How Google Killed Two-Factor Authentication (and the reactions)

http://www.few.vu.nl/~vvdveen/bandroid.html

Mixed reactions from Google, but we have their attention. iOS and Windows Phone have similar remote install features YET

Easy, version-independent fix: explicit activation

Mobile phone based 2FA seems doomed

The Media is (mostly) clueless

Radhesh Krishnan
Herbert Bos
Victor van der Veen
VU University Amsterdam
System and Network Security Group

Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)
How Google Killed Two-Factor Authentication (and the reactions)

http://www.few.vu.nl/~vvdveeen/bandroid.html

Mixed reactions from Google, but we have their attention
iOS and Windows Phone have similar remote install features YET

Easy, version-independent fix: explicit activation

Mobile-phone based 2FA seems doomed

The Media is (mostly) clueless

Radhesh Krishnan
Herbert Bos
Victor van der Veen
VU University Amsterdam
System and Network Security Group

Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)
Mixed reactions from Google, but we have their attention
iOS and Windows Phone have similar remote install features
...but no API to read SMS messages YET

Easy, version-independent fix: explicit activation

Mobile-phone based 2FA seems doomed
financial institutions will come to the same conclusion

The Media is (mostly) clueless

Radhesh Krishnan
Herbert Bos
Victor van der Veen
VU University Amsterdam
System and Network Security Group

Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)
Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)
How Google Killed Two-Factor Authentication (and the reactions)

http://www.few.vu.nl/~vvdveen/bandroid.html

Mixed reactions from Google, but we have their attention
iOS and Windows Phone have similar remote install features YET

Easy, version-independent fix: explicit activation
Mobile-phone based 2FA seems doomed
All institutions will come to the same conclusion

The Media is (mostly) clueless

Radhesh Krishnan
Herbert Bos
Victor van der Veen
VU University Amsterdam
System and Network Security Group

Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)
Two-Factor Authentication

Mobile Phone Two-Factor Authentication

- e-banking
  - (1) Hi, my name is ...
  - (2) Please transfer €100,- to X
  - (3) Code XX-123
  - (4) Code XX-123, to transfer €100 to X

2FA Threat-Model
Man-in-the-Browser (MitB)

- Compromised PC
  - Dridex
  - SpyEye
  - Carberp
  - Zeus
  - ...

Banking credit

2FA stops attacks

Attackers can initiate transactions, but no longer confirm them

New Attack Variants

1) Modify ongoing transactions:
   - Please transfer €100,- to X
   - Please transfer €100,- to Y
   - Mitigated by including target account information in TAN codes:
     - Code YY-123, to transfer €100 to Y (instead of X)

2) Infect the 2nd Factor:
   - Once the PC is in control, SPAM the mobile
   - Social engineer the victim into installing malware

Malicious apps capable of forwarding SMS data

Zeus in the Mobile | SpitMo | CitMo | ...

Google Bouncer

Detects and removes malicious apps from the Playstore
- Static Analysis
- Dynamic Analysis

Current malware relies on 'sideloading'
- Allow app installation from unknown sources

Two-Factor Authentication

Withdraw money from an ATM
- Insert your bank card (that you possess)
- Enter your pin-code (that you know)
- Get your money

Two-factor authentication is expensive...
...so use something everybody has...

SMS

Multi-Factor Authentication

Patented in 1984

Use of multiple components for identification
- Something you know (password, pin-code ...)
- Something you possess (bank card, token, ...)
- Something you are (fingerprint, iris, ...)

Relies on the separation of components
An attacker needs to control all

Inflecting the 2nd Factor
is not straightforward

A user must explicitly allow app installation

Two-Factor Authentication
the reactions
Multi-Factor Authentication

Patented in 1984

Use of multiple components for identification
  • Something you **know** (password, pin-code ...)
  • Something you **possess** (bank card, token, ...)
  • Something you **are** (fingerprint, iris, ...)

Relies on the separation of components
  An attacker needs to control all

*Infected the 2nd Factor*
Two-Factor Authentication

Withdraw money from an ATM
• Insert your bank card (that you possess)
• Enter your pin-code (that you know)
• Get your money

Two-factor authentication is expensive...
...so use something everybody has...

SMS

New Attack Variants
Mobile Phone Two-Factor Authentication
e-banking

(1) Hi, my name is ...
(2) Please transfer €100,- to X
(4) Code XX-123

(3) Code XX-123, to transfer €100 to X

2FA Threat-Model
(3) Code XX-123, to transfer €100 to X

2FA Threat-Model
Man-in-the-Browser (MitB)

Compromised PC
- Dridex
- SpyEye
- Carberp
- ZeuS
- ...

Banking credentials stolen

2FA stops attacks

Attackers can initiate transactions, but no longer confirm them
New Attack Variants

1) Modify ongoing transactions:

Please transfer €100, to X
Please transfer €100,- to Y

Mitigated by including target account information in TAN codes:
Code YY-456, to transfer €100 to Y (instead of to X)

2) Infect the 2nd Factor:
• Once the PC is in control, SPAM the mobile
• Social engineer the victim into installing malware

Malicious apps capable of forwarding SMS data

Zeus in the Mobile | SpitMo | CitMo | ...
Infecting the 2nd Factor is not straightforward

A user must explicitly allow app installation

Google Bouncer

Detects and removes malicious apps from the Playstore
  • Static Analysis
  • Dynamic Analysis

Current malware relies on 'sideloaded'
  • Allow app installation from unknown sources
Let's Integrate Everything!

The "Internet of Things" Web 2.0?

Let's Synchronize...
- your browser
- Because that is the smart in smartphone
- your devices
- ... with your phone!

2. Apptivation

App is installed via remote install

We need only one user interaction:
1. Open the app directly (via app-icon) or install notification
2. Click a custom URI (myapp://open.me)

3. Intercept Messages

Control over the phone
Install a SMS receiver, for each incoming SMS:
1. Store it
2. Detect TAN/2FA codes and delete these (pre-alert code)
3. Webview request to our malicious server
4. Download and execute a connect back (remote shell) binary

Control over the browser
1. Log into e-banking environment
2. Initiate transaction
3. Confirm with intercepted TAN

The Google Way

Manage your phone from your browser
- Locate it
- Wipe it (in case of emergency)
- Install apps

Yes, it's shown in your browser only

The Google Way

Sure this is safe
You can only remote install apps from Play.
Google Bouncer will protect you

Apps are inactive after installation
- A user must start them once explicitly
- Only then can we start on boot, intercept messages, ...

1. Bypassing Bouncer

Already done in the past, multiple times
Bouncer evolves though

Assume that Bouncer cannot detect malicious code

Why not upload a benign app instead?
A simple mechanism that fetches items from remote server
We craft the app code, and server:
- Memory corruption
- Use known WebView vulnerabilities

Use Demo

You can find us in Google Play!

We are on good terms with the banks...
...so let's break something more fun instead

Google Authenticator!
Because that's not SMS based

The Fix

Google
1. Always require on phone confirmation for app installs
2. Do not allow app activation through clicked URIs
3. Disable the remote install feature
4. Perhaps look at our hiding tricks?

The user
1. Watch out for unknown app installs
2. Use a separate account for Android

The Google Authenticator user
- (Logout)
- Use a non-android phone for your backup phone number

How Google Killed (and the
Let's Integrate Everything!

{ubiquitous|anywhere|pervasive} computing
The "Internet of Things"
Web 2.0?

Let's Synchronize...
  • your browser
  • your e-mail
  • your contacts
  • ...
... with your phone!

2. Apptivation
The Google Way

Manage your phone from your browser

• Locate it
• Wipe it (in case of emergency)
• Install apps!

Permissions are shown in your browser only
No phone interaction

1. Bypassing Bouncer
The Google Way
Sure this is safe
You can only remote install apps from Play. Google Bouncer will protect you.

Apps are **inactive** after installation
- A user must start them once explicitly
- Only then can we start on boot, intercept messages, ...

Elevate MitB to MitMo
Elevate MitB to MitMo
And intercept SMS messages

Assuming control over the browser, we need to:

1. Bypass Bouncer
2. Steer the user into activating the app
   Required only once
3. Intercept SMS... and profit!

The Fix
1. Bypassing Bouncer

Already done in the past, multiple times
Bouncer evolves though

Assume that Bouncer can detect malicious code

Why not upload a vulnerable app instead?

A simple news app that fetches items from remote server x
We can control the app code, and server x:
  - Jekyll on iOS: When Benign Apps Become Evil (Usenix Sec '13)
    -Insert a memory corruption
      - use known webview vulnerabilities
      - ...
    - Matthias Neugschwandtner, Martina Lindorfer @ LEET '13

You can find us in Google Play!

Live Demo
2. Apptivation

App is installed via remote-install

We need only one user interaction:
1. Open the app directly (via app-icon) (or install notification)
2. Click a custom URI (myapp://open.me)

3. Intercept Messages
Direct open

"Hey, what is this app?"
Hey, what is this app?
Custom URI
Abuse synchronization

- Send a mail to self
- Replace links in Google Documents
- Post a URL to the user's Facebook wall
- Replace 'recent tabs'
- ...
- Replace bookmarks

1. We control the browser: replace bookmarks, retain functionality
2. Bookmarks now link to our controlled web server
3a. (old Chrome) The loaded webpage triggers an intent redirect
3b. (new Chrome) The webpage redirects after a user touch
3. Intercept Messages

Control over the phone
Install a SMS receiver, for each incoming SMS:
1. store it
2. detect TAN/2FA codes and delete these (pre-kitkat only)
3. webview request to our malicious server
4. download and execute a connect back (remote shell) binary

Control over the browser
1. Log into e-banking environment
2. Initiate transaction
3. Confirm with intercepted TAN
We are on good terms with the banks... 
...so let's break something more fun instead

Google Authenticator!
Because that's not SMS based
right?

A few hours after talking to Nick - head of Android platform security - Kralevich, our app and developer account got banned...
Having more fun instead
The Fix

Google
1. Always require on-phone confirmation for app installs
2. Do not allow app activation through clicked URIs
3. Disable the remote install feature (or make it optional)
4. Perhaps look at our hiding tricks?

The user
1. Watch out for unknown app installs
2. Use a separate account for Android

The Google Authenticator user (or Azure or ...)
Use a non-android phone for your backup phone number
How Google Killed Two-Factor Authentication (and the reactions)

http://www.few.vu.nl/~vvdveen/bandroid.html

Radhesh Krishnan
Herbert Bos
Victor van der Veen
VU University Amsterdam

System and Network Security Group

Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)
Radhesh Krishnan

Google Killed Two-Factor Authentication

(and the reactions)

Easy, version-independent fix: explicit activation

Mixed reactions from Google, but we have their attention.

...but no API to read SMS messages YET.

Mobile-phone based 2FA seems doomed financial institutions will come to the same conclusion

The Media is (mostly) clueless.

Visit: w.few.vu.nl/~vvdveen/
February 2015.
security@android.com

Reported our findings via e-mail

Response on February 13:

This attack is mitigated in two ways:
1) Bouncer
2) App Activation

(we did not know about (2) at that time...)
February 2015.
MALPAY

Research project that involves the three major Dutch banks
ABN AMRO | Rabobank | ING

Demonstration by Radhesh

These guys use SMS-based 2FA!
and were not happy
March 2015.
Google

Sent a copy of our attack paper

**Response** on March 31

*The ability to launch an inactive app from the browser via an intent is not intentional.*

*We have opened an internal bug... (to be continued)*
March 2015.
NCSC

(dutch) National Cyber Security Center: 
*The central information hub and center of expertise for cyber security in the Netherlands*

Sent a copy of our attack paper

Response on April 2
*This paper is not within our RD policy*
April 2015.
NCSC Conference

Presentation and Demonstration at the NCSC One Conference by Radhesh and Herbert
June 2015.
Hitting the News. International!

Volkskrant
Telegraaf
Parool
Trouw
NOS
RTL
NU.nl
Gazet van Antwerpen
DeMorgen
...
computerworld

- Preposterous blabla about 'big Android leak'
- VU researchers and media raise FUD

FUD: Fear, Uncertainty and Doubt.
In other words: we are guilty of fearmongering

This blog attracts almost **200k visitors per month**
So we invited the author to have a chat
computerworld revisited

Still mildly negative, but:

*Google killed the added value of mobile phone two-factor authentication*
July 2015. FAQ

We setup a FAQ page, explaining
- that MitB still happens
- the purpose of 2FA
- that allowing sideloading is not required
- how hard it is to detect

- how to activate apps
- that poor design decisions are also bugs
- how this should be fixed
- what the user can do
and things cooled down
July 2015. Google

Some Google engineers have personally communicated that they agree with our assessment.

The official reply was, however:

*There was a lot of discussion about this, but in the end we decided that it's working as intended and have no plans to change the behavior.*

Moreover, we supposedly made 'misleading' statements:

1. Not all permissions are available (no system permission)
2. Not all forms of mobile-phone 2FA
3. On Android 4.4+ SMS cannot be deleted
1. A victim's PC or browser must be compromised before the attack.

2. The attacker must upload a malicious app to the victim's computer, which can exploit vulnerabilities.

3. The app must be installed, possibly by the victim themselves.

4. The app must be activated, either by the victim or by the attacker.

5. The victim must click or tap a trigger to activate the app.

6. The app was uploaded and installed, which allows the attacker to gain control.

7. The attacker can then manipulate the victim's PC or browser.

"Sufficient security barriers"
"sufficient security barriers"

5. The victim must be using an SMS-based 2FA mechanism for their bank and not an app or hardware-token based mechanism
   - Does this mean SMS-based 2FA is obsolete? Better call your bank to switch!
   - The fallback for those apps (Google/Azure Authenticator) is ... SMS

6. The victim will still immediately see that an SMS-based TAN has been received and can contact their bank
Almost 40% of the Android users is still at Android < 4.4.
In addition:
   - Do you call your bank if you receive a weird TAN?
   - Do you check for incoming TAN codes at 3 in the morning?
   - Does your mom?
Mixed reactions from Google, but we have their attention

iOS and Windows Phone have similar remote install features

...but no API to read SMS messages YET

Easy, version-independent fix: explicit activation

Mobile-phone based 2FA seems doomed

financial institutions will come to the same conclusion

The Media is (mostly) clueless
How Google Killed Two-Factor Authentication (and the reactions)

[Link to the full article]

Mixed reactions from Google, but we have their attention
iOS and Windows Phone have similar remote install features YET
Easy, version-independent fix: explicit activation

Mobile phone based 2FA seems doomed

The Media is (mostly) clueless

Radhesh Krishnan
Herbert Bos
Victor van der Veen
VU University Amsterdam
System and Network Security Group

Andrubis | TraceDroid (app analysis)
PathArmor @ CCS '15 (context-sensitive CFI)