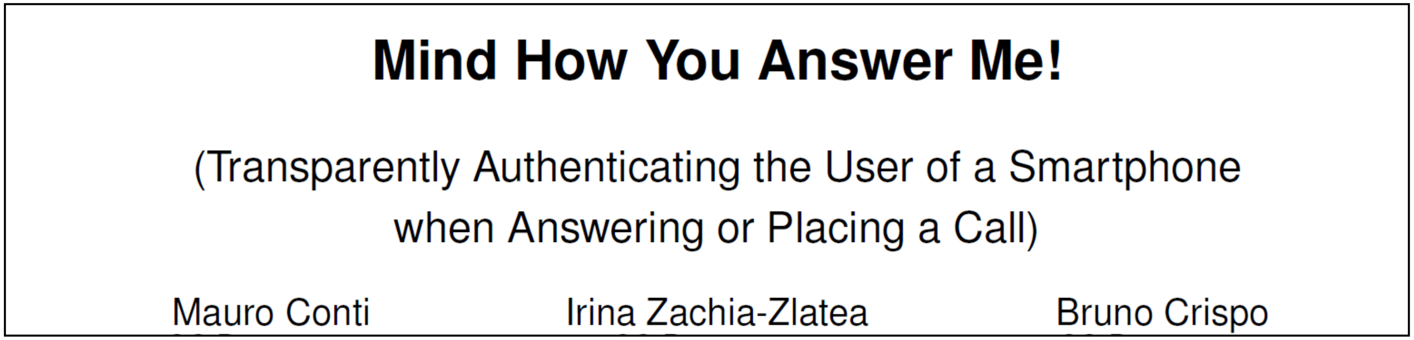
Identify places (“contexts”) meaningful to user

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

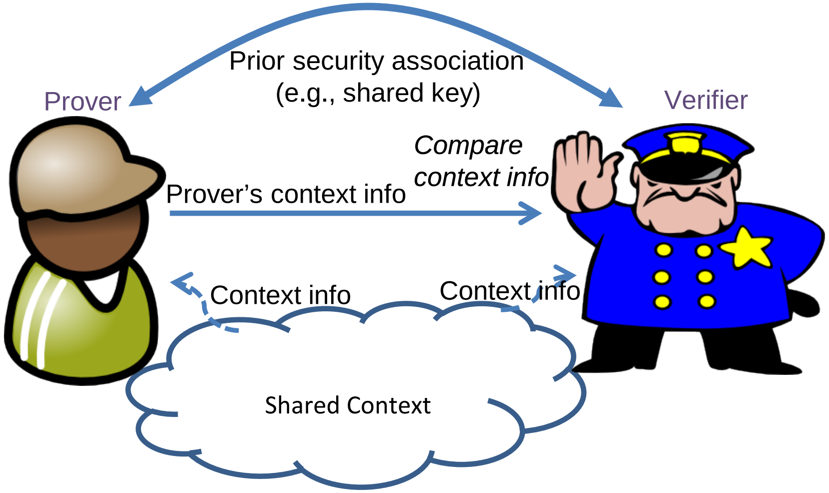
Other contextual security solutions

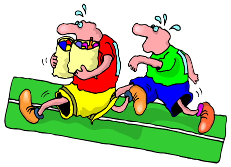
Comparing contexts for zero-interaction auth.

BlueProximity

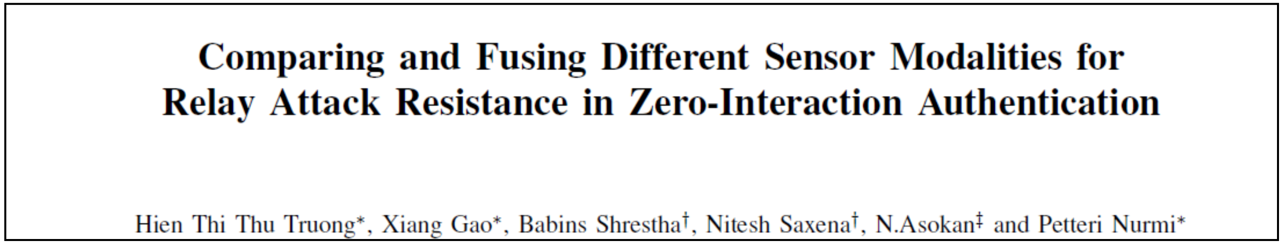
locks/unlocks your desktop tracking a bluetooth device

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





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



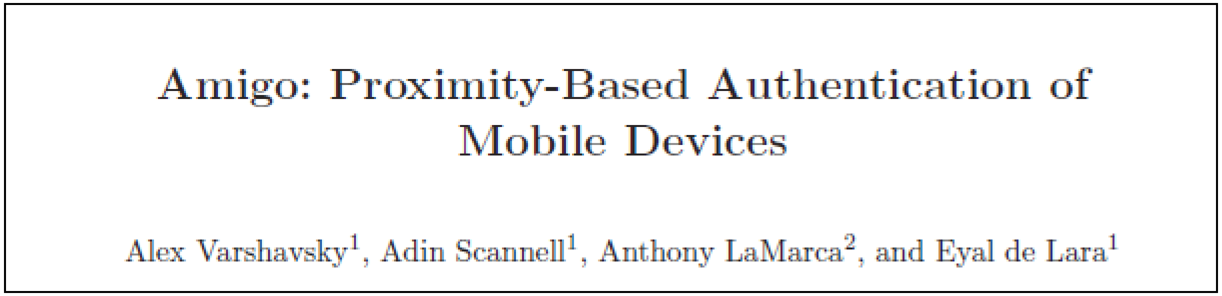
Comparing and Fusing Different Sensor Modalities for Relay Attack Resistance in Zero-Interaction Authentication

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

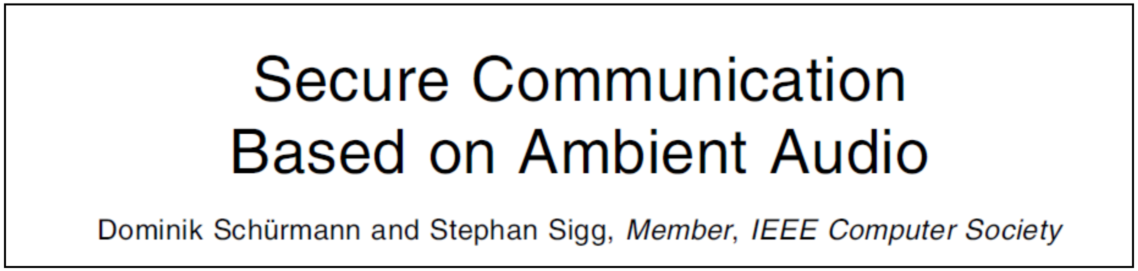
Key agreement based on shared context



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

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

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 mobile security is challenging but worthy
  - Lack thereof results in surprising costs
  - Needs changes under-the-hood (protocols, algorithms, ...)
- No satisfactory solutions yet for several problem instances
- Can contextual security help?

Slides of this talk: http://asokan.org/asokan/TCE2014

Contact info: http://asokan.org/asokan/