Finding Backdoors in Applications

Presented by Loke Hui Yi & Max Chee
GovTech Singapore
OUTLINE

• About Us
• What We Do
• Our Modus Operandi
• Developer Backdoors – Windows Application
• Developer Backdoors – iOS Application
• Questions?
ABOUT US

• Government Technology Agency of Singapore (GovTech) is the Government Sector Lead driving technology innovation within the Singapore government

• Implementor of Singapore’s Smart Nation vision

• The Cyber Security Group in GovTech is the Cyber Sector Lead for the Singapore Government, looking into public sector resilience and ecosystem defence
ABOUT US – COMMUNITY CONTRIBUTIONS

**VLC Medio Player MKV Use After Free**

<table>
<thead>
<tr>
<th>Disclosed</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>05/24/2018</td>
<td>03/19/2019</td>
</tr>
</tbody>
</table>

**Author(s)**
- Eugene Ng - GovTech
- Winston Ho - GovTech

---

**Hall of Fame**

Thanks to the following researchers for reporting important security issues:

- Mohamed Ouad
- Amal Jacob
- Harie_cool
- proAbiral
- Abhaychandra_Chede
- Kazan71p
- Shahzad_Sadiq
- Loke Hui Yi (GovTech)
- Khor Teck Chung (GovTech)
- Jason Chan (GovTech)

---

**Cross-site Scripting (XSS)**

Affecting `node-red-dashboard` package, versions <2.17.0

**Credit**
- Goh Jing Loon (GovTech)

**CVE**
- CVE-2019-10756
- CWE-79

---

**Add CVE-2020-3452 detector plugin #10**

- Merged
- copybara-service merged 1 commit into google-master from GovTech-CSG-master 22 days ago

- Conversation: 16
- Commits: 1
- Checks: 0
- Files changed: 12

**GovTech-CSG** commented 29 days ago

Greetings! This is the first contribution from Cyber Security Group, GovTech Singapore. We are excited to develop plugins for the Tsunami Security Scanner.

The first plugin detects CVE-2020-3452, a directory traversal attack on the web services interface of Cisco Adaptive Security Appliance (ASA) Software and Cisco Firepower Threat Defense (FTD) Software. It has been exploited widely since the public release of the POC. We hope this plugin helps defenders quickly detect and patch this vulnerability.

We have followed the folder and naming conventions of Google’s custom plugins, creating a separate `govtech` folder to contain this and hopefully future contributions, similar to how Google’s plugins are contained in a separate `google` folder.

The detector itself tests for the existence of a non-sensitive known file `browser_inc_lua` based on the public POC and does not attempt further exploitation.

We have signed the Google CLA.
Manuka: A modular, scalable OSINT honeypot targeting pre-attack reconnaissance techniques

Eugene Lim
Kee Hock Tan
Bernard Lim
Kenneth Tan

Date: Thursday, August 6 | 10:00am–11:00am
Track: OSINT – Open Source Intelligence
Session Type: Arsenal

Manuka is an Open-source intelligence (OSINT) honeypot that monitors reconnaissance attempts by threat actors and generates actionable intelligence for Blue Teams. It creates a simulated environment consisting of staged OSINT sources, such as social media profiles and leaked credentials, and tracks signs of adversary interest, closely aligning to MITRE’s PRE-ATT&CK framework. Manuka gives Blue Teams additional visibility of the pre-attack reconnaissance phase and generates early-warning signals for defenders.

https://github.com/spaceraccoon/manuka
WHAT WE DO

Adversary Simulations

Source Code Review

Mobile / IoT / Web / Social Engineering PT

Product Security Assessment

Infrastructure Vulnerability Assessments

Bug Bounties

What We Do
OUR MODUS OPERANDI

• No fixed methodology as the applications we assess can vary widely

- Interesting strings in the program
- Light fuzzing
- Inline functions
- Code coverage
Example of using fuzzing on another product: Generate all permutations of HTML tags and all possible attributes

- The target software should cleanse all JavaScript from the input source
DEVELOPER BACKDOORS –
WINDOWS APPLICATION

Initial Goal

• We audit the software and environments used by government agencies to ensure that they are secure

Application Environment

• Application is a database driven COTs product requiring users to log in
• Regular users use the application’s GUI to do work. A regular user cannot do admin operations such as changing other users’ passwords nor seeing their login sessions.
Look for interesting strings in the program via IDA

• This string was not encountered in normal operation, and seemed to suggest a potential elevation of permissions

• We traced the function calls that used this string and found that it was being called when the program was starting up
DEVELOPER BACKDOORS – WINDOWS APPLICATION

Look at the startup flow for anything interesting
DEVELOPER BACKDOORS – WINDOWS APPLICATION

• Turns out that the function using our interesting string checking if certain keys were pressed in order to determine if it should appear

• Pressing the right combination of keys on application startup presented us with this dialog

• Paydirt!
DEVELOPER BACKDOORS – WINDOWS APPLICATION

• But what about the password?

Force password method to always return true using WinDBG

Figure out the password
DEVELOPER BACKDOORS – WINDOWS APPLICATION

What we knew
• The password is not hardcoded

• Probably not stored in the database as the prompt appears before the user signs in
After some digging...

- The application loads a function (generateSuperSecretPassword) in another DLL to generate the password on the fly, passing in a hardcoded seed value.

- It then calls another function (callChkPassWord) in the same DLL to prompt and compare the passwords.
DEVELOPER BACKDOORS – WINDOWS APPLICATION

• Recipe for super secret password generation
• Ingredients:
  • Seed value
  • Today’s date time object
• Method:
  • /, %, <<, >>, ^, &
  • Mix well and prepare to serve
• Return: 6 char string that changes every week
Since the password didn’t change that often, we pulled it out from RCX
DEVELOPER BACKDOORS – WINDOWS APPLICATION

• We are greeted with a console once we’ve entered the password
DEVELOPER BACKDOORS – WINDOWS APPLICATION

• Hey! We can run stored DB procedures!

• Here’s a list of login sessions for users, something a normal user should not have access to
DEVELOPER BACKDOORS –
WINDOWS APPLICATION

Impact
• On elevating a user by entering the password, the user can run database queries (stored procedures or raw SQL) to do activities such as changing passwords of other users, checking log in sessions and potentially viewing other sensitive data

Why
• The developers put it in to ease their remote maintenance operations

Mitigation?
• The password changes every week. However the seed was the same, so anyone with an installation of the program could generate the same password to be used on other machines
DEVELOPER BACKDOORS – WINDOWS APPLICATION

Challenges
• Restrictive testing environment
• Short testing time
• Communication with the developers

Tools Used
• IDA
• WinDBG

Key Takeaway
• The application was not built with security in mind, hence it was easy to RE and figure out their secret password
• Applications should enforce security by design instead, especially those dealing with sensitive data, and not merely assume their environment to be secure
• Handover to Max

• Transition slide, to be edited out
Initial Goal

- To demonstrate the impact of local authentication bypass

Why?

- Password managers hold tasty loot!!!
- Local authentication for master passwords to unlock
- Verification checks for master password can be as simple as true/false
I MEAN... JUST LOOK AT THIS???

• Verification of master password is done by just comparing if string is equal.
• Input password is not used to decrypt the vault

Assembly Code for Password Comparison

| LDR          | X25, [X8,#selRef_synchronize@PAGEOFF] |
| MOV          | X0, X22 ; void *                     |
| MOV          | X1, X25 ; char *                     |
| BL           | _objc_msgSend                         |
| ADRP         | X8, #selRef_stringForKey@PAGE         |
| LDR          | X1, [X8,#selRef_stringForKey@PAGEOFF] ; char * |
| ADRP         | X2, #cfstr_Kloginpassword@PAGE        ; "KloginPassword" |
| ADD          | X2, X2, #cfstr_Kloginpassword@PAGEOFF ; "KloginPassword" |
| MOV          | X0, X22 ; void *                     |
| BL           | _objc_msgSend                         |
| MOV          | X29, X29                              |
| BL           | _objc_retainAutoreleasedReturnValue   |
| MOV          | X23, X0                               |
| ADRP         | X8, #selRef_isEqualToString@PAGE      |
| LDR          | X26, [X8,#selRef_isEqualToString@PAGEOFF] |
| MOV          | X1, X26 ; char *                      |
| MOV          | X2, X20                               |
| BL           | _objc_msgSend                         |
| CBZ          | W0, loc 10007F244                     |
DEVELOPER BACKDOORS – IOS APPLICATION

• So… should you entrust your passwords to random free password managers found online?

• As a learning experience, we selected “Lock Password Manager” to conduct our tests. It can be downloaded from the AppStore, averaging 4 of 5 stars with over 500 ratings.
DEVELOPER BACKDOORS – IOS APPLICATION

• Set a master password for Password Manager

• Didn’t set password hint. Who needs one?
DEVELOPER BACKDOORS – IOS APPLICATION

• Password manager ask to backup master password

• Complains forgotten master password will lead to “no way of accessing the app”

• “There are no recovery passcode”
DEVELOPER BACKDOORS – IOS APPLICATION

• Hooking framework “THEOS” used to observe application flow for password entry

Explaination for Theos Logos

%hook – Opens a hook block for the specified class
%log - Prints the method arguments in system log
%orig - Calls the original hooked method
%end - Close the hook block

Tweak to observe application behaviour

%hook NBAppDelegate
- (unsigned char)isUseSecureClearData { %log; unsigned char r = %orig; HBLogDebug("isUseSecureClearData return value: = %hhu", r); return r; }
- (unsigned char)PAPasscodeViewControllerDidEnterPasscode:(id)arg1 enterPasscode:(id)arg2 { %log; unsigned char r = %orig;
  HBLogDebug("PAPasscodeViewControllerDidEnterPasscodereturn value: = %hhu", r); return r; }
- (void)PAPasscodeViewControllerDidEnterPasscode:(id)arg1 { %log; %orig; }
- (void)PAPasscodeViewController:(id)arg1 didFailToEnterPasscode:(long long)arg2 { %log; %orig; }
- (void)IsSuccessLogin:(unsigned char)arg1 { %log; %orig; }
- (long long)genErrorLoginTimes { %log; long long r = %orig; HBLogDebug(" = %lld", r); return r; }
- (void) setDateFlag { %log; %orig; }
%end
Observe application flow for right/wrong password entry

Application logic/call stack for right password entry

```
<NBApDelegate: 0x2813e1a80> PAPasscodeViewControllerDidEnterPasscode:<PAPasscodeViewController: 0x10284a000> enterPasscode:123456
<NBApDelegate: 0x2813e1a80> setDateFlag
<NBApDelegate: 0x2813e1a80> isUseSecureClearData
isUseSecureClearData return value: = 0
<NBApDelegate: 0x2813e1a80> IsSuccessLogin:1

Application logic/call stack for wrong password entry

```

```
<NBApDelegate: 0x2812eefc0> PAPasscodeViewControllerDidEnterPasscode:<PAPasscodeViewController: 0x106148a00> enterPasscode:112233
<PAPasscodeViewController: 0x106148a00> didFailToEnterPasscode:2
<NBApDelegate: 0x2812eefc0> IsSuccessLogin:0

Application logic/call stack for wrong password entry

```

```
<NBApDelegate: 0x2812eefc0> PAPasscodeViewController: <PAPasscodeViewController: 0x106148a00> didFailToEnterPasscode:2
<NBApDelegate: 0x2812eefc0> IsSuccessLogin:0
<NBApDelegate: 0x2812eefc0> isUseSecureClearData
isUseSecureClearData return value: = 0
```
Observe application flow for right/wrong password entry

### Application logic/call stack for right password entry

```
<NBAppDelegate: 0x2813e1a80> PAPasscodeViewControllerDidEnterPasscode:<PAPasscodeViewController: 0x10284a000> enterPasscode:123456
<NBAppDelegate: 0x2813e1a80> setDateFlag
<NBAppDelegate: 0x2813e1a80> isUseSecureClearData
[isUseSecureClearData return value: = 0]
<NBAppDelegate: 0x2813e1a80> IsSuccessLogin:1
<NBAppDelegate: 0x2813e1a80> notificationLoadAds
[PasscodeViewController return value: = 1]
```

### Application logic/call stack for wrong password entry

```
<NBAppDelegate: 0x2812eefc0> PAPasscodeViewControllerDidEnterPasscode:<PAPasscodeViewController: 0x106148a00> enterPasscode:112233
[NBAppDelegate: 0x2812eefc0] PAPasscodeViewController:<PAPasscodeViewController: 0x106148a00> didFailToEnterPasscode:2
[NBAppDelegate: 0x2812eefc0] IsSuccessLogin:0
[NBAppDelegate: 0x2812eefc0] isUseSecureClearData
[isUseSecureClearData return value: = 0]
```
• Compare the input password against the master password set
  • Returns true (value 1), else return false (value 0)
DEVELOPER BACKDOORS – IOS APPLICATION

• On right password entry,
  • Calls `[NBApplDelegate setDateFlag]`
  • Calls `[NBApplDelegate isSuccessLogin]` passing 1 as argument

```assembly
ADRP   X8, #selRef_setDateFlag@PAGE
LDR    X1, [X8,#selRef_setDateFlag@PAGEOFF] ; char *
MOV    X0, X21 ; void *
BL     _objc_msgSend ; -[NBApplDelegate setDateFlag]
ADRP   X8, #selRef_IsSuccessLogin@PAGE
LDR    X1, [X8,#selRef_IsSuccessLogin@PAGEOFF] ; char *
MOV    W24, #1
MOV    W2, #1
MOV    X0, X21 ; void *
BL     _objc_msgSend ; -[NBApplDelegate IsSuccessLogin]
ADRP   X8, #selRef_dismissViewControllerAnimated_completion@PAGE
LDR    X1, [X8,#selRef_dismissViewControllerAnimated_completion@PAGEOFF]
B      loc 10007F2AC
```
DEVELOPER BACKDOORS – IOS APPLICATION

So far so good…

• When the right master password is entered, IsSuccessLogin will be set to true, allowing the application to display saved passwords

• In the same vein, if the wrong master password is entered, the application should just prompt an error

• However…
• On wrong password entry,
  • Checks if user input password was equal to *#06##*

```
[Code Snippet]
```

**Assembly Code for Wrong Password**

```
loc_10007F244
ADR
ADD
MOV
MOV
BL
BL
JMP
```

**LOCATIONS**
- loc_10007F244: "*#06##"
- loc_10007F2D0
DEVELOPER BACKDOORS – IOS APPLICATION

• Secret backdoor in password manager
  • User input of *#06#* will write false to key-pair kpasswordon
  • Removes the need for master password to unlock
• Tracing through important key-pair value in the NSUserDefaults,
  • kpasswordon determines if master password is needed to access app
  • kloginpassword stores the originally set master password
  • Both are located in group.com.estill.passwordmgr.plist

```swift
v10 = objc_msgSend(v8, "systemVersion");
v11 = (void *)objc_retainAutoreleasedReturnValue(v10);
objc_msgSend(v11, "floatValue");
v13 = v12;
objc_release(v11);
objc_release(v9);
if (v13 >= 7.0 )
{
  v16 = objc_msgSend(&OBJC_CLASS__NSUserDefaults, "alloc");
  v15 = objc_msgSend(v16, "initWithSuiteName", CFSTR("group.com.estill.passwordmgr"));
}
else
{
  v14 = objc_msgSend(&OBJC_CLASS__NSUserDefaults, "standardUserDefaults");
  v15 = (void *)objc_retainAutoreleasedReturnValue(v14);
}
v17 = v15;
objc_msgSend(v15, "setBool:forKey:", 0LL, CFSTR("KTouchILauch"));
objc_msgSend(v17, "synchronize");
v56 = v17;
v18 = objc_msgSend(v17, "stringForKey:", CFSTR("kloginPassword"));
v19 = objc_retainAutoreleasedReturnValue(v18);
v20 = v59;
```
DEVELOPER BACKDOORS – IOS APPLICATION

• Extracted group.com.estill.passwordmgr.plist from iphone
  • Found unencrypted master password in plist!!!
DEVELOPER BACKDOORS – IOS APPLICATION

- So much for telling users to backup their master password

"NO RECOVERY PASSCODE"
DEVELOPER BACKDOORS – IOS APPLICATION

- Total of 4 apps were found, 3 free and 1 paid version
- Total of 3 password managers created by this developer
DEVELOPER BACKDOORS – IOS APPLICATION

- Tried the other app (Password Manager Finger Print Lock for iPhone Safe)
- Same interface as the original app
- Has the same backdoor *#06#*
- High chance that paid password manager “Password Manager Pro Keep passcode.s safe & lock private account.s” also have the same backdoor!!!
DEVELOPER BACKDOORS – IOS APPLICATION

**Impact**

- Someone with physical access to your phone could use the backdoor and get all your sweet, sweet passwords

**Why**

- Who knows? Backdoor recovery code?

**Mitigation?**

- Reported to developer multiple times since 2 Jan 2020. Since there was no reply from developer, I also tried to reach out to Apple. To date, the vulnerable app is still available on the AppStore.
INGOTS OF HARD WORK
FOLLOW OUR DEVELOPMENT!

GovTech CSG Medium Page
https://medium.com/csg-govtech
SNEAK PREVIEW!
CAPTURE THE FLAG (CTF) COMPETITION

• GovTech CSG will be organizing our first inaugural CTF competition!
  o Why? We want to create something fun for the community!

• There will be a range of attractive prizes to be won, not just for the top teams!

• It will be a co-event with Stack 2020
  o By the way, Stack 2020 is a free virtual event this year!

• More details will be released by us soon!
QUESTIONS?

Feel free to reach us at

LinkedIn: angelystor & max-chee

Twitter: @angelystor