Secure Association of Ubiquitous Wireless Devices

Secure pairing of wireless devices

- **Pairing**: setup of a secure association and security context for subsequent communication. e.g.:
  - Bluetooth phone and a headset
  - Wireless printer and a PDA
  - Enrolling a phone or PC into a home WLAN
  - Emerging settings: Wireless USB, WiMedia

What devices?

- ❖Desktops
- ❖Laptops
- ❖PDAs
- ❖Phones
- ❖MP3 Players
- ❖Wireless Headsets
- ❖Cameras
- ❖Device (e.g., TV) Remotes
- ❖Access Points
- ❖FAX-s/Copiers/Printers
- ❖Sensors? RFIDs?
- ❖Pacemakers? Dialysis devices?

_____

p.s. How many devices?

p.p.s. What are their means of input and output?

Input? Output?

INPUT

- Keyboard, keypad, touch-screen
- Microphone (audio-in)
- Photo camera
- Video camera
- Vibration detector
- Infrared
- WLAN
- Cellular
- Bluetooth
- Ultrasound
- Accelerometer
- Scanner

etc., etc.,

OUTPUT

Screen

- LED
- Speaker (audio-out)
- Beeper
- Vibration ability
- Printing
- Infrared
- WLAN
- Cellular
- Bluetooth
- Ultrasound

etc., etc.

Problem Definition

How to set up a security association (authenticated secure communication channel) where:

- No prior context exists (no PKI, common TTPs, key servers, shared secrets, etc.)
- Ordinary non-expert users
- Cost-sensitive commodity devices

Diffie-Hellman Key Agreement

- How to share a secret where none existed…

Public values: large prime p, generator g
Alice has secret value a, Bob has secret b

1. A → B: $g^a \bmod p$
2. B → A: $g^b \bmod p$
3. Bob does: $(g^a \bmod p)^b \bmod p = g^{ab} \bmod p$
4. Alice does: $(g^b \bmod p)^a \bmod p = g^{ab} \bmod p$

- Eve <u>cannot</u> compute $\mathbf{g^{ab} \bmod p}$

So, are we done yet?

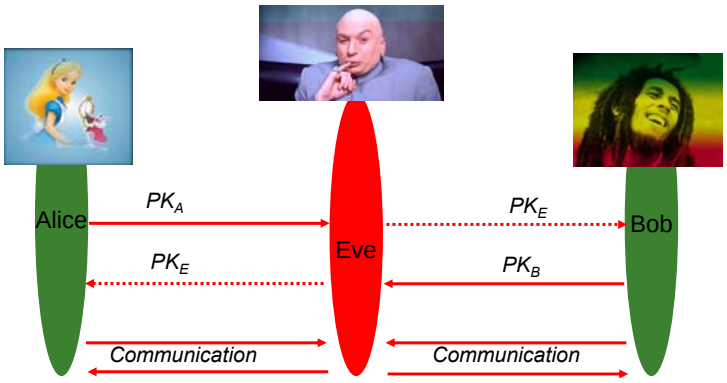
Problem: Man-in-the-Middle (MitM) Attacks

Mallory (M) can impersonate Alice to Bob, and Bob to Alice!

1. A → B/M: $g^a \bmod p$
2. M → A: $g^m \bmod p$
3. M/A → B: $g^m \bmod p$
4. B → A/M: $g^b \bmod p$
5. Bob does: $(g^m \bmod p)^b \bmod p = \mathbf{g^{bm} \bmod p}$
6. Alice does: $(g^m \bmod p)^a \bmod p = \mathbf{g^{am} \bmod p}$

Why? No authentication…

Man-in-the-Middle (MitM) Attacks



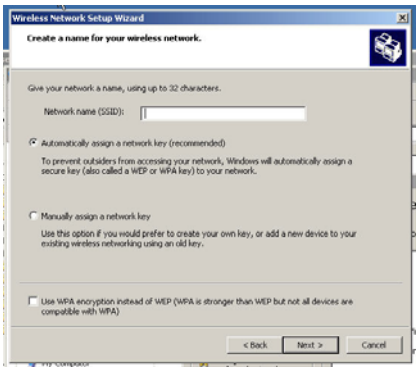
Eve controls the communication!

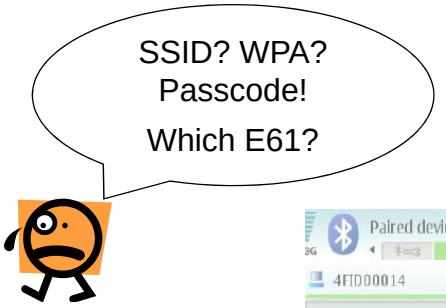
How Serious are MitM Attacks?

- Wireless communication is "*invisible*" or human-imperceptible
  - People can't tell which devices are "talking"
  - A rogue device might not be "visible" or identifiable as such
- A neighbor can easily execute an MitM attack
  - If neighbor has a faster computer, it can easily respond faster than the legitimate device(s)
  - Meanwhile, legitimate device(s) may also be "silenced" by DoS
- **Easy to mount with high success rate!**

Solution?

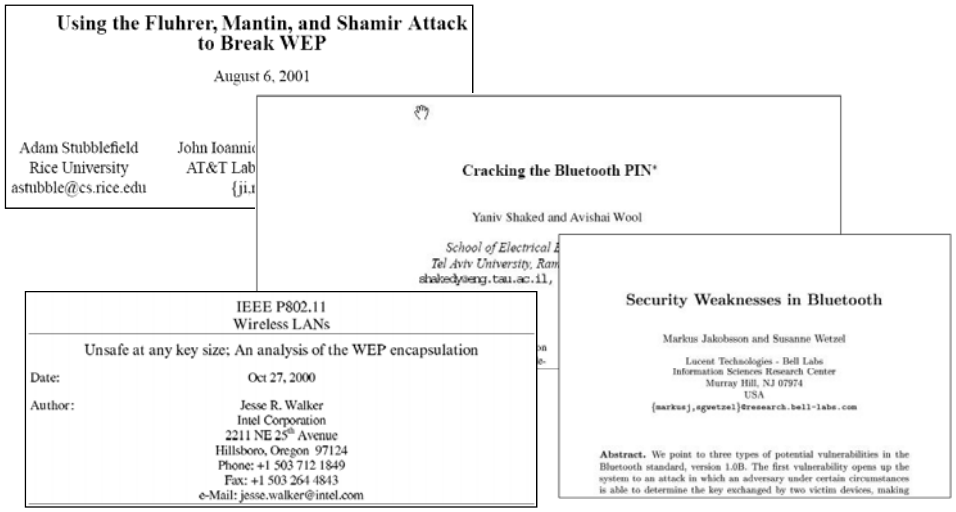
Current mechanisms are not intuitive





... not for all devices!

… and not very secure



Using the Fluhrer, Mantin, and Shamir Attack to Break WEP

August 6, 2001

Adam Stubblefield
Rice University
astubble@cs.rice.edu

John Ioanni
AT&T Lab
{ji.r

Cracking the Bluetooth PIN*

Yaniv Shaked and Avishai Wool

School of Electrical E
Tel Aviv University, Ram
shakedyweng.tau.ac.il,

IEEE P802.11
Wireless LANs

Unsafe at any key size; An analysis of the WEP encapsulation

Date: Oct 27, 2000

Author: Jesse R. Walker
Intel Corporation
2211 NE 25th Avenue
Hillsboro, Oregon 97124
Phone: +1 503 712 1849
Fax: +1 503 264 4843
e-Mail: jesse.walker@intel.com

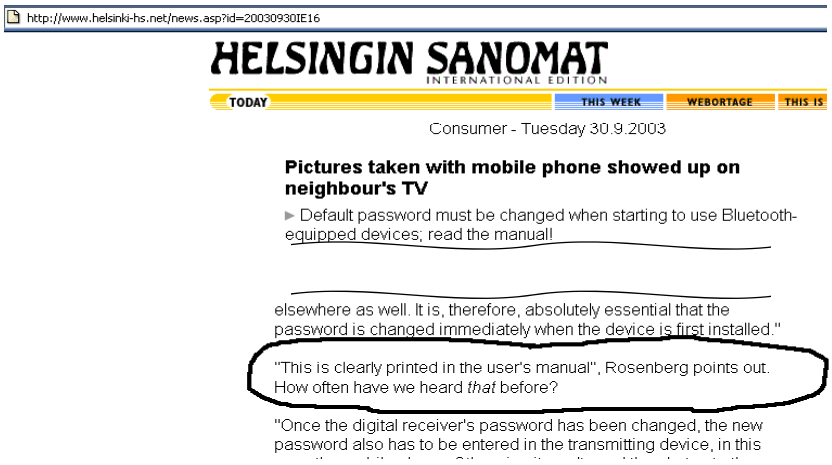
Security Weaknesses in Bluetooth

Markus Jakobsson and Susanne Wetzel

Lucent Technologies - Bell Labs
Information Sciences Research Center
Murray Hill, NJ 07974
USA
{markusj,agwetzel}@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

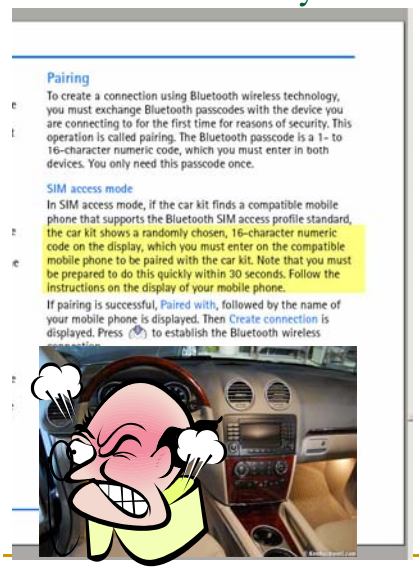
▸ Default password must be changed when starting to use Bluetooth-equipped devices; read the manual!

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 measures damage usability



- Bluetooth pairing was designed with moderate security in mind
- Car kits allow a car phone to retrieve and use session keys from a simcard
- Car kit requires higher level of security
  - users have to enter 16-character passcodes

More secure = Harder to use?

Goal: Secure, intuitive, inexpensive methods for device pairing

- Two (initial) problems to solve
  - Discovery: finding the other device and likely to establish an insecure channel.
  - Authenticated key agreement: setting up cryptographic keys for subsequent communication
- Assumption: Peer devices are physically identifiable
- Idea:
  1. Use a human-perceivable (out-of-band) channel to transport authenticated information (e.g. checksum of the public keys, or public key itself)

Some examples of what has been proposed thus far…

Resurrecting Duckling

F. Stajano and R. Anderson, IWSP '99

- **Problem:** how to set up keys in a ubiquitous computing environment?
  - Devices use wireless communication
  - Setup keys between household devices and a PDA
- **Solution?**

The Resurrecting Duckling

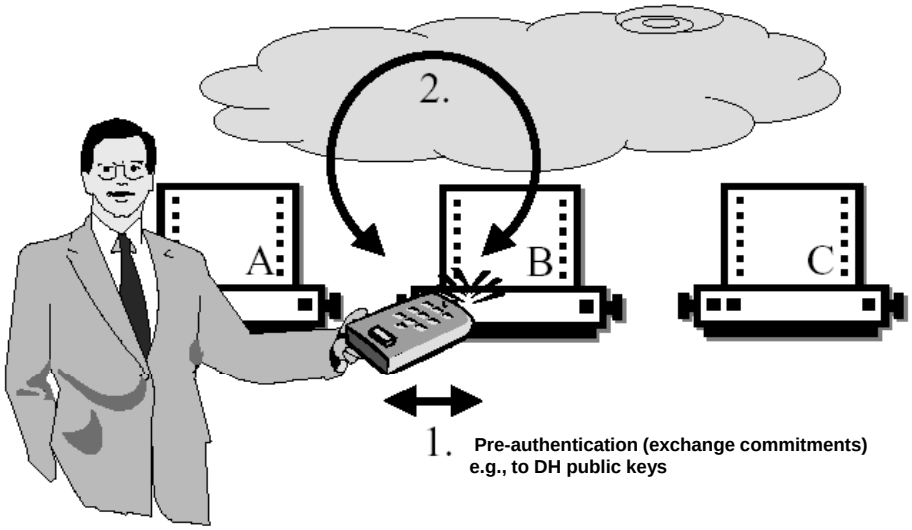
- **Solution:** set up keys using **trusted communication channel**
  - No cryptographic keys to setup this channel
  - Physical contact establishes a secure channel
  - E.g., a simple wire
- Caveats:
  - homogeneous physical interfaces
  - awkward cables

"Talking to Strangers"

Balfanz, et al. NDSS '02

- Addresses practical shortcomings of Duckling
  - Devices have no interfaces for physical contact
  - Cables are cumbersome
- Propose Infra-red as a "**Location-Limited Side Channel"**
  - Assumed to be immune to MitM attack
  - Many of today's (yesterday's) devices equipped with IR

Talking to Strangers



Talking to Strangers

- Pros
  - Works(-ed) on many commodity devices
- Cons
  - Most users do not know where their IR port is
  - Most devices require IR to be explicitly turned on
  - IR is invisible, attacker may still be able to mount MitM attack
    - E.g., two remotes, one TV

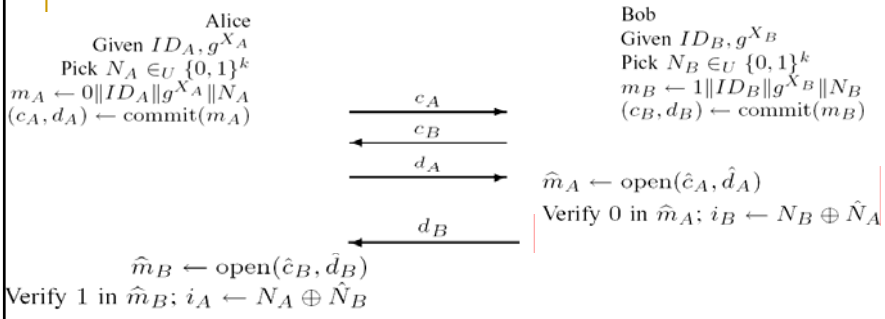
Key Agreement in P2P Wireless Networks

M. Cagalj, et al., Proc. of IEEE, Special Issue on Security and Cryptography, 2006

- Avoids use of side-channels
- Uses Diffie-Hellman to establish keys
- Three techniques to combat MitM
  - Visual comparison of short strings
  - Distance bounding
  - Integrity codes
- All 3 authenticate public DH parameters:

g^A and g^B

DH using Short String Comparison (DH-SC)



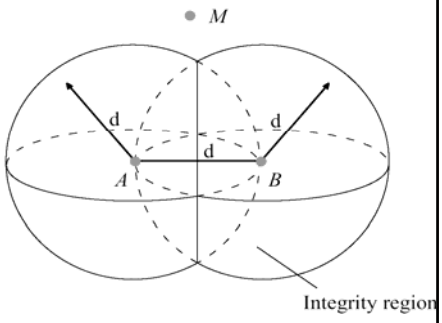
Alice and Bob "visually" compare i_A and i_B

If $i_A = i_B$, Alice and Bob output "Accept" $\hat{m}_B$ and $\hat{m}_A$, respectively.

Hold on! We will go over a similar protocol in detail.

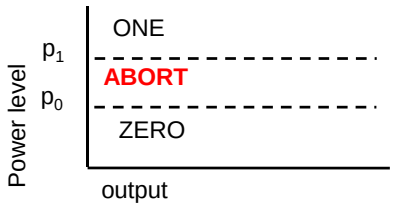
DH using Distance Bounding (DH-DB)

- Using precise timing by the radio interface, one can limit maximum possible distance between devices *A* and *B*
- Results in an **integrity region** which provides proximity verification
- If users can visually verify there are no other users / devices within the integrity region, then $i_A = i_B$
- How does this work?



DH using Integrity Codes (DH-IC)

- The sending radio transmits at only 2 power levels
  - Power level 0 indicates a logical 0
  - Power level p indicates a logical 1
- The receiver applies 2 thresholds (p_0 and p_1)
  - Signals above p_1 are a logical 1
  - Signals below p_0 are a logical 0
  - Signals between p_0 and p_1 **abort** the protocol



DH using Integrity Codes (DH-IC)

•Transmit messages in code words with a fixed number of 1's

•Attacker can inject 1's, but cannot remove 1's

•The receiver must be turned on and listening on the correct channel during the sender's transmission

•Example:

Messages:	00	01	10	11
Code words:	0001	0010	0100	1000

•In pairing: transmit the small verification string using integrity codes, so that attacker cannot change it!

Ad Hoc Group Device Pairing

- N. Asokan and P. Ginzboorg, "Key Agreement in Ad-hoc Networks," *Computer Communications*, vol. 23, no. 17, pp. 1627–1637, 2000.

- Problem: how to set up a session key between a group of people/devices their who meet and have no prior context
- Shared password approach
- No PKI, no TTP
- Fresh password is chosen and manually shared among those present in the room (e.g., by writing on blackboard)
- Password used to derive a strong shared session key using either group DH or group-EKE (GDH)
- Requires each user to type in the password
  - Devices must have a keyboard/keypad

FYI: See paper on keyboard snooping (emanations) from IEEE S&P'04

BLUETOOTH?

Seeing-is-Believing (SiB)

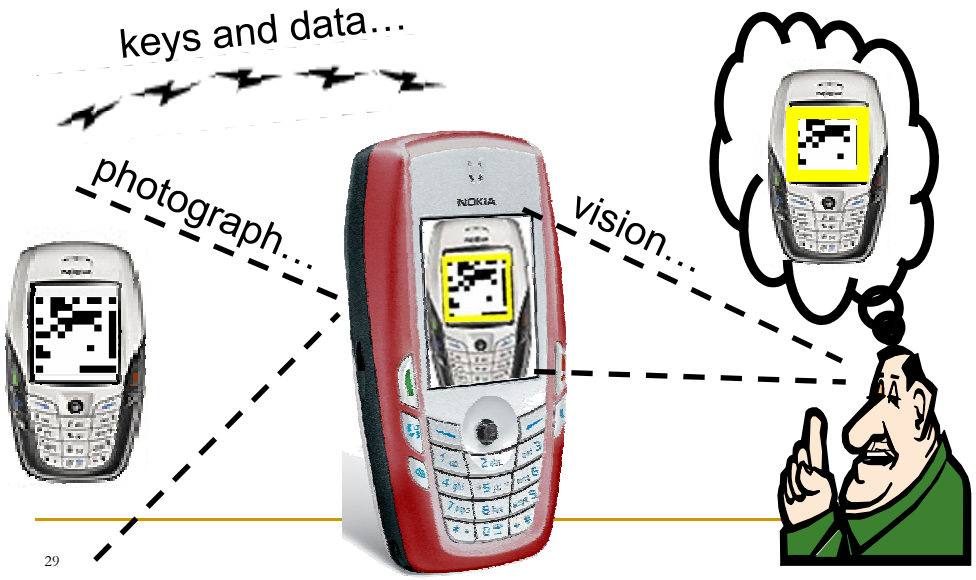
McCune, et al. IEEE Security &Privacy '05

- Difficult to achieve **demonstrative identification** of devices communicating wirelessly with no prior context
- Prior work proposes the use of a **location-limited side-channel** to authenticate devices
  - Infrared, ultrasound, physical contact
- Proposals to-date too cumbersome for non-expert users
  - None of them convince the user that they are really communicating with ***the target*** device

Seeing-Is-Believing

- Camera phones have sufficient resources to scan 2D barcodes
- Some have high-quality screens which can display freshly-generated barcodes
- Using them together yields a ***visual***, location-limited channel
- Visual channel ***can*** provide **demonstrative identification** of communicating parties to the user
- Enables strong human-assisted authentication

Basic SiB Protocol



Basic SiB Protocol

	A		B
1	$h_A \leftarrow Hash(PK_A)$		
2		h_A (visual)	
3		PK_A (other)	$h' \leftarrow Hash(PK_A)$
4			$if h' \neq h_A$ *then abort*

Rohs & Gfeller, 2003

SiB Caveats

- Not all devices have big enough displays to show two-dimensional bar codes
- Not all devices have good-enough cameras
- Sometimes devices cannot be placed sufficiently near
- There might not be enough light for pictures

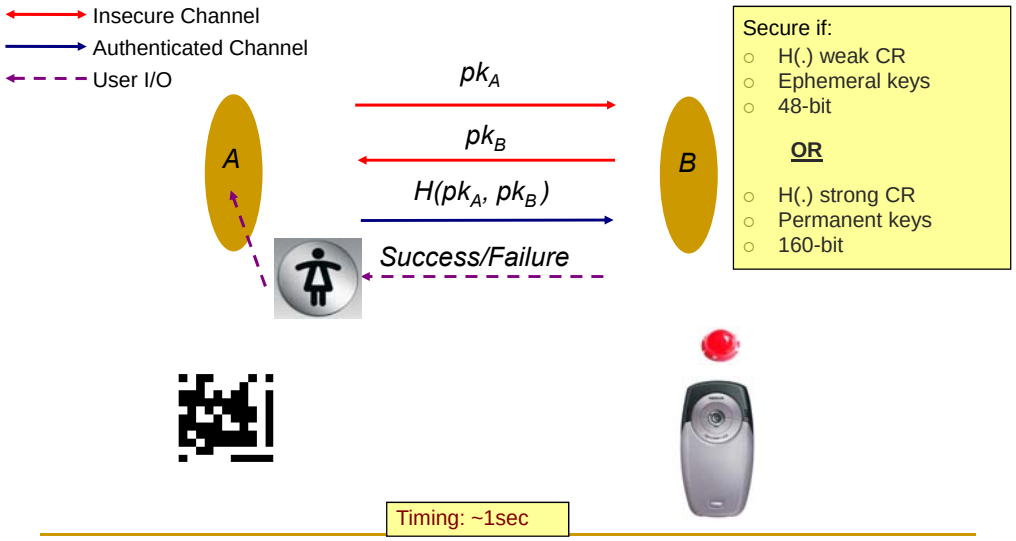
Blinking Lights:

Secure Device Pairing based on a Visual Channel
Saxena, et al., IEEE Security & Privacy 2006

Main Idea:

- One device blinks
- The other takes a video clip
- Video clip parsed to extract an authentication string

Mutual Authentication in a Single Step



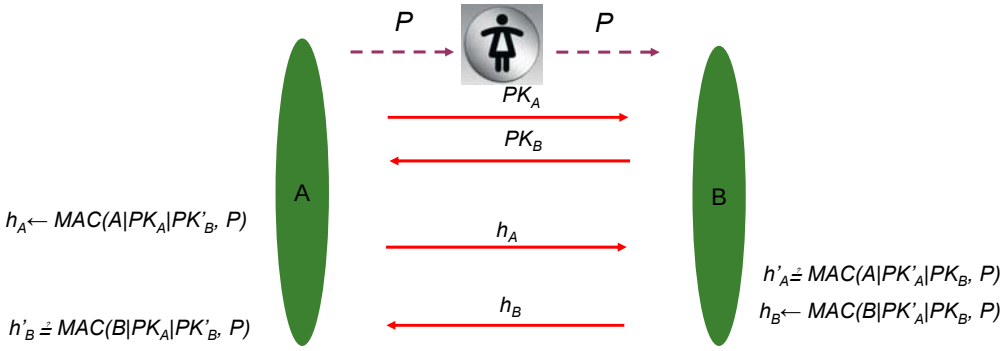
BEDA: Button-Enabled Device Association

- Can accommodate almost any pair of devices
  - very basic interface almost universally available: just a single button
- Better user experience and fewer usability-related security flaws
  - protocol suite (depending on specific hardware availability) to maximize usability
  - tested for usability

BEDA Protocol (Phase 1)

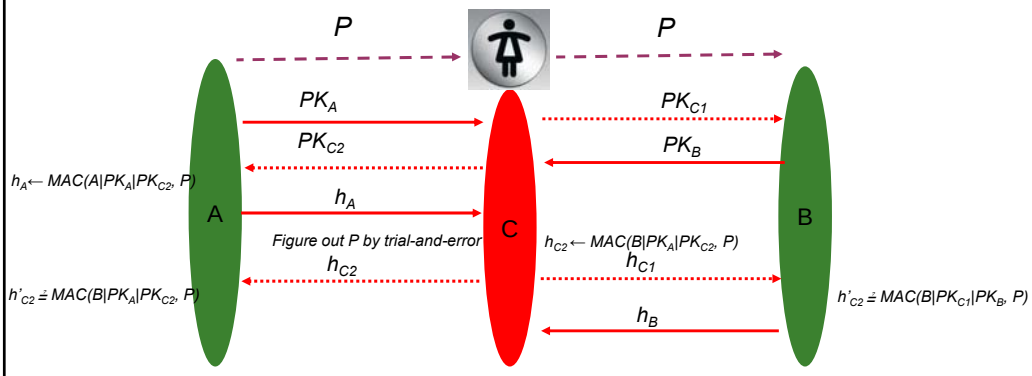
- Just using a button on both devices
  - Both devices acquire a secret from the user
    - simultaneously press-and-release buttons on both devices
    - use the elapsed times between button actions to calculate the same short secret on both devices
- If output interface is available on either device
  - One device chooses a random secret and user transfers it to other device
    - device with output signals user when to press a button
    - interval between button presses are used to reconstruct the secret

Authentication using a short passkey: a first attempt



- P is a short passkey (e.g., 4 digits)
- MAC() is a message authentication code: e.g., HMAC-SHA2
- But a man-in-the-middle attack can easily defeat this protocol!

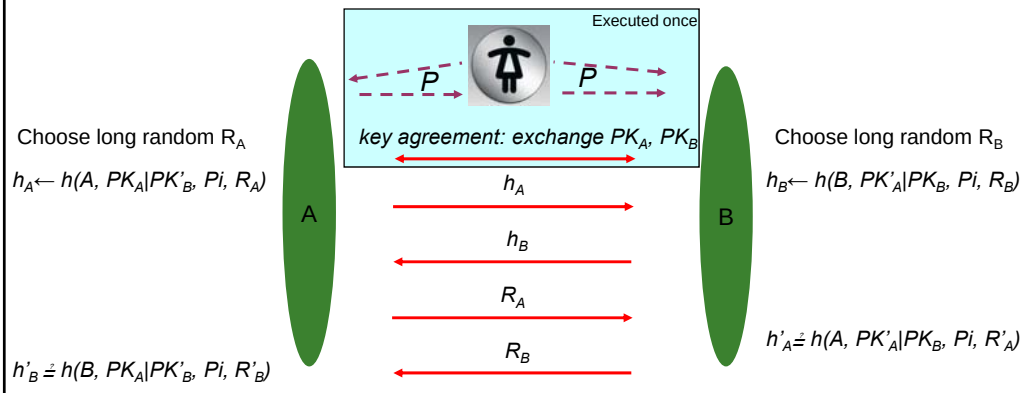
Man-in-the-middle attack



- Guess a value x for P; compute $h_x = MAC(A|PK_A|PK_{C2}, X)$; Check $h_A \stackrel{?}{=} h_x$
- If P is a n-digit PIN, attacker needs at most 10^n guesses (one MAC each)
- A regular PC can compute 100,000 MACs in 1 second

BEDA Protocol (Phase 2)

Authentication using secret short passkeys



- One-time passkey *P* is split into *i* parts ($i > 1$): 4-round exchange repeated *i* times
- *h()* is a hiding commitment; in practice SHA-256
- Up to $2^{-(k-1)}$ (unconditional) security against man-in-the-middle (k is the length of *P*)

Implementation

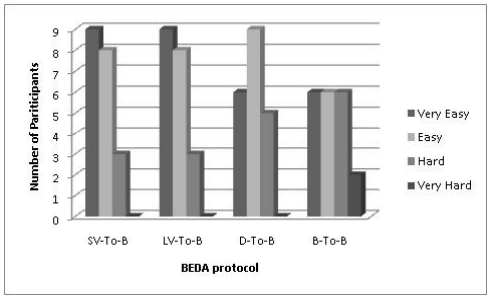
- Using 300ms as the unit of measurable time, obtained a 3-bit random value between each action (button press)
- After observing 7 actions, 21-bit secret is constructed
- As means of output, implemented simple display (blinks) and vibration versions

Usability Results

- 20 subjects testes
- 77%: BEDA is easier than current Wi-Fi pairing
- 36%: BEDA is easier than current Bluetooth pairing
- Simple and fun to use

Method	Average completion time in seconds	Average number of retrials for success
B-To-B	53.2 (sd*=32.5)	2.45 (sd=1.43)
D-To-B	72.8 (sd=39.4)	1.45 (sd=0.89)
SV-To-B	60.1 (sd=18.3)	1.35 (sd=0.49)
LV-To-B	56.6 (sd=19.4)	1.20 (sd=1.41)

*sd= Estimated standard deviation

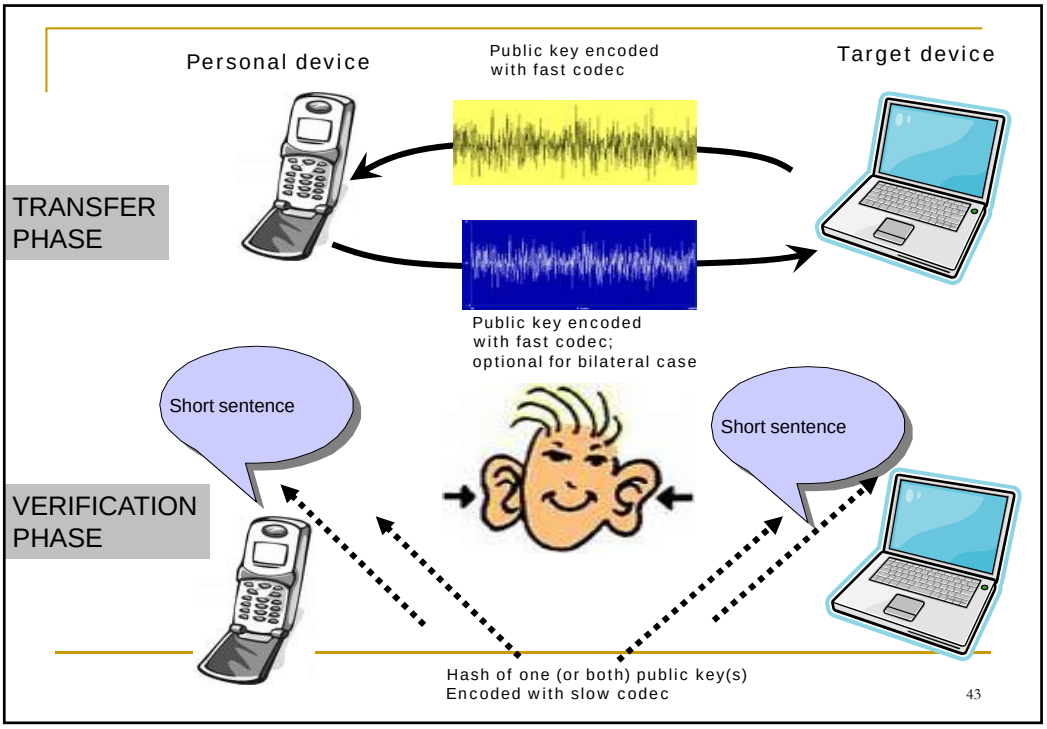


Limitations & Strengths

- Limitations
  - Assumes
    - An available insecure connection
    - At least a button for input
    - An output on one of the devices (to show outcome of the pairing)
  - Not well-suited for disabled and (maybe) elderly users
- Strengths
  - Works on devices with simplest form of user interface
  - Provides security for ordinary users
  - Generally positive user opinions

HAPADEP: Human Assisted Pure Audio Device Pairing

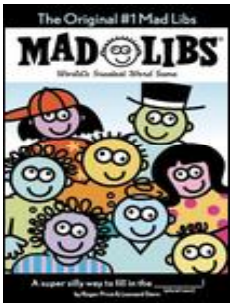
- Highlights
  - No assumption of established communication channel
    - no common interface at time of pairing
    - no configuration and discovery problems
  - Uses audio as the sole communication channel
    - Audio is
      - perceptible (Authentication, DoS, attacker identification)
      - broadcast in nature (no configuration, discovery etc.)
  - Audio input/output needed on both devices for bilateral key exchange.



How to convert verification data to sentences?

- Generate a non-sensical, English-like sentence (e.g., Mad-Libs) from hash
  - Use each 10-bit section of the digest as an index into a catalogue of words
  - One catalogue for each part of speech, e.g., verb, noun etc.
- Produces sentences such as:
  - "John flexibly drinks a building"

  or
  - "Alice always vacuums an elephant"



How to transmit data over audio?

- Should be faster than speech (sentences) and more pleasant than modem noise
- Not as fast as modems or as pleasant as Mozart
- But, pleasant enough to be not disturbing and fast enough to transmit a public key in 2-5 seconds.

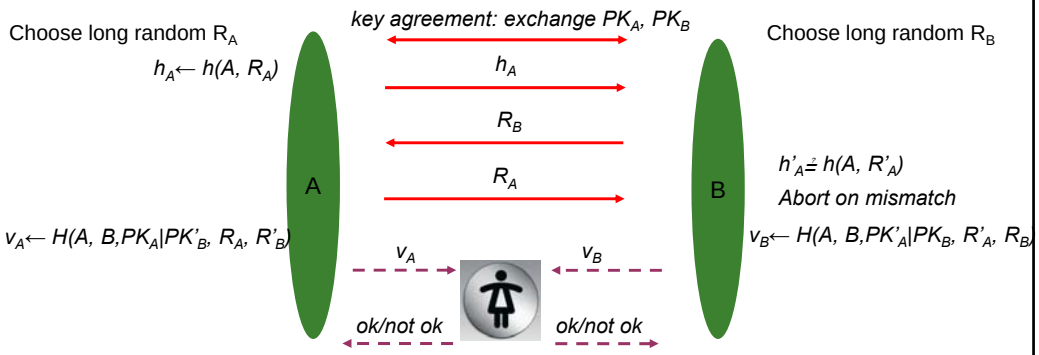
Limitations

- Needs audio input/output on both devices.
- Too much noise or quiet environments may be problematic.
- Not for people with certain disabilities (e.g. deafness)

Overcoming hardware limitations

- What if devices do not have both speaker and a microphone.
  - For transfer phase
    - A common interface is needed and PK's can be transferred over it
      - If one device has microphone and the other has speaker, configuration string can still be sent over audio to avoid discovery and setup
  - For verification phase
    - Sentences can be displayed
    - So, devices having either a speaker or a display would do just fine!
- "Loud and Clear" (ICDCS'06)

Loud and Clear
(Authentication using non-secret short check codes)

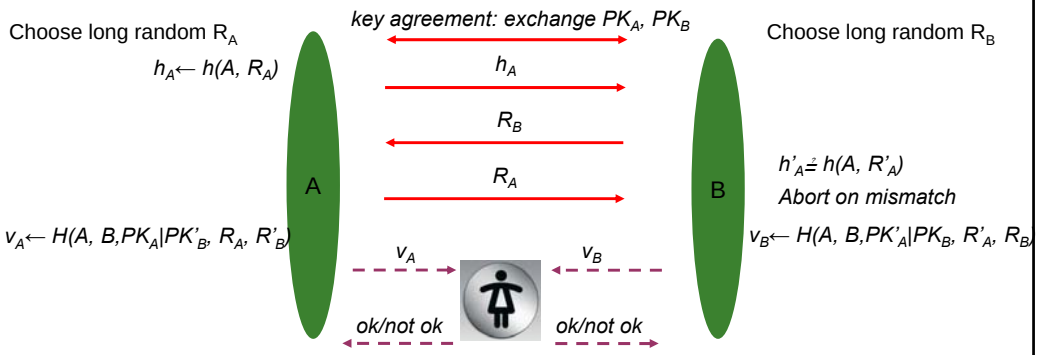


User approves acceptance if v_A and v_B match

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

H*()* is a mixing function; in practice SHA-256 output truncated to 4 digits

Loud and Clear
(Authentication using non-secret short check codes)



User approves acceptance if v_A and v_B match

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

H*()* is a mixing function; in practice SHA-256 output truncated to 4 digits

Strengths

- L&C and HAPADEP together
  - Can accommodate wide variety of devices.
  - Provide very high level security against MITM attack. (non-secret based protocols)
  - Are user friendly (they “speak” our language)
  - Solve both the discovery and key agreement problems
  - HAPADEP: more resistant to DoS attacks.

Recap: Recent proposals

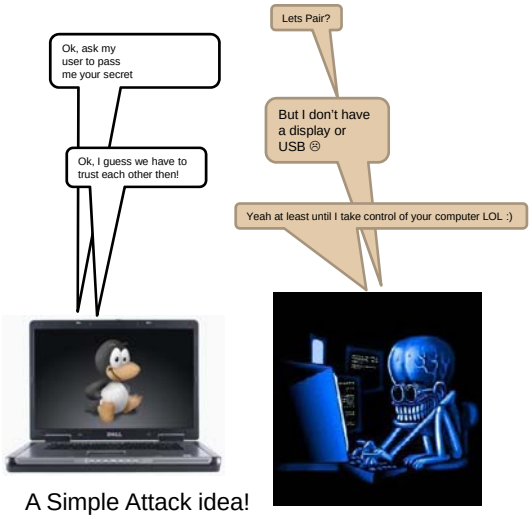
- Using different out-of-band channels for better security and usability
- Solutions suggested the use of
  - Cables
    - Resurrecting Duckling, [Stanajo, et al. IWSP'99]
  - IrDA, Camera and barcodes/LEDs
    - Talking to Strangers, [Balfanz, et al. NDSS'02]
    - Seeing-is-believing, [McCune, et al. S&P'05]
    - SIB revisited, [Saxena, et al. S&P'06]
  - Exotic hardware
    - Accelerometers → "Shake well before use", [Mayrhofer, et al. Pervasive'07]
    - Ultrasound, laser transceivers and many others....
- Most are simply impractical!
  - Even the viable ones require IrDA, camera, etc.

Current Standardization Activities

- WiFi
  - WiFi Protected Setup, Jan 2007
  - Windows Connect Now
- Bluetooth
  - Secure Simple Pairing, Feb 2007
- Wireless USB Association Models Supplement, 2007
- What is common;
  - If available use USB stick/cable or NFC transceivers.
  - Or, if devices have display and a keypad, make the user transfer or compare 4 to 8 digits.
  - Otherwise, insecure key exchange.
  - Numerous problems already!
    - Kuo, et al. [USEC'07], Asokan, et al. [ESAS'07], Uzun, et al. [USEC'07]

Current Standardization Activities

- Algorithm
  DO{
  try to accommodate variety of devices
  }WHILE
  not having proper methods or design
  RETURN
  failure



A Simple Attack idea!

Usability in secure pairing

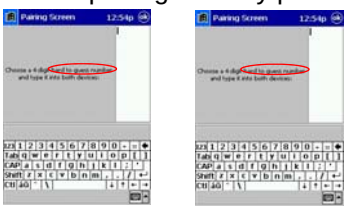
- "Usability Analysis of Secure Pairing methods"
  Uzun, et al. [USEC'07]
  - Highly-educated users, 2 groups, 40 people in each
  - Objectives:
    - study pairing proposals in emerging standards
    - identify possible user-interaction methods
    - evaluate/compare methods
    - find implementation strategies that maximize their usability and security

Tested user interaction methods

- Tested 5 different user interaction methods
  - 3 using **short non-secret** check code based protocols
  - 2 Using a **short secret** passkey based protocols
- With different key lengths and GUI designs
- Interviewed users about their perception and experience
- Will briefly mention only 4 specific test cases.

Choose-and-Enter

- User chooses a number as the passkey and types it into the both devices. (Like in current Bluetooth pairing in many phones)

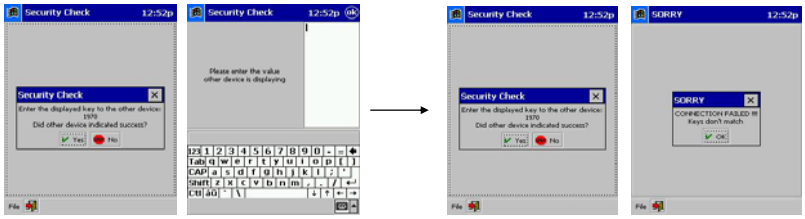


- 42.5% of the participants used very predictable repeating or in-sequence numbers. All admitted reading the warning!
- Human = very bad RNG (known for decades!!!)
- No matter how secure the underlying cryptographic protocol is, such interaction design is insecure!

Short secret passkey

Copy-and-Confirm

- One device shows a number and asks user to type it into the second device. User confirms on the first device only after seeing success on the second.



- 10% didn't wait for success indication before confirming on the first device.
  - Sticking to interacting with one device
  - Press "yes" to continue!
- This interaction design is also insecure

Short non-secret checksum

Compare-and-Confirm

- Each device shows a number and asks user to compare shown values.
  - Straight-forward implementation of YES/NO question.
    - **20%** pressed "yes" on non-matching values (they didn't read instructions!)
    - INSECURE!
- Different question, uncommon answers (same/different).
  - No fatal errors in tests, but 2.5% cancelled the connection on matching values.

- Good for now, but learning effect? May get worse in time!

Short non-secret checksum

Copy

- One device shows a number as a passkey and user types it into the second device. Devices accept or abort automatically.

- Results
  - Users cannot make a mistake that would result in an attack. Devices accept/reject automatically. 97% transferred the number correctly.
  - Due to the nature of the protocol, transferred number has to be kept secret here.

Short secret passkey

Recapping good methods

- Compare and Confirm
  - Good only under certain GUI implementations
    - New Bluetooth Standard not guided the GUI
      - All implementations we know use a trivial yes/no question!
- Copy
  - Good usability but the passkey value should be kept secret and used only once.
    - Wi-Fi Standards not enforced it
      - Routers with a permanent passkey sticker appeared!
    - Now it is enforced
      - Many products choose not to support it

Problem recap: where are we today?

- Security problems (e.g. Attacks on Bluetooth, WEP)
  - New secure protocols address it (secure only if implemented correctly)
  - Standards implementing these still have flaws!!!
- Usability problems (e.g. Human =? RNG)
  - New standards and proposals provide better usability
  - Still missing
    - Standardized/guided UI

- Practicality
  - Still none of them can securely pair even the most common scenarios
    - Cell-phone and a Bluetooth Headset
    - Wireless Router and a PC

What is next?

- Design space for 2 device pairing is not fully explored yet. And, still no universal solution that does it all.
- Group pairing scenarios for 2+ devices.
- Pairing with interface-less devices e.g. RFID, some sensors

Conclusions

- Secure Device Pairing problem has 3 dimensions: security, usability and practicality.
- Use secure cryptographic protocols that stay secure even when Homer Simpson uses it
- If the user is involved, tasks should be intuitive and not burdensome
  - Taking pictures/videos is one way
  - Listening is another
  - Reading is yet another
  - And there other others, such as shaking…
- Exotic hardware assumptions (laser transceivers, accelerometers, etc.) don't let us solve the problem in real-life, at least not yet…