black hat
EUROPE 2019

DECEMBER 2-5, 2019
EXCEL LONDON, UK

BlueMaster: Bypassing and Fixing Bluetooth-based Proximity Authentication

Youngman Jung and Junbum Shin
Samsung Research

Yeongjin Jang
Oregon State University

Disclaimer

Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Samsung Electronics and Oregon State University.

AGENDA

- Bluetooth-based Proximity Authentication
- Preliminaries
- Security Analysis - Proposed Approach
- New Vulnerabilities
- Mitigations
- Conclusion

AGENDA

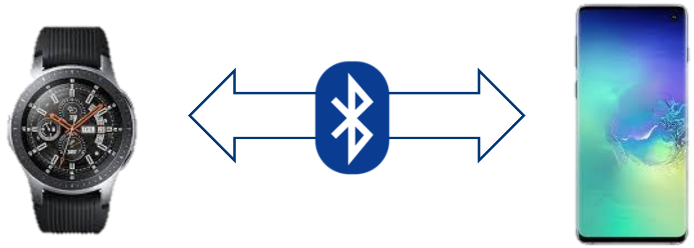
- ***Bluetooth-based Proximity Authentication***
- Preliminaries
- Security Analysis – Proposed Approach
- New Vulnerabilities
- Mitigations
- Conclusion

Bluetooth-based Proximity Authentication

- Types of Authenticators
  - Something you know (e.g. Password, PIN, Pattern)
  - Something you are (e.g. fingerprint, face, iris)
  - Something you have (e.g. key, smart card, usb token)

- Bluetooth-based Proximity Authentication
  - Proximity of your device

e.g. Android Smart Lock – Trusted Device



unlock the phone
without user authentication
if a registered Bluetooth device is connected

Bluetooth-based Proximity Authentication

- Types of Authenticators
  - Something you know (e.g. Password, PIN, Pattern)
  - Something you are (e.g. fingerprint, face, iris)
  - **Something you have (e.g. key, smart card, usb token)**
- Bluetooth-based Proximity Authentication (Goal: to improve convenience and security)
  - Authentication: Having a **securely paired Bluetooth device** serves as a proof of **something you have**
  - + Proximity Check: Measuring the signal strength (RSSI) of the established **Bluetooth connection** (Works within distance <100m)
  - Use cases:
    - Unlock a device: Android (Smart Lock)
    - Lock a device: Windows (Dynamic Lock)

Android Smart Lock
Ask no passcode if trusted device exists

- What is Android Smart Lock?
  - A convenient main-screen unlock feature (**truste**
  - Skip user authentication (passcode/fingerprint/f any of pre-registered, trusted device is connecte

Goal: To replace user authentication (e.g., passcode/fingerprint/face-unlock)

- When is it introduced?
  - 2014 by Google, starting from Android 5.0 Lollipop
- How to use this?
  - Pair and register a device as Trusted Device

Connection Established

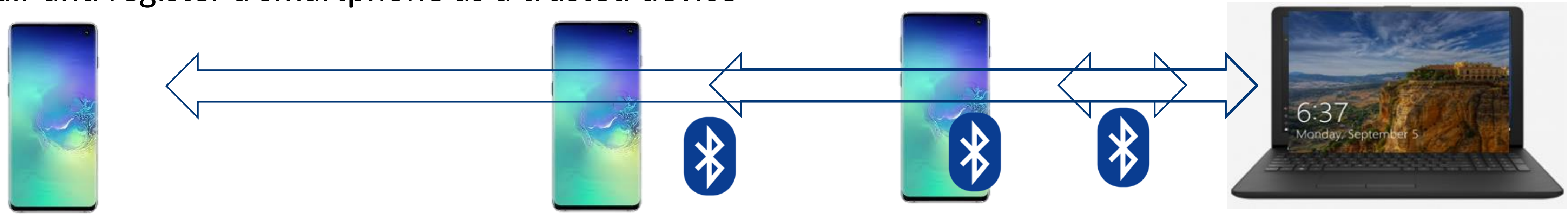
DO NOT ASK PASSCODE

Windows Dynamic Lock
Lock your PC if you are away from it

- What is Windows Dynamic Lock?
  - Automatically locks your PC when you goes out
  - Actually, Windows 10 measures distance betwee
  - By measuring the signal strength (RSSI) of the Bl
- When is it introduced?
  - 2017 by Microsoft (Windows 10, 1703)
- How users are using this?
  - Pair and register a smartphone as a trusted device

Goal: To provide an additional security Layer to the Lock screen

If your smartphone moves away from your PC e.g., RSSI < -10db, then it will lock the PC dynamically



Bluetooth-based Proximity Authentication

For **Secure** Bluetooth-based Proximity Authentication,

We need answers to the following questions:

1. **How can we utilize Bluetooth for Authentication?**
2. **How can we utilize Bluetooth for Proximity Checking?**

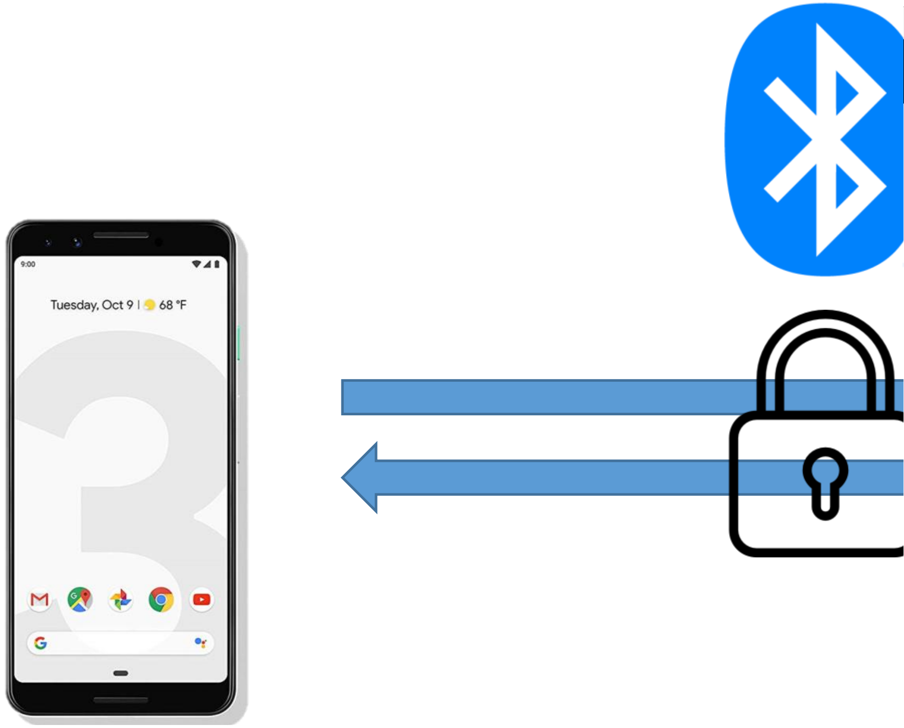
Bluetooth-based Proximity Authentication

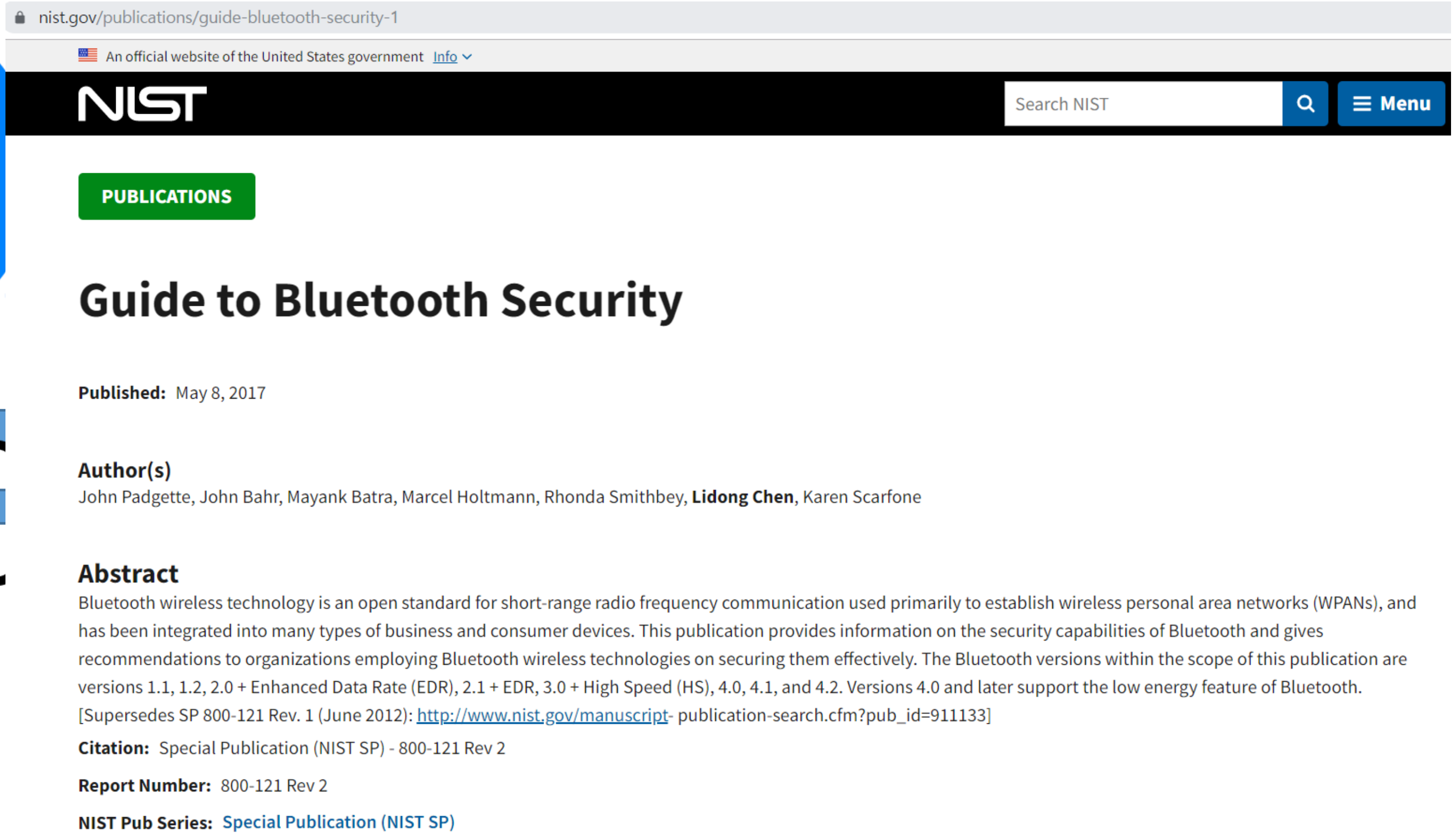
black hat EUROPE 2019

Bluetooth Security

→ *Secure for Communication?*

YES!




nist.gov/publications/guide-bluetooth-security-1
An official website of the United States government Info
NIST
Search NIST
Menu
PUBLICATIONS

Guide to Bluetooth Security

Published: May 8, 2017

Author(s)
John Padgette, John Bahr, Mayank Batra, Marcel Holtmann, Rhonda Smithbey, **Lidong Chen**, Karen Scarfone

Abstract
Bluetooth wireless technology is an open standard for short-range radio frequency communication used primarily to establish wireless personal area networks (WPANs), and has been integrated into many types of business and consumer devices. This publication provides information on the security capabilities of Bluetooth and gives recommendations to organizations employing Bluetooth wireless technologies on securing them effectively. The Bluetooth versions within the scope of this publication are versions 1.1, 1.2, 2.0 + Enhanced Data Rate (EDR), 2.1 + EDR, 3.0 + High Speed (HS), 4.0, 4.1, and 4.2. Versions 4.0 and later support the low energy feature of Bluetooth. [Supersedes SP 800-121 Rev. 1 (June 2012): http://www.nist.gov/manuscript- publication-search.cfm?pub_id=911133]

Citation: Special Publication (NIST SP) - 800-121 Rev 2

Report Number: 800-121 Rev 2

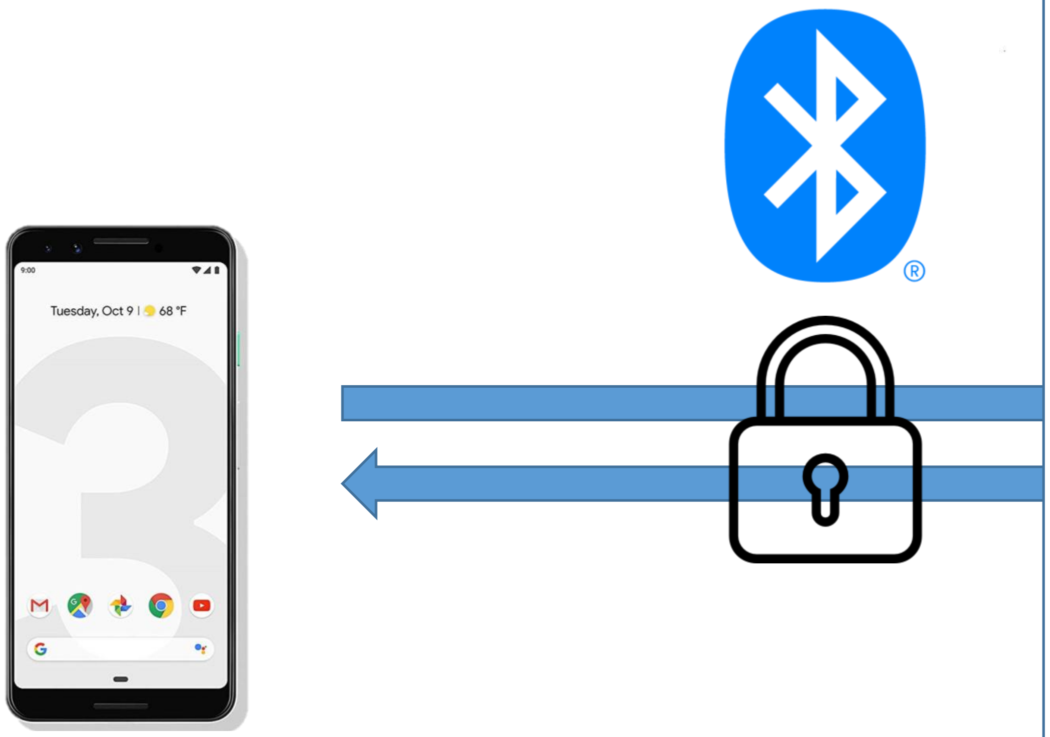
NIST Pub Series: Special Publication (NIST SP)

Bluetooth-based Proximity Authentication

Bluetooth Security

→ *Secure for Proximity Authentication?*

NO!



Martin Hurfurt (`2015)

- Shows insecurity for Smart Lock using Trusted Device because it uses a service not protected by Bluetooth Security

Beccaro and Collula (`2015)

- Same problems occur in 3rd party apps

Fixed by Google (`2015. 4)

- Since Android 5.1 (Changelog (Line 8883))

Secure?

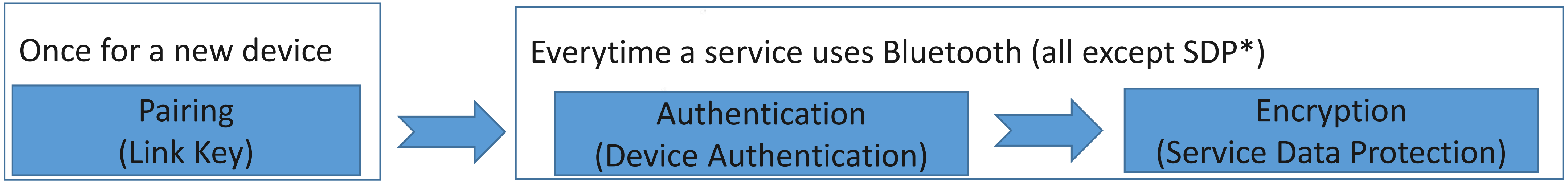
AGENDA

- Bluetooth-based Proximity Authentication
- ***Preliminaries***
  - Bluetooth Security 101
  - Proximity Authentication vs. Bluetooth Security
- Security Analysis – Our Approach
- New Vulnerabilities
- Mitigations
- Conclusion

Bluetooth Security 101 – Security Components

*Security Components (Security Mode 4) of Bluetooth BR/EDR**

* Bluetooth BR/EDR: for handling a lot of data, Bluetooth LE: for less power consumption



Once for a new device

Pairing (Link Key)

Everytime a service uses Bluetooth (all except SDP*)

Authentication (Device Authentication)

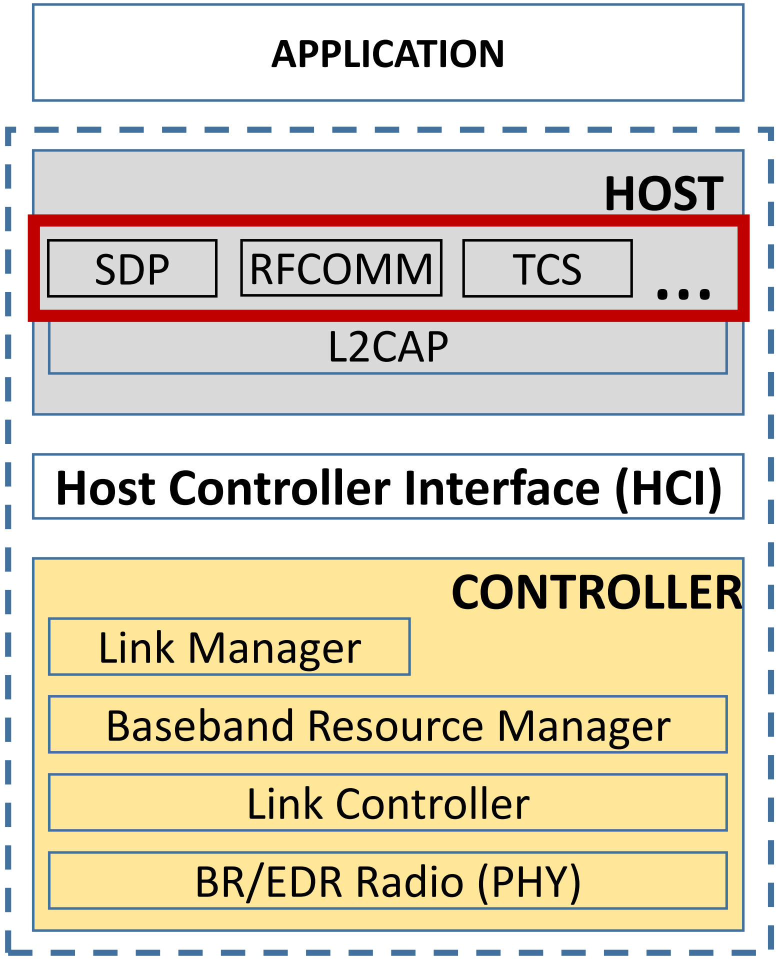
Encryption (Service Data Protection)

* SDP: Service Discovery Protocol ← Not protected by Bluetooth Security

	Secure?	Note (Secure when it is properly used)
Pairing and Link Key Generation	**Yes**	Secure Simple Pairing – **Secure against MITM attack** (Elliptic Curve Diffie-Hellman public key cryptography, P-256)
Authentication	**Yes**	Secure Authentication (**Mutual Authentication** using a link key)
Encryption	**Yes**	**AES** CCM Encryption
Service Security Levels (Service Level 4)	**Yes** except SDP	Service Level 4 - Requires **MITM protection** and encryption using 128-bit equivalent strength for link and encryption keys

Bluetooth Security 101 – Security Components

Bluetooth Architecture - Bluetooth Basic Rate/ Enhanced Data Rate



- **SDP** allows devices to discover what services each other support, and what parameters to use to connect to them.

Insecure

- **RFCOMM** provides a simple reliable data stream to the user.
  - Many Bluetooth applications use RFCOMM because of its widespread support and publicly available API on most operating systems
- **TCS** (Telephony Control Protocol) and others

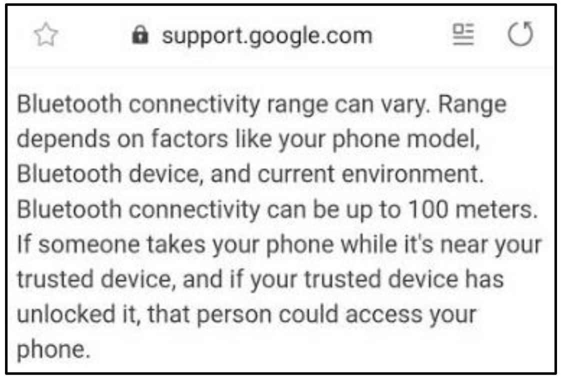
Secure

Bluetooth Security 101 - Proximity

Graduality of Bluetooth Proximity measure

- Bluetooth Connection (~ 100 m) – Android Smart Lock, Windows Dynamic Lock

Type	Operating Range
Class 1	~100m
Class 1.5	~30 m
Class 2	~ 10 m
Class 3	~ 1m



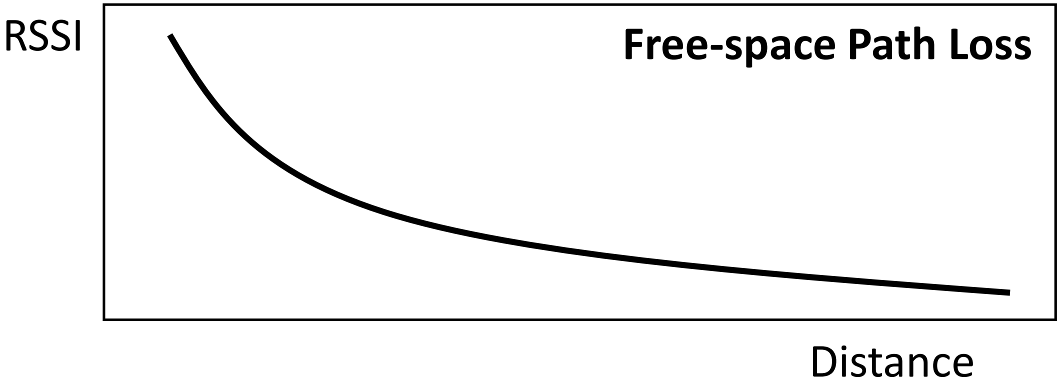
Bluetooth connectivity range can vary. Range depends on factors like your phone model, Bluetooth device, and current environment. Bluetooth connectivity can be up to 100 meters. If someone takes your phone while it's near your trusted device, and if your trusted device has unlocked it, that person could access your phone.

⇨

Smart Lock (Trusted Device)

Works up to **100 meters**

- Signal Strength (RSSI) – Windows Dynamic Lock
  - RSSI is one of the most widely used tech.
    to measure distances between two devices



How to make proximity authentication secure?

For Secure Bluetooth-based Proximity Authentication,

We need answers to the following questions:

1. **How can we utilize Bluetooth for Authentication?**
2. **How can we utilize Bluetooth for Proximity Checking?**

How to make proximity authentication secure?

! How to authenticate a trusted device using Bluetooth Security without adding additional messages for authentication?

- MAC Address: AA:BB:CC:DD:EE:FF
- Class of Device: Smart Watch
- Device Name: JUNG’s Watch

…

Connection

RSSI: -8 ..

RSSI: -10..

RSSI: -12 ..

How to detect when a trusted device is too far away? !

How to make proximity authentication secure?

Bluetooth Components/Features used in Android Smart Lock/Windows Dynamic Lock

Properties	Smart Lock	Dynamic Lock
MAC Address (Device Address)	●	●
Class of Device	X	●
RSSI	X	●
Link Establishment	X	●
Insecure Connection (SDP) (A Connection in Security Mode 4 - Level 0)	●	●
Secure Connection (e.g RFCOMM) (A connection in Security Mode 4 - Level 4)	●	●
A Message over RFCOMM	X	X

How to make proximity authentication secure?

Device Address (MAC Address)

- Devices are identified using a device address. (48 bits in length)

→ Easily exposed and easily manipulated (No Security)

```
# bdaddr –i hci0 xx:xx:xx:xx:xx:xx
```

Class of Device (COD)

- A value representing the type of device (e.g. Headphone: Connected for calls and audio)
  - Informational Purpose in the Device Discovery Phase

→CoD is checked using SDP (No Security)

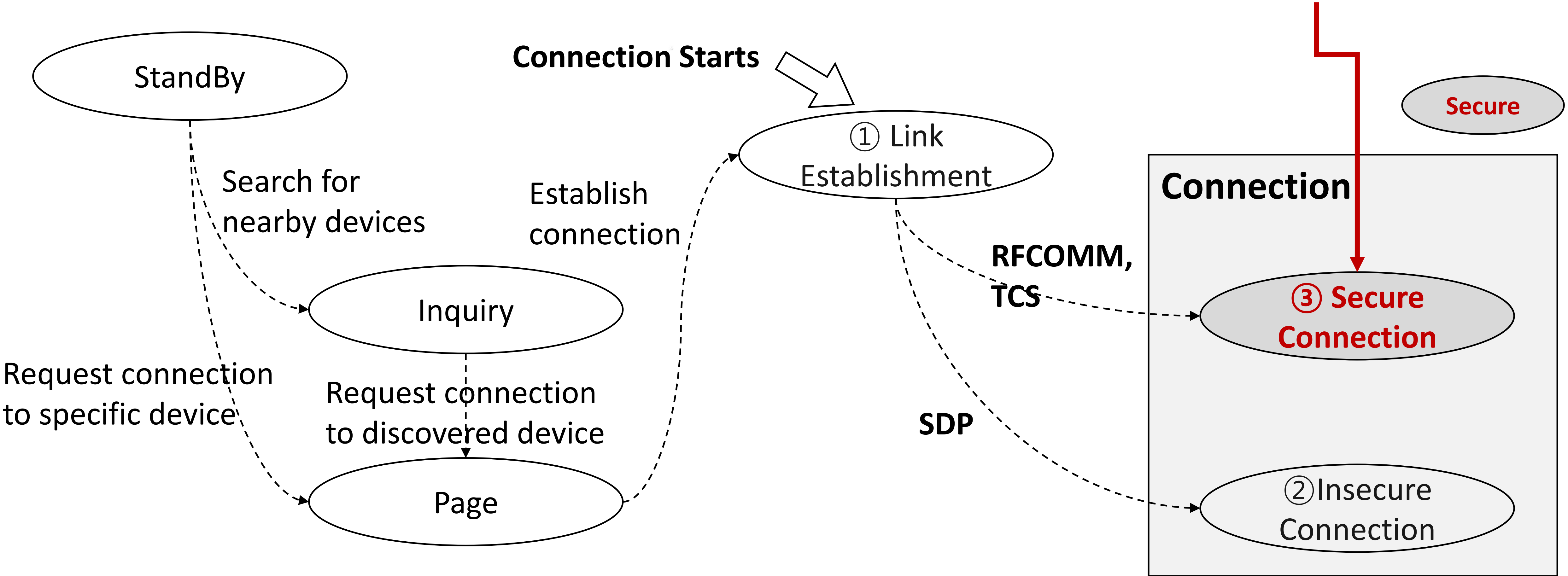
Received Signal Strength Indicator (RSSI)

→Secure? (No if a MAC spoofing attack succeeds)

How to make proximity authentication secure?

Security: ① Link Establishment, ② Insecure Connection, ③ Secure Connection,

and ④ message over RFCOMM



How to make proximity authentication secure? Summary

Properties	Smart Lock	Dynamic Lock	Authentication	Proximity (RSSI)
MAC Address (Device Address)	●	●	.	.
Class of Device	X	●	.	.
RSSI	X	●	.	Condi. Usable
Link Establishment	X	●	.	.
Insecure Connection (SDP)	●	●	.	.
Secure Connection (e.g RFCOMM)	●	●	Usable	.
A Message over RFCOMM	X	X	Usable	.

These properties should not be used or should be used with care.

AGENDA

- Bluetooth-based Proximity Authentication
- Preliminaries
- Security Analysis - Proposed Approach
- New Vulnerabilities
- Mitigations
- Conclusion

Security Analysis for Bluetooth-based Proximity Authentication

- Lesson #1
  - Device authentication methods over Bluetooth that are relying on untrusted properties of a connection, such as the MAC Address, are insecure.
- Lesson #2
  - Device proximity authentication methods over Bluetooth must check both device authentication and device proximity at the same time, via a secure channel.
- Our Hypothesis
  - Failing to follow either Lesson 1 or 2 would result in an insecure authentication

Security Analysis for Bluetooth-based Proximity Authentication

Methodology:

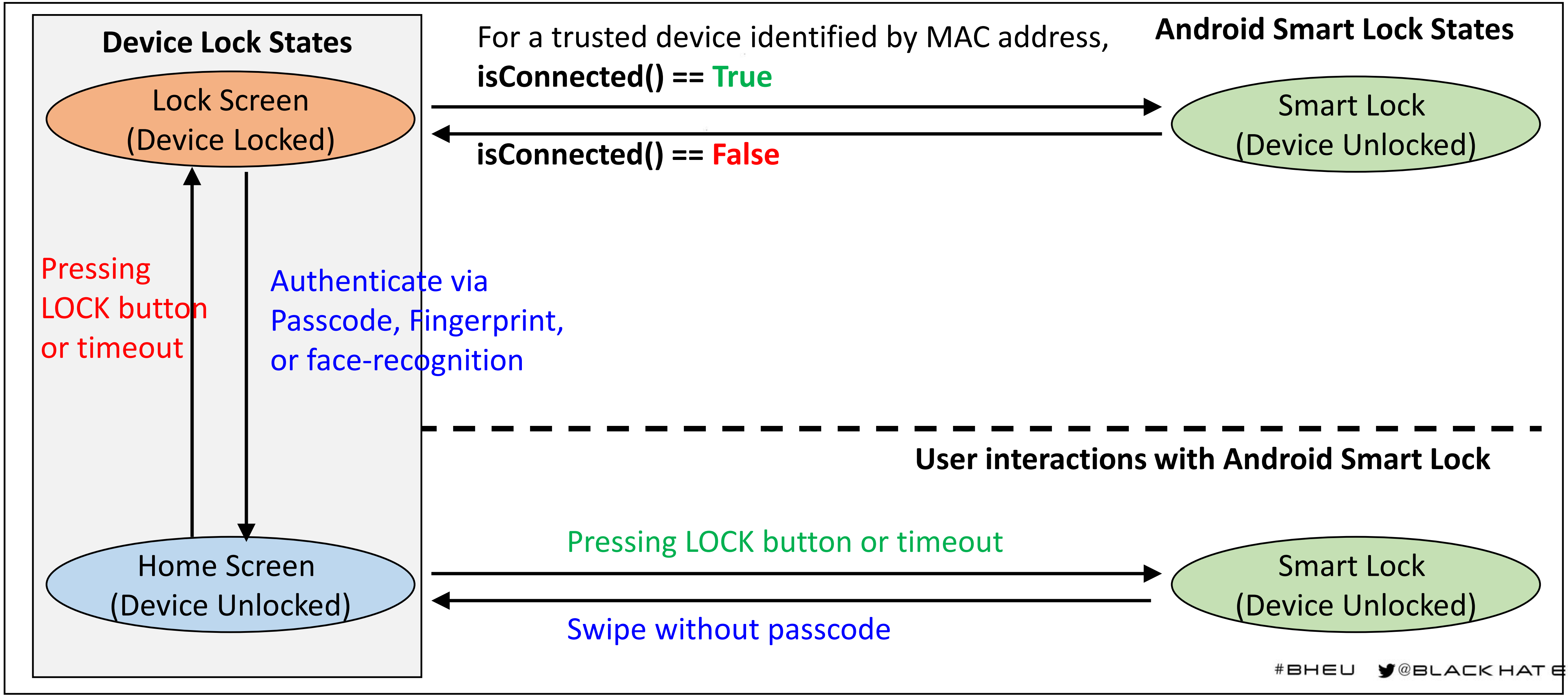
Analyze Authentication State Transition for Connection Security Properties

- Understand Authentication/Authorization State
  - When and how a device grant access?
  - How a device authenticate the other device?
  - How a device checks the proximity of the other device?

- Capture the corresponding connection state
  - What is the security level of the connection when authentication is done?

Working Example: Analyze the 2015 Attack

Authentication / Authorization State Diagram of Android Smart Lock



Device Lock States

Lock Screen (Device Locked)

Home Screen (Device Unlocked)

Pressing LOCK button or timeout

Authenticate via Passcode, Fingerprint, or face-recognition

For a trusted device identified by MAC address,

isConnected() == True

isConnected() == False

Android Smart Lock States

Smart Lock (Device Unlocked)

User interactions with Android Smart Lock

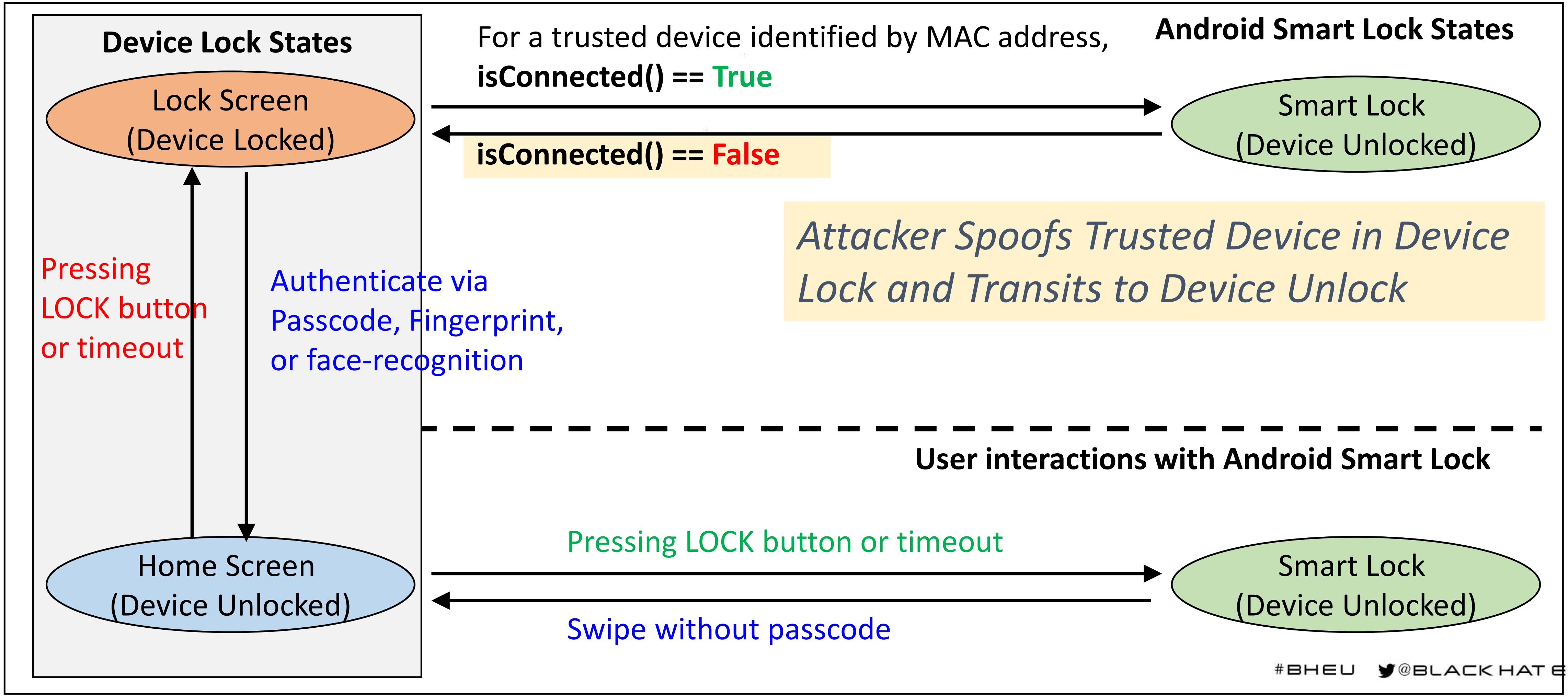
Pressing LOCK button or timeout

Swipe without passcode

Smart Lock (Device Unlocked)

Working Example: Analyze the 2015 Attack (cont'd)

Authentication / Authorization State Diagram of Android Smart Lock



Device Lock States

Lock Screen
(Device Locked)

Pressing
LOCK button
or timeout

Authenticate via
Passcode, Fingerprint,
or face-recognition

Home Screen
(Device Unlocked)

For a trusted device identified by MAC address,
isConnected() == True

isConnected() == False

Android Smart Lock States

Smart Lock
(Device Unlocked)

Attacker Spoofs Trusted Device in Device Lock and Transits to Device Unlock

User interactions with Android Smart Lock

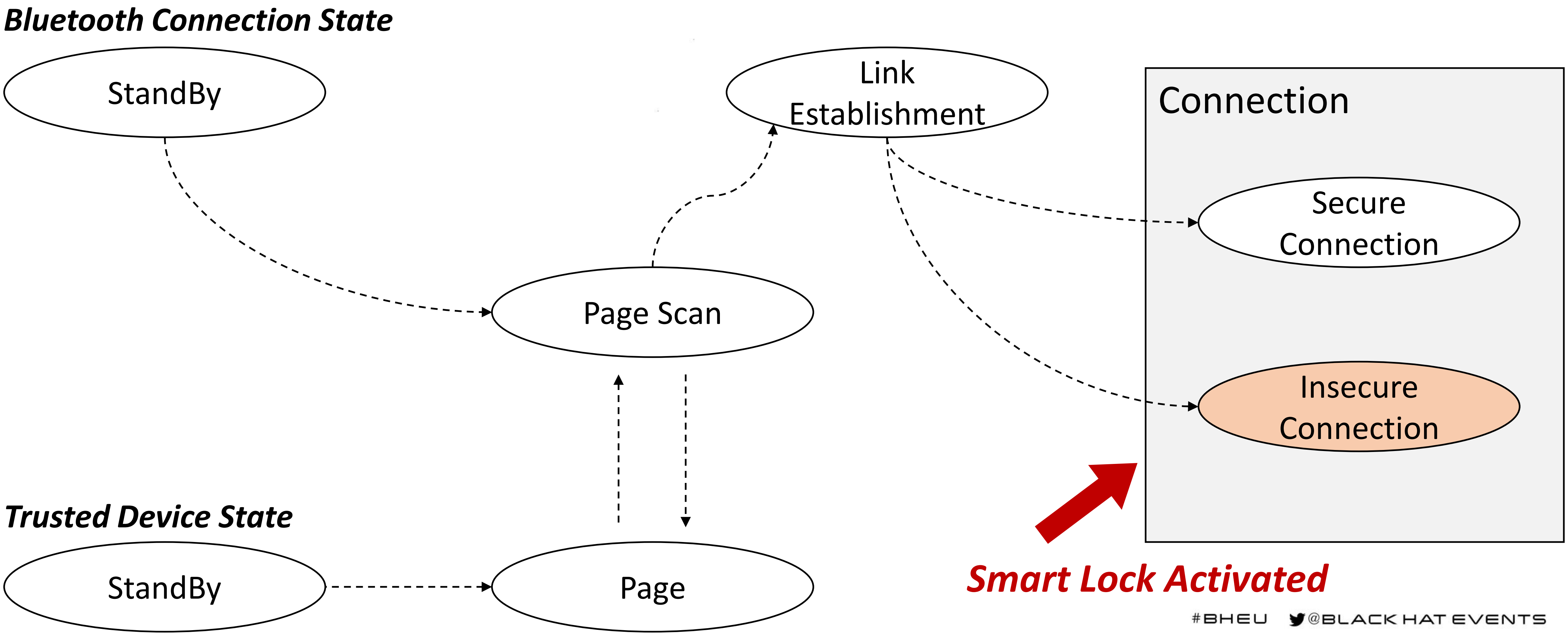
Pressing LOCK button or timeout

Swipe without passcode

Smart Lock
(Device Unlocked)

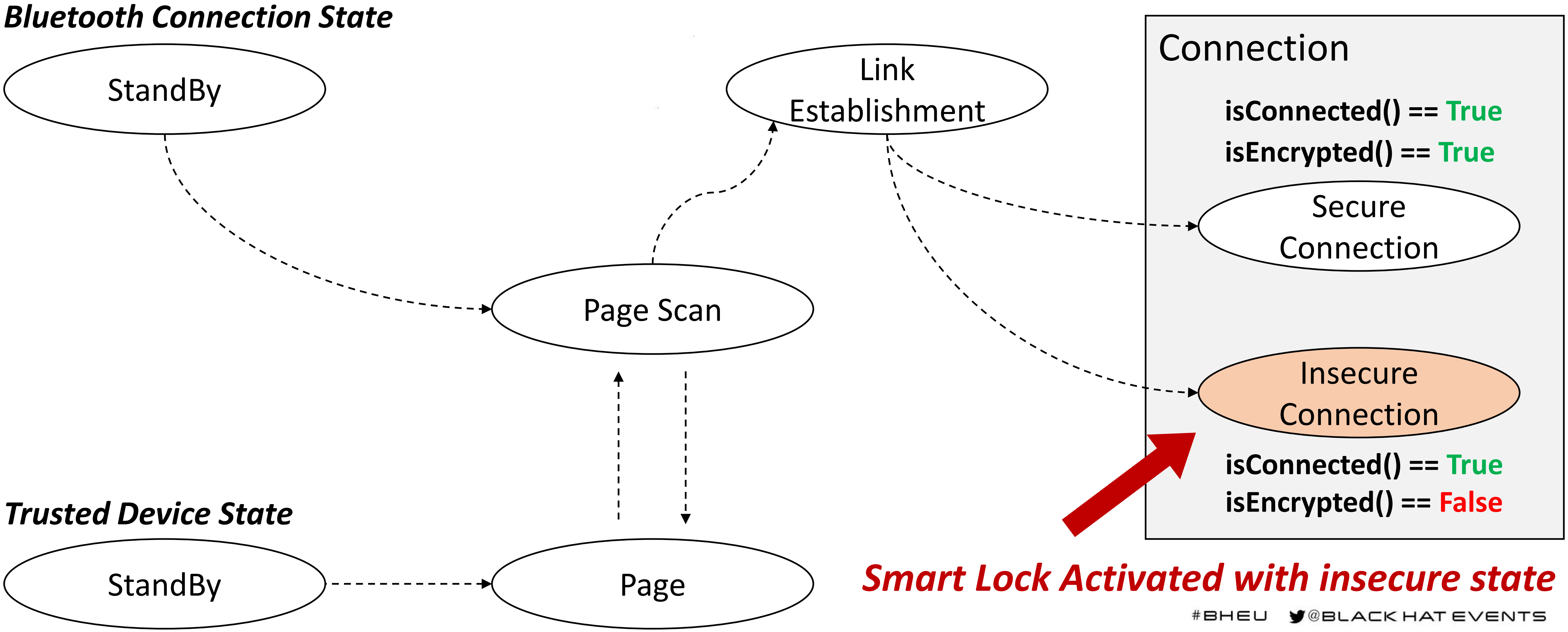
Working Example: Analyze the 2015 Attack (cont'd)

Connection State Diagram of Android Smart Lock



Working Example: Analyze the 2015 Attack (cont'd)

Connection State Diagram of Android Smart Lock

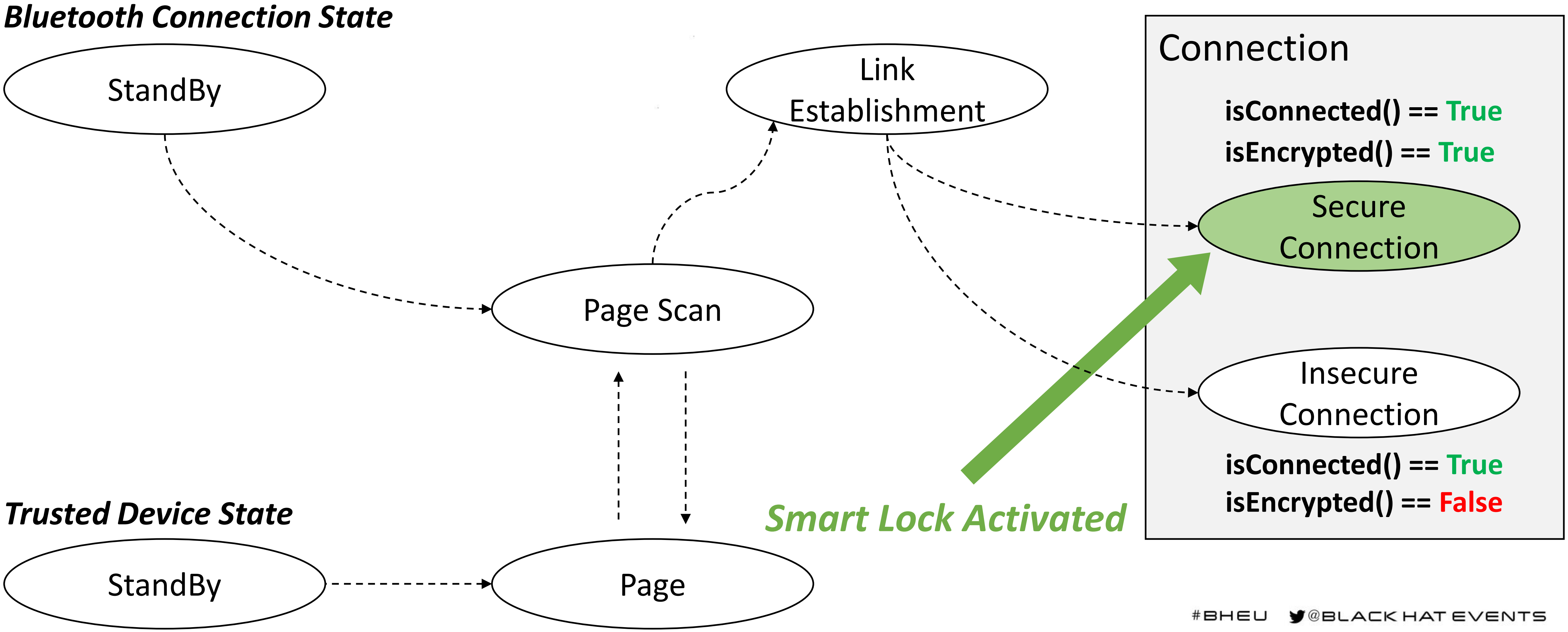


Smart Lock Activated with insecure state

Working Example: Analyze the 2015 Attack (cont'd)

black hat EUROPE 2019

Connection State Diagram of Android Smart Lock (Patched)



AGENDA

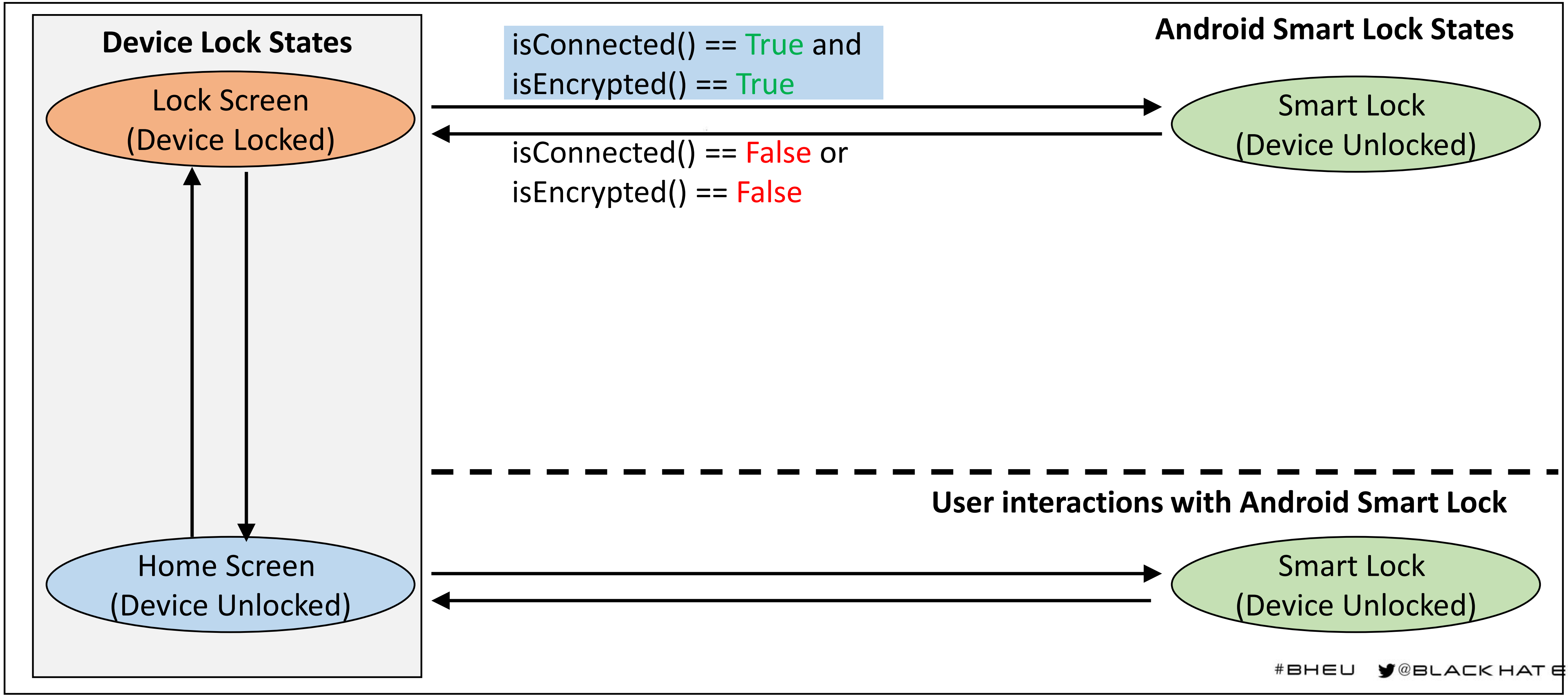
- Bluetooth-based Proximity Authentication
- Preliminaries
- Security Analysis – Proposed Approach
- ***New Vulnerabilities***
  - Smart Lock
  - Dynamic Lock
- Mitigations
- Conclusion

New Vulnerabilities

- Google resolved the issue by adding additional check `isEncrypted() == True`
  - Use only the connections from previously paired devices to enable Android Smart Lock
- Making an insecure Connection created by SDP can no longer unlock a device

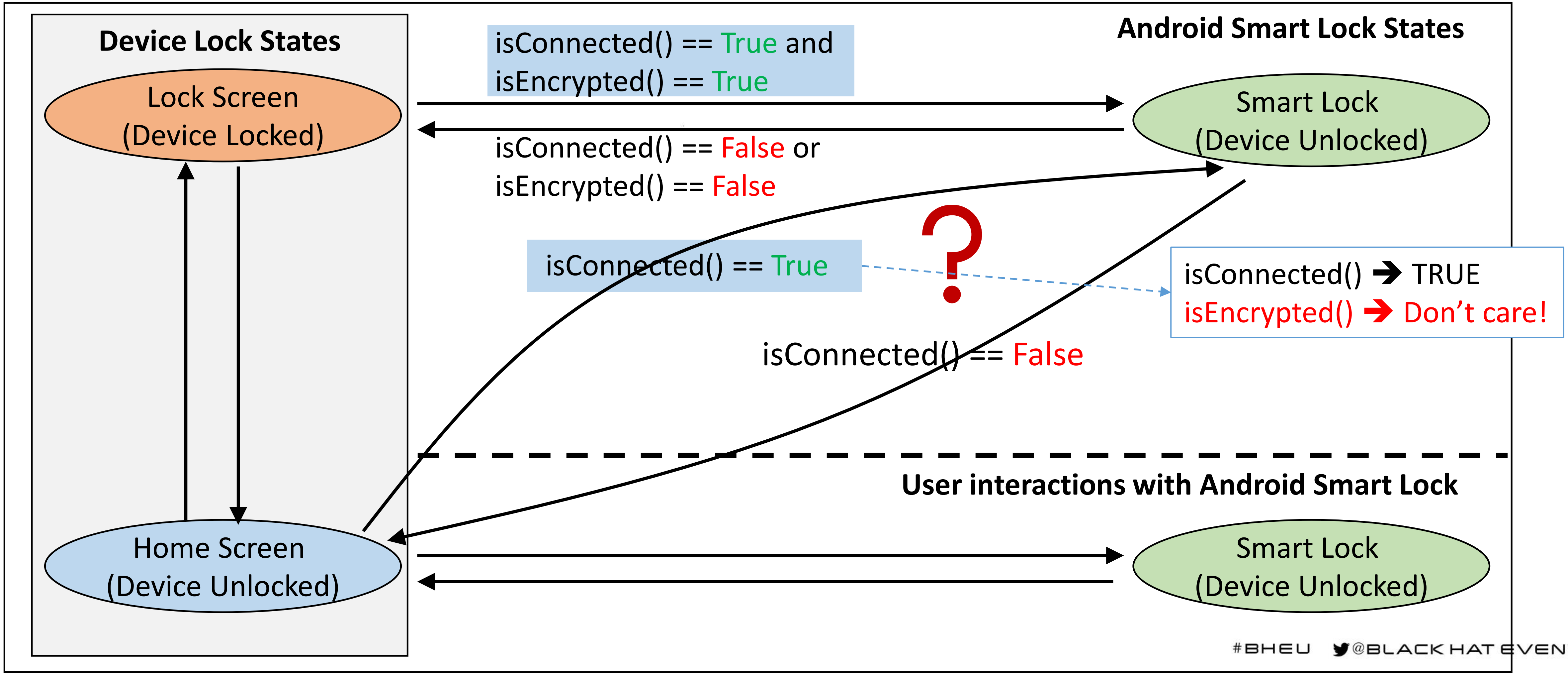
Root Cause: "Bluetooth Connection" is not Secure
→ Does every path become secured?

Authentication / Authorization State Diagram of Android Smart Lock



New Attack: Analyze Android Smart Lock in 2019

Authentication / Authorization State Diagram of Android Smart Lock



Device Lock States
Lock Screen (Device Locked)
Home Screen (Device Unlocked)
Android Smart Lock States
Smart Lock (Device Unlocked)
isConnected() == True and isEncrypted() == True
isConnected() == False or isEncrypted() == False
isConnected() == True
?
isConnected() → TRUE
isEncrypted() → Don't care!
isConnected() == False
User interactions with Android Smart Lock
Smart Lock (Device Unlocked)

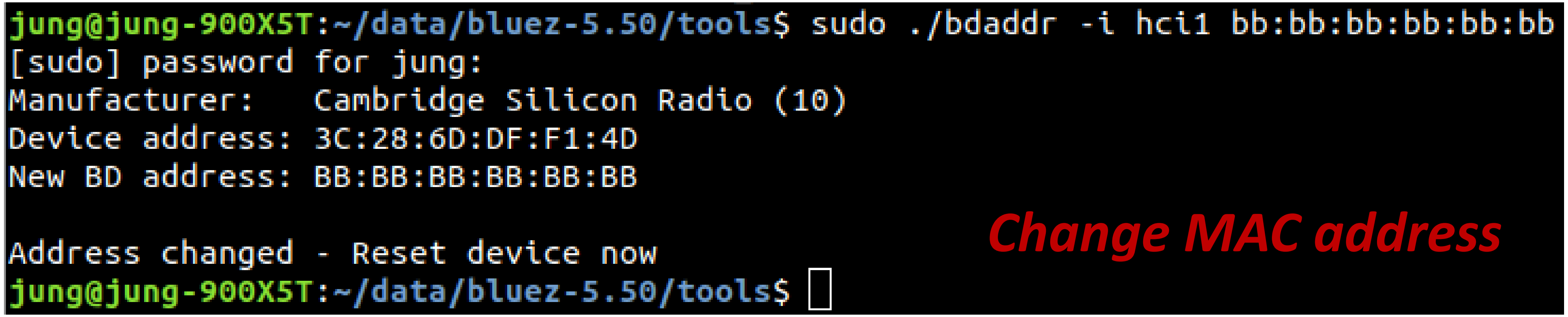
New Attack: Analyze Android Smart Lock in 2019 (cont'd)

DEMO TIME !

This demo describes the vulnerability reported in 2015 and the vulnerabilities we found.

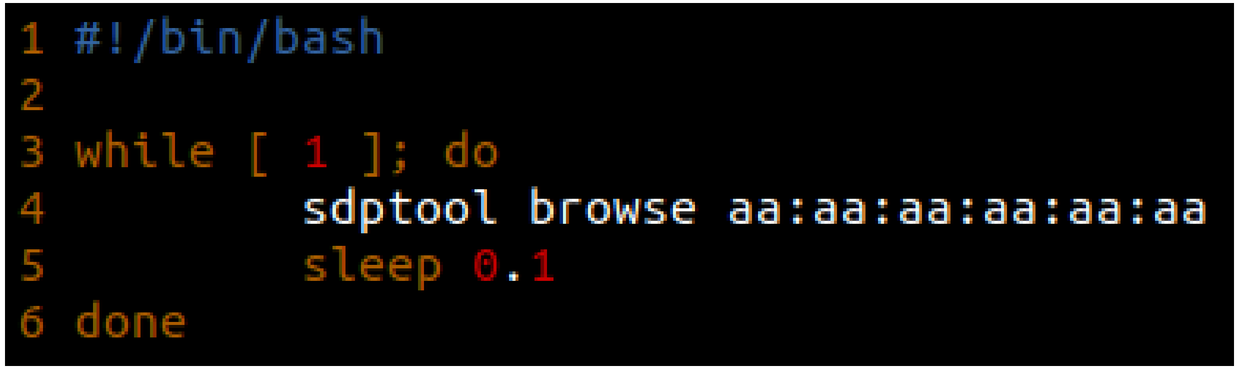
New Attack: Analyze Android Smart Lock in 2019 (cont'd)

Using the tools provided by bluez



```
jung@jung-900X5T:~/data/bluez-5.50/tools$ sudo ./bdaddr -i hci1 bb:bb:bb:bb:bb:bb
[sudo] password for jung:
Manufacturer:   Cambridge Silicon Radio (10)
Device address: 3C:28:6D:DF:F1:4D
New BD address: BB:BB:BB:BB:BB:BB

Address changed - Reset device now
jung@jung-900X5T:~/data/bluez-5.50/tools$
```

Change MAC address



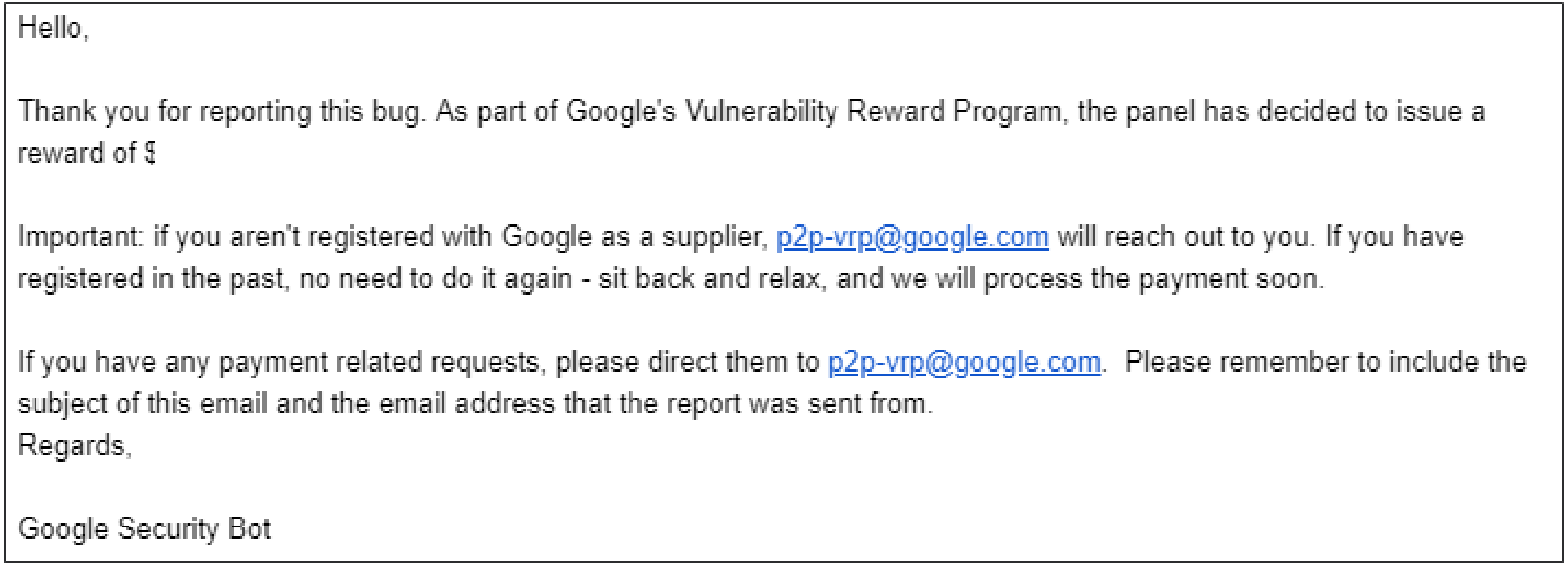
```
#!/bin/bash

while [ 1 ]; do
        sdptool browse aa:aa:aa:aa:aa:aa
        sleep 0.1
done
```

- *SDP creates a temporary connection*
- *Calling SDP repeatedly creates a* ***persistent connection***

New Attack: Analyze Android Smart Lock in 2019 (cont'd)

Responsible Disclosure

- April 5 Report / April 16 Acceptance / July 17 Complete Patch



Hello,

Thank you for reporting this bug. As part of Google's Vulnerability Reward Program, the panel has decided to issue a reward of $

Important: if you aren't registered with Google as a supplier, p2p-vrp@google.com will reach out to you. If you have registered in the past, no need to do it again - sit back and relax, and we will process the payment soon.

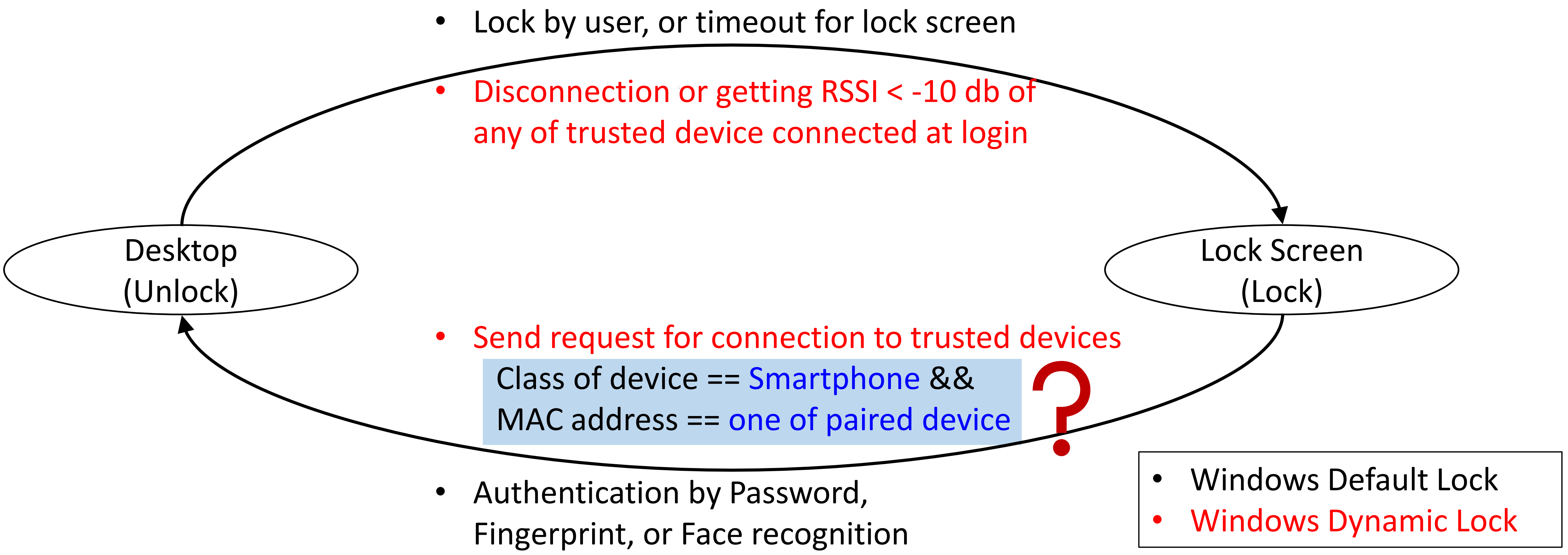
If you have any payment related requests, please direct them to p2p-vrp@google.com. Please remember to include the subject of this email and the email address that the report was sent from.
Regards,

Google Security Bot

New Attacks: Analyze Windows Dynamic Lock

black hat EUROPE 2019

Authentication / Authorization State Diagram of Windows Dynamic Lock



- Lock by user, or timeout for lock screen
- Disconnection or getting RSSI < -10 db of any of trusted device connected at login

Desktop (Unlock)

Lock Screen (Lock)

- Send request for connection to trusted devices

Class of device == Smartphone &&
MAC address == one of paired device

?

- Authentication by Password, Fingerprint, or Face recognition

- Windows Default Lock
- Windows Dynamic Lock

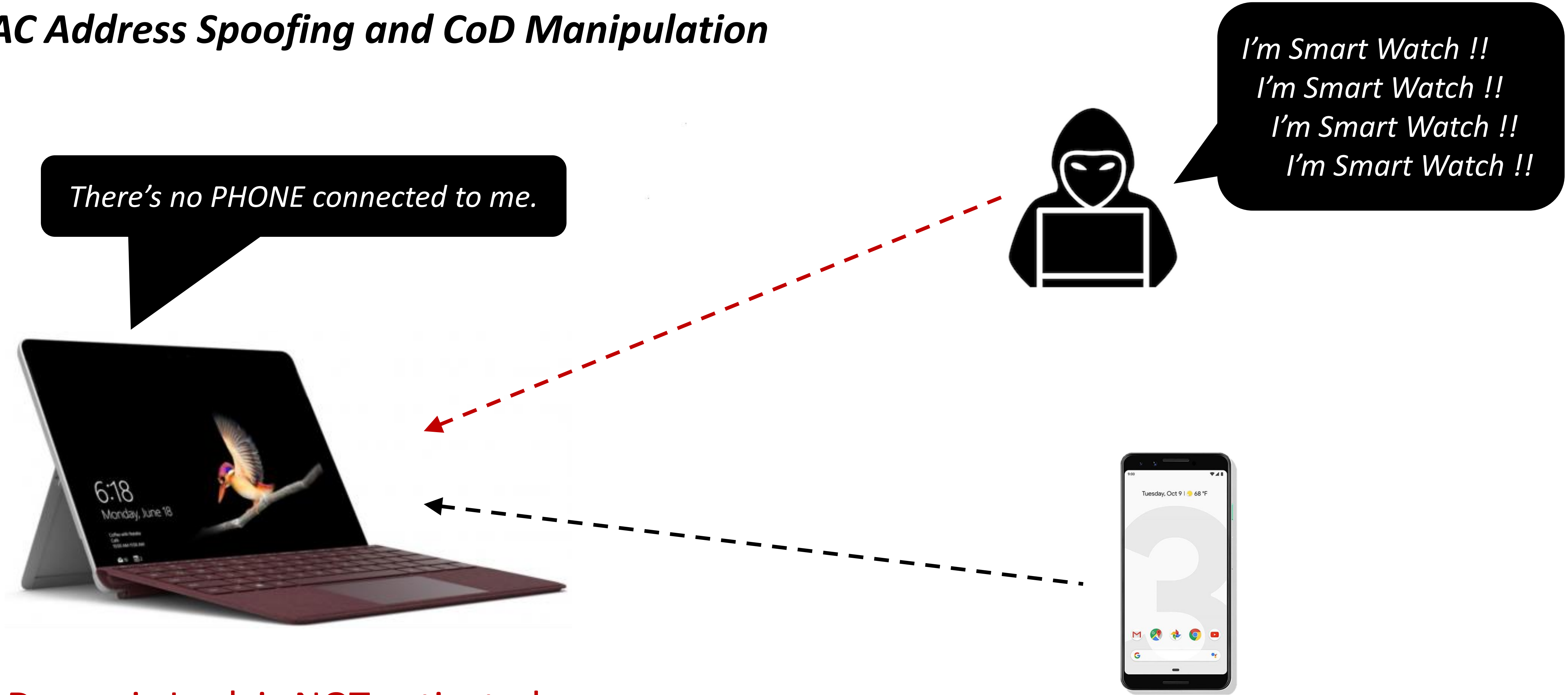
New Attacks: Analyze Windows Dynamic Lock (cont'd)

DEMO TIME !

We will use <u>Surface Go (Windows 10 1909)</u>
to demonstrate the vulnerability
of Windows Dynamic Lock.

New Attacks: Analyze Windows Dynamic Lock (cont'd)

MAC Address Spoofing and CoD Manipulation



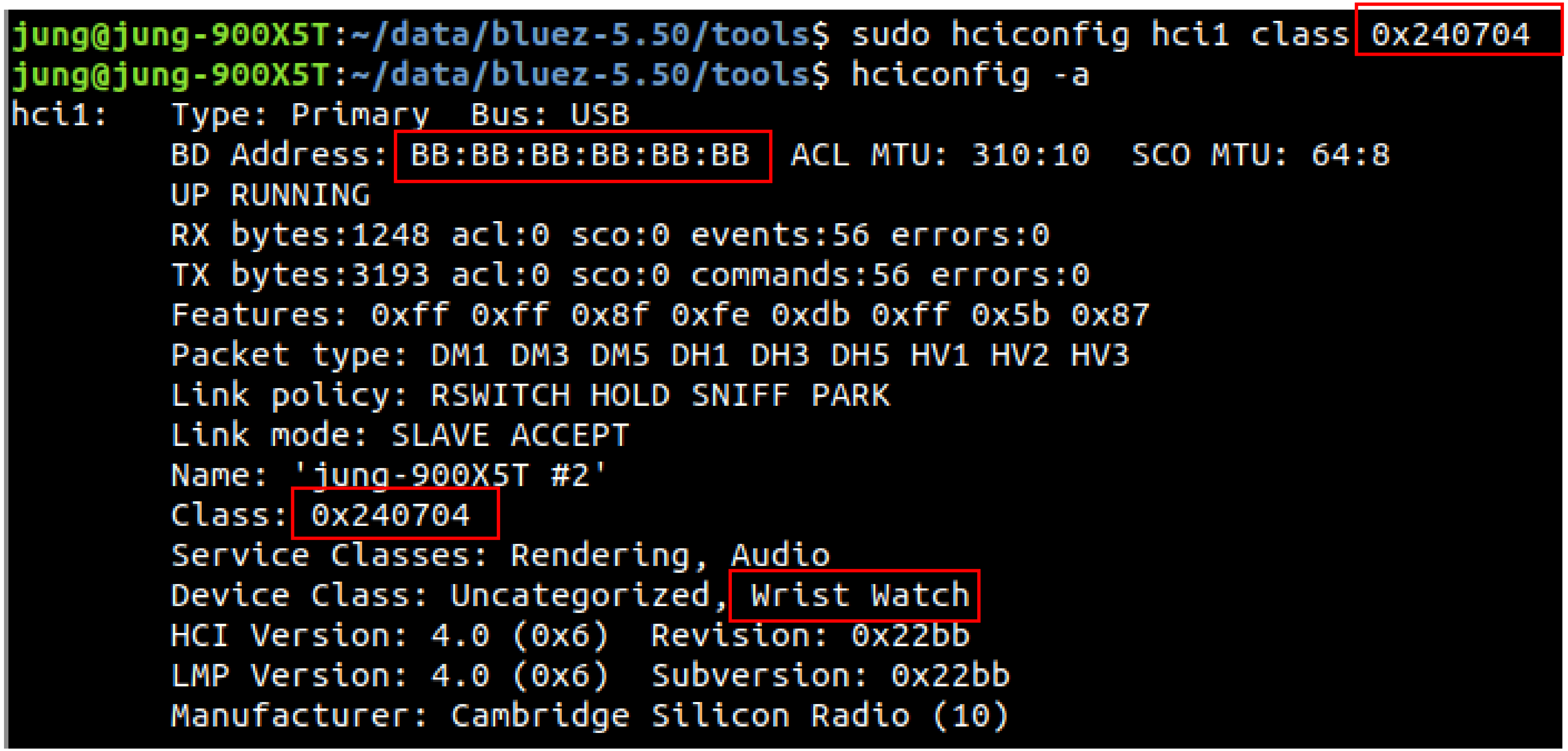
I'm Smart Watch !!
I'm Smart Watch !!
I'm Smart Watch !!
I'm Smart Watch !!

There's no PHONE connected to me.

Dynamic Lock is NOT activated.

New Attacks: Analyze Windows Dynamic Lock (cont'd)

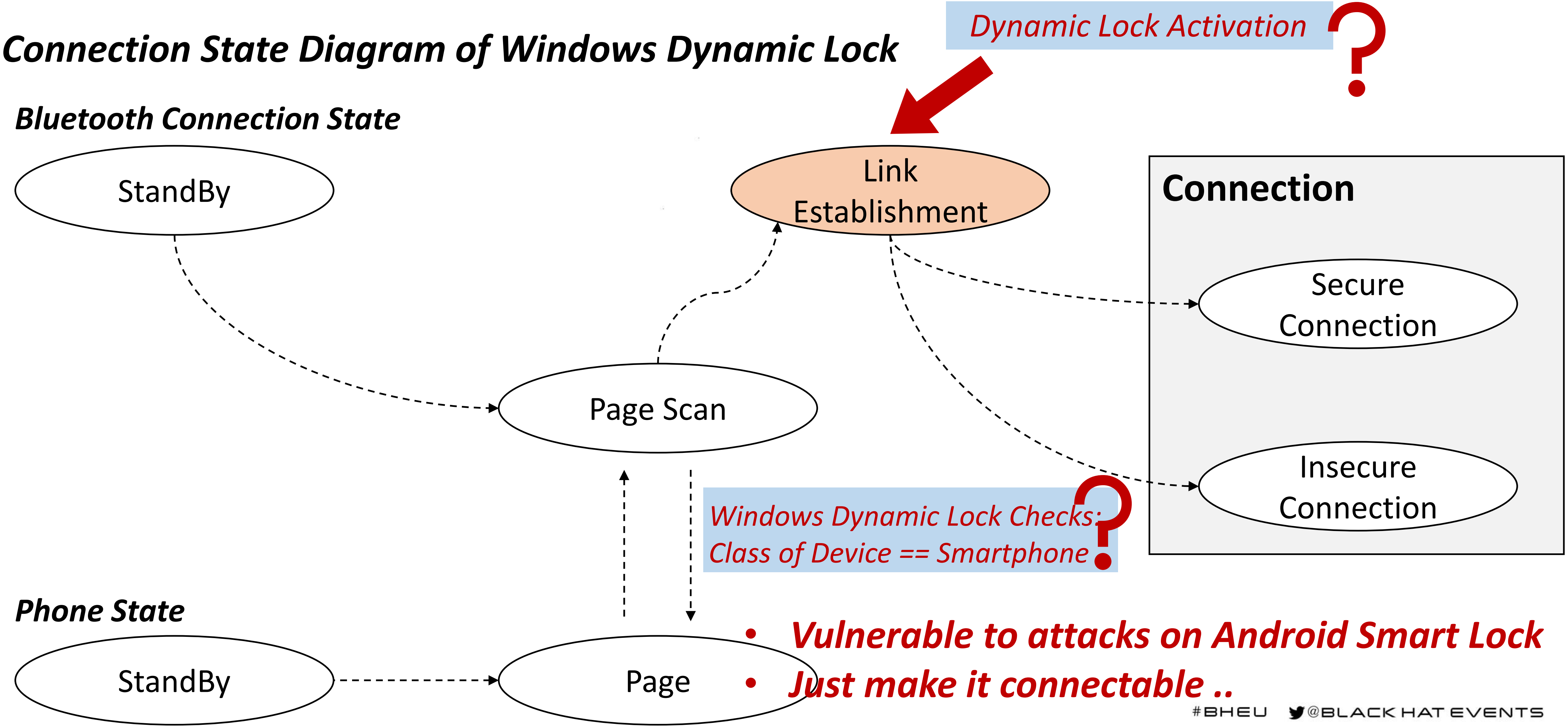
MAC Address Spoofing and CoD Manipulation



```
jung@jung-900X5T:~/data/bluez-5.50/tools$ sudo hciconfig hci1 class 0x240704
jung@jung-900X5T:~/data/bluez-5.50/tools$ hciconfig -a
hci1:   Type: Primary  Bus: USB
        BD Address: BB:BB:BB:BB:BB:BB  ACL MTU: 310:10  SCO MTU: 64:8
        UP RUNNING
        RX bytes:1248 acl:0 sco:0 events:56 errors:0
        TX bytes:3193 acl:0 sco:0 commands:56 errors:0
        Features: 0xff 0xff 0x8f 0xfe 0xdb 0xff 0x5b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH HOLD SNIFF PARK
        Link mode: SLAVE ACCEPT
        Name: 'jung-900X5T #2'
        Class: 0x240704
        Service Classes: Rendering, Audio
        Device Class: Uncategorized, Wrist Watch
        HCI Version: 4.0 (0x6)  Revision: 0x22bb
        LMP Version: 4.0 (0x6)  Subversion: 0x22bb
        Manufacturer: Cambridge Silicon Radio (10)
```

New Attacks: Analyze Windows Dynamic Lock (cont'd)

Connection State Diagram of Windows Dynamic Lock



Bluetooth Connection State

StandBy

Page Scan

Link Establishment

Dynamic Lock Activation

?

Connection

Secure Connection

Insecure Connection

Windows Dynamic Lock Checks: Class of Device == Smartphone

?

Phone State

StandBy

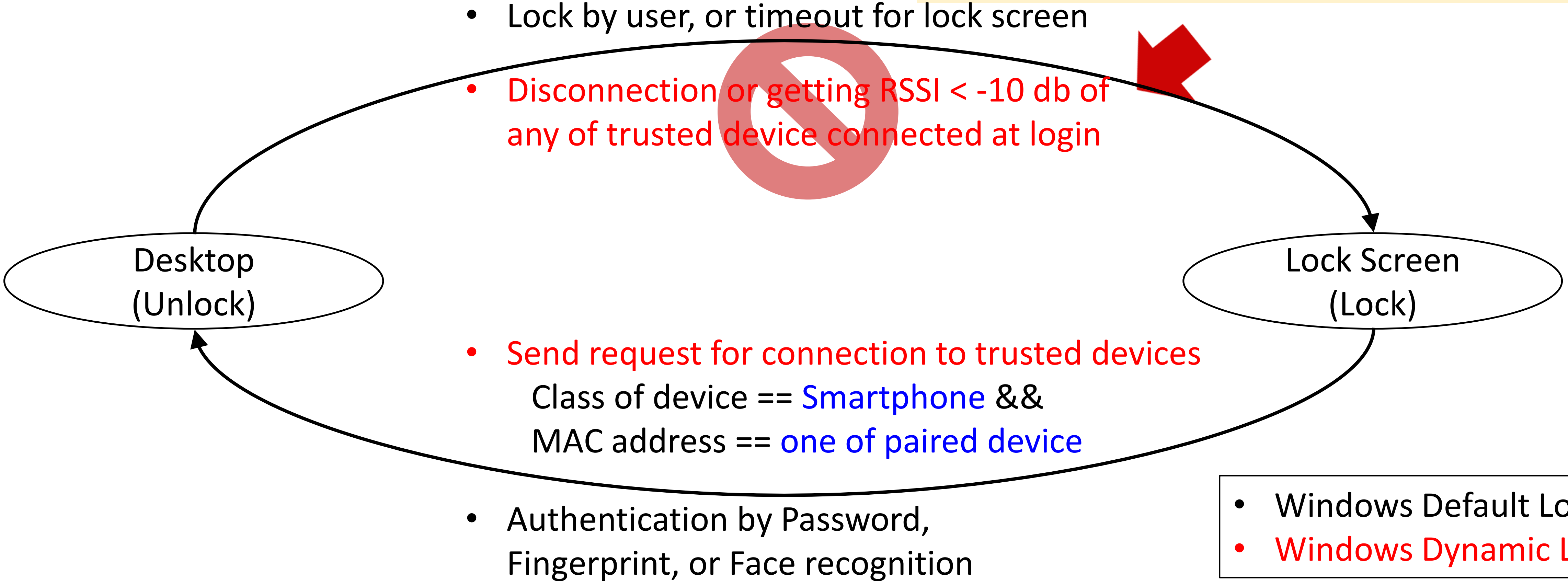
Page

- ***Vulnerable to attacks on Android Smart Lock***
- ***Just make it connectable ..***

New Attacks: Analyze Windows Dynamic Lock (cont'd)

Proximity Manipulation

If an attacker attempts to connect to a laptop using SDP, the connection can be maintained and signal strength can be high.



- Lock by user, or timeout for lock screen
- Disconnection or getting RSSI < -10 db of any of trusted device connected at login

Desktop (Unlock)

Lock Screen (Lock)

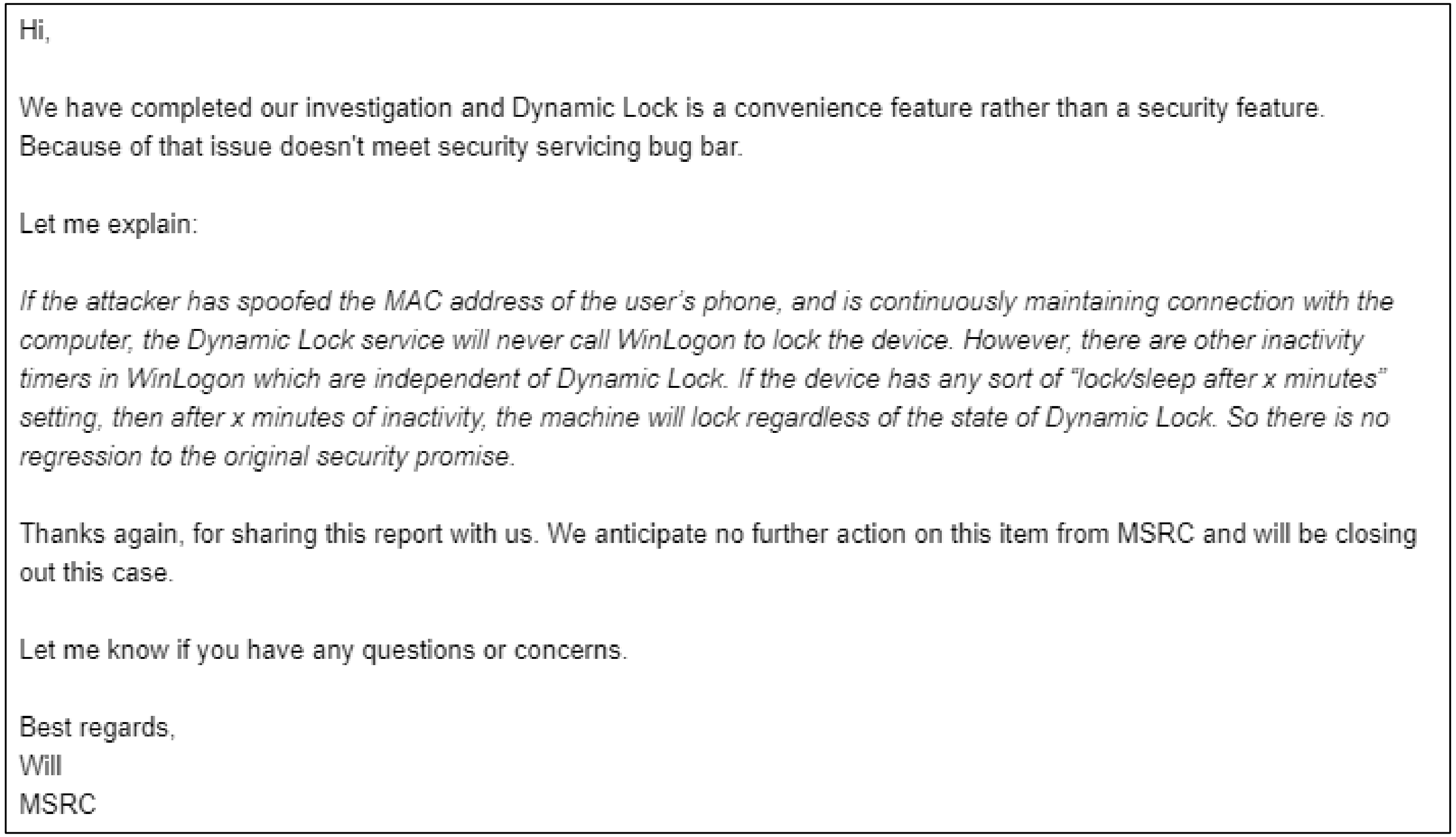
- Send request for connection to trusted devices
  Class of device == Smartphone &&
  MAC address == one of paired device
- Authentication by Password, Fingerprint, or Face recognition

- Windows Default Lock
- Windows Dynamic Lock

New Attacks: Analyze Windows Dynamic Lock (cont'd)

Responsible Disclosure (May 14)

- Windows Dynamic Lock does not affect to the original security promise (by Microsoft)
- Even if Windows Dynamic Lock is not activated, the laptop is locked by the lock screen timeout



Hi,

We have completed our investigation and Dynamic Lock is a convenience feature rather than a security feature. Because of that issue doesn't meet security servicing bug bar.

Let me explain:

If the attacker has spoofed the MAC address of the user's phone, and is continuously maintaining connection with the computer, the Dynamic Lock service will never call WinLogon to lock the device. However, there are other inactivity timers in WinLogon which are independent of Dynamic Lock. If the device has any sort of "lock/sleep after x minutes" setting, then after x minutes of inactivity, the machine will lock regardless of the state of Dynamic Lock. So there is no regression to the original security promise.

Thanks again, for sharing this report with us. We anticipate no further action on this item from MSRC and will be closing out this case.

Let me know if you have any questions or concerns.

Best regards,
Will
MSRC

AGENDA

- Bluetooth-based Proximity Authentication
- Preliminaries
- Security Analysis – Proposed Approach
- New Vulnerabilities
- ***Mitigations***
- Conclusion

Mitigations

Know what is provided by Bluetooth Security, and use only secure components of a Bluetooth connection.

- Connecting to a previously paired, trusted device is not necessarily secure
  - Bluetooth connection can be in one of security level (0 – 4)
  - Only the encrypted connection (Security Level 4) is secure and trusted

- When to use encrypted connection?
  - Use only the encrypted connection for Authentication
  - If the functionality is not related to the device’s security, you may use unencrypted connection

Mitigations

Completely Cut-off insecure authentication / connection state transition paths

- Obtain the state diagram of both authentication and connection management logic
- Analyze the diagram for any insecure state transition paths
  - Identify and apply fix for all insecure paths
- Lesson: Google was aware of the root cause of the 2015 vulnerabilities, but its fix leaves an alternative path that misses security check (**`isEncrypted() == True`**)

Applying this Analysis in the Software Development Lifecycle (SDL)

→Verify that authentication is not triggered by Untrusted Properties

- Vulnerability Detection Tool
  - Simulate the attack for detecting potential vulnerabilities

Mitigations

Bind insecure properties with SECURE components

- Obtain RSSI only from encrypted connection
  - Check if the connection is in the Security Level 4 before measuring RSSI

AGENDA

- Bluetooth-based Proximity Authentication
- Preliminaries
- Security Analysis – Proposed Approach
- New Vulnerabilities
- Mitigations
- ***Conclusion***

Conclusion

- Convenient Bluetooth-based proximity authentication methods could result in an insecurity

- We proposed a method to analyze the security following Bluetooth Security 101 and found several new vulnerabilities:
  - A new vulnerability in Android Smart Lock bypassing was proposed
    - The first vulnerability reported in 2015 was improperly fixed, allowing attackers bypass the lockscreen
  - Four new vulnerabilities in Windows Dynamic Lock were proposed
    - It utilizes RSSI value from a connection, however, does not check if the connection is trusted or not
    - It is vulnerable to attacks on Smart Lock (using SDP)

- The root cause can be defeated by applying the proposed analysis method.

Take Home Messages

- Back to Basic: Don’t trust anything before verification
  - Bluetooth provides both secure and insecure features.
  - Check if the connection is encrypted
  - Check if the RSSI value is measured for an encrypted connection

- Apply system-state/Bluetooth analysis in the Security Development Lifecycle (SDL)
  - Take account the state of both the system and Bluetooth connection
  - SHOULD NOT authorize access if connection is untrusted
    - SHOULD NOT have a state transition to authorized state via untrusted values

- Try our vulnerability detection tool to your favored Bluetooth authentication methods
  - https://github.com/0-10000/Bluemaster

Thank you

for your attention !

yman.jung@samsung.com
junbum.shin@samsung.com
yeongjin.jang@oregonstate.edu

Please contact us by e-mail for more details