



OCTOBER 1-2, 2020
BRIEFINGS

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



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 Media Player MKV Use After Free

Disclosed	Created
05/24/2018	03/19/2019
Author(s)	
Eugene Ng - GovTech Winston Ho - GovTech	



WalkMe DAP

Enterprise System

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

[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 18  Commits 1  Checks 0  Files changed 12



GovTech-CSG commented 29 days ago

Contributor

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.

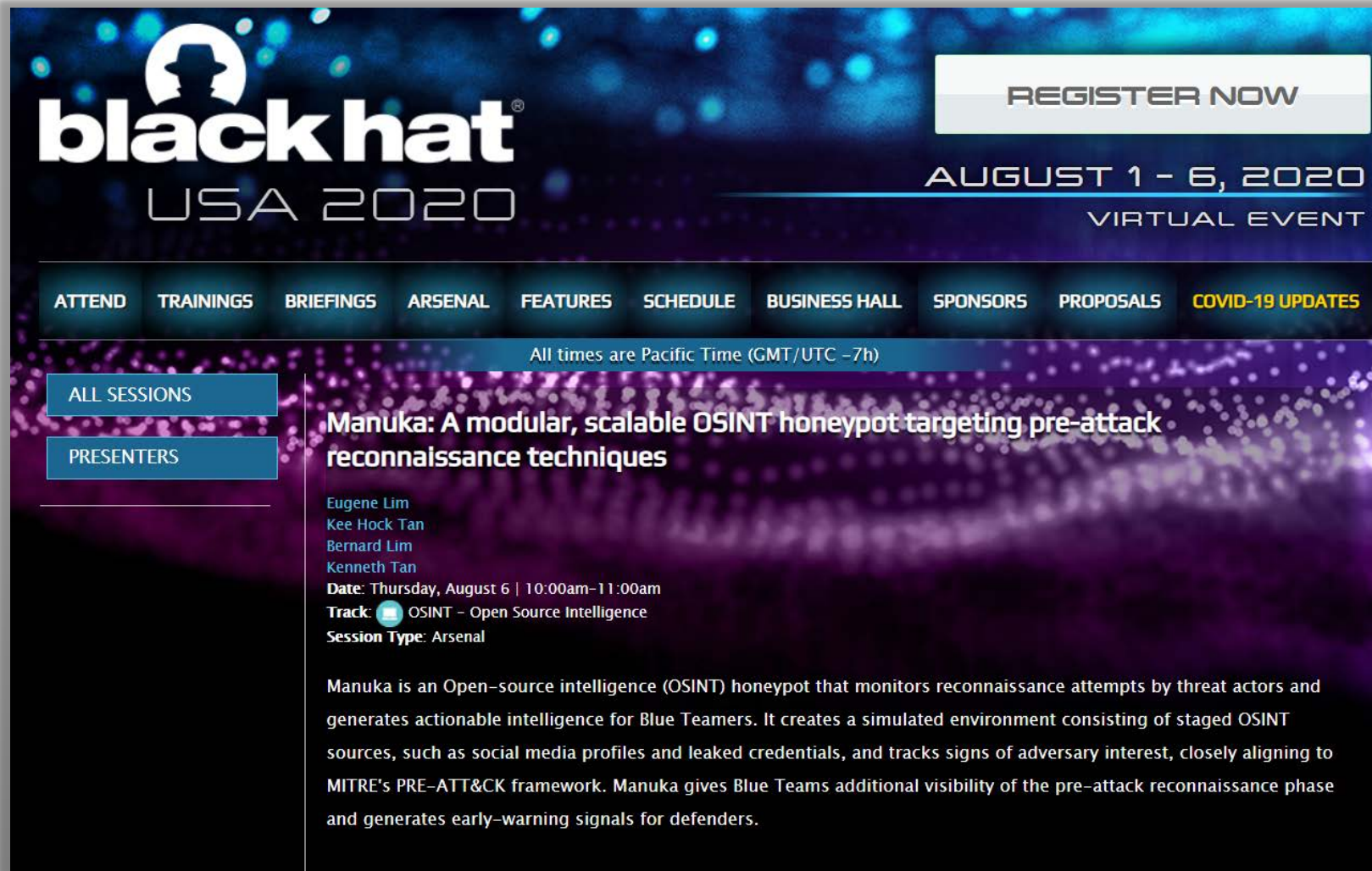
The detector itself tests for the existence of a non-sensitive known file `browser_inc.lua` based on the public POC and does not attempt further exploitation.

We have signed the Google CLA.



CYBER SECURITY
GROUP

ABOUT US – COMMUNITY CONTRIBUTIONS



black hat
USA 2020

REGISTER NOW

AUGUST 1 - 6, 2020
VIRTUAL EVENT

ATTEND TRAININGS BRIEFINGS ARSENAL FEATURES SCHEDULE BUSINESS HALL SPONSORS PROPOSALS COVID-19 UPDATES

All times are Pacific Time (GMT/UTC -7h)

ALL SESSIONS
PRESENTERS

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 Teamers. 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>

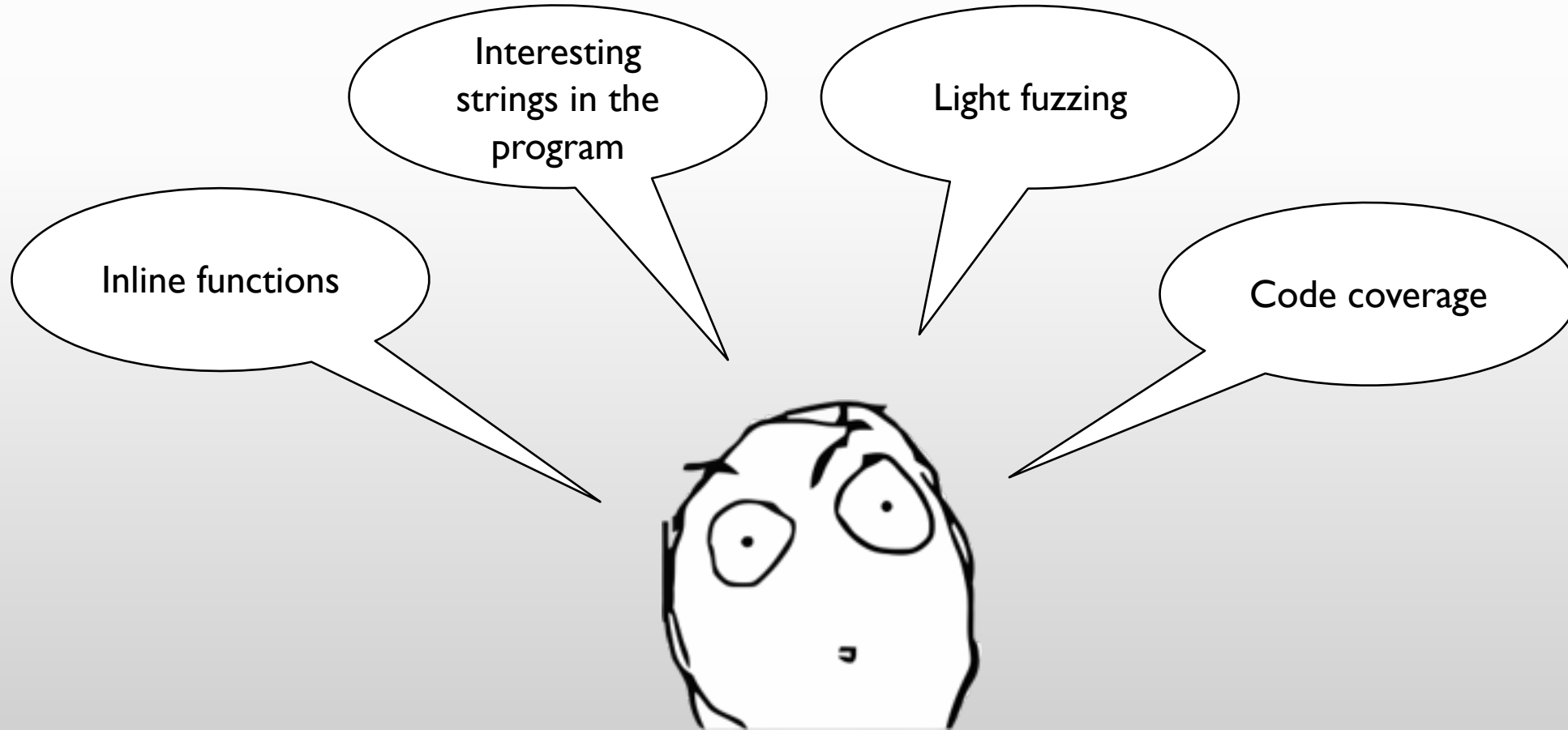


CYBER SECURITY
GROUP



OUR MODUS OPERANDI

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



OUR MODUS OPERANDI - IN ACTION

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

```
em onafterprint
em onbeforeprint
em onerror
em onhashchange
em onload
em onmessage
em onoffline
em ononline
em onpagehide
em onpageshow
em onpopstate
em onresize
em onstorage
em onunload
em onblur
em onchange
em oncontextmenu
em onfocus
em oninput
em oninvalid
...
<dt onvolumechange="console.log("dt:onvolumechange")">dt onvolumechange</dt>
<br>
<dt onwaiting="console.log("dt:onwaiting")">dt onwaiting</dt>
<br>
<dt ontoggle="console.log("dt:ontoggle")">dt ontoggle</dt>
<br>
<em onafterprint="console.log("em:onafterprint")">em onafterprint</em>
<br>
<em onbeforeprint="console.log("em:onbeforeprint")">em onbeforeprint</em>
<br>
<em onerror="console.log("em:onerror")">em onerror</em>
<br>
<em onhashchange="console.log("em:onhashchange")">em onhashchange</em>
<br>
<em onload="console.log("em:onload")">em onload</em>
<br>
... <em onmessage="console.log("em:onmessage")">em onmessage</em> == $0
<br>
<em onoffline="console.log("em:onoffline")">em onoffline</em>
<br>
<em ononline="console.log("em:ononline")">em ononline</em>
<br>
<em onpagehide="console.log("em:onpagehide")">em onpagehide</em>
<br>
```

Original

```
em onafterprint
em onbeforeprint
em onerror
em onhashchange
em onload
em onmessage
em onoffline
em ononline
em onpagehide
em onpageshow
em onpopstate
em onresize
em onstorage
em onunload
em onblur
em onchange
em oncontextmenu
em onfocus
...
<dt>dt onttimeupdate</dt>
<br>
<dt>dt onvolumechange</dt>
<br>
<dt>dt onwaiting</dt>
<br>
<dt>dt ontoggle</dt>
<br>
<em>em onafterprint</em>
<br>
<em>em onbeforeprint</em>
<br>
<em>em onerror</em>
<br>
<em>em onhashchange</em>
... <br> == $0
<em>em onload</em>
<br>
<em>em onmessage</em>
```

Cleaned

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.



DEVELOPER BACKDOORS – WINDOWS APPLICATION

Look for interesting strings in the program via IDA



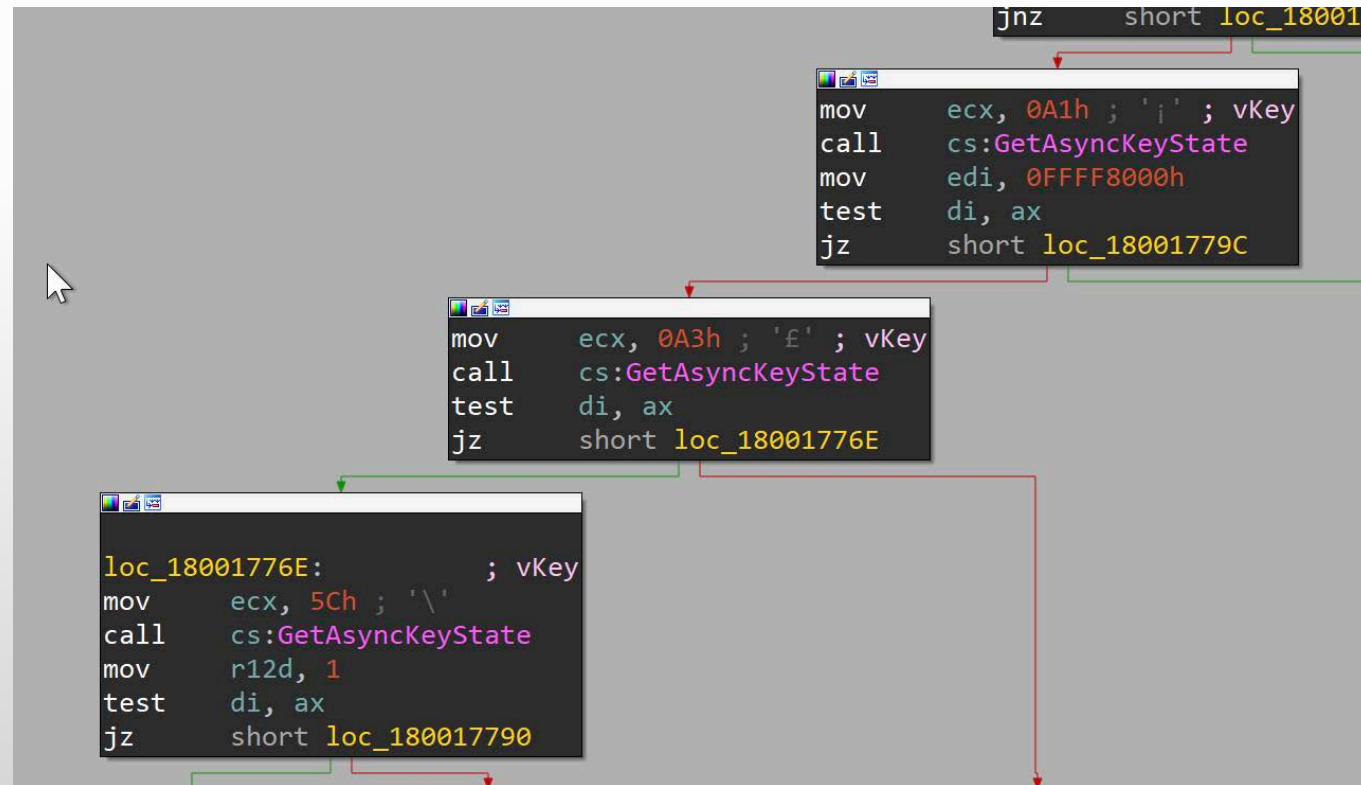
.rdata:0000000...	0000000B	C	[REDACTED]
.rdata:0000000...	00000024	C	Enter password for [REDACTED]

- 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



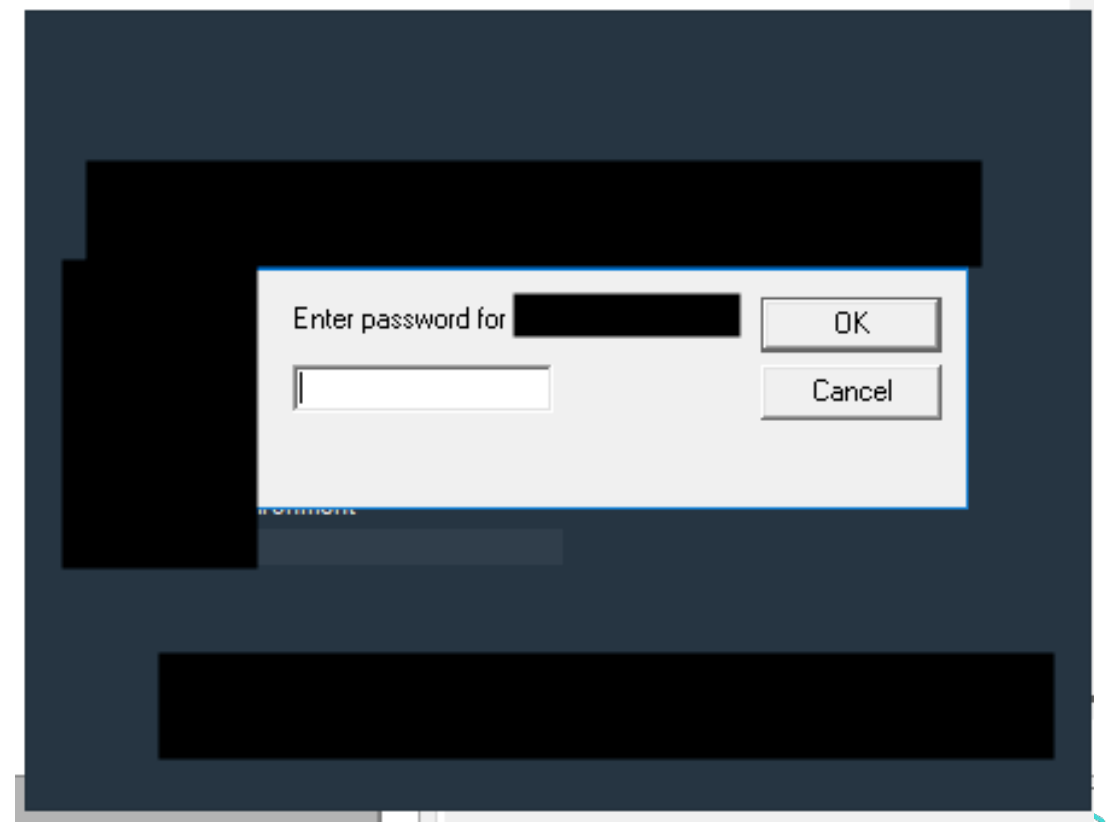
DEVELOPER BACKDOORS – WINDOWS APPLICATION

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



DEVELOPER BACKDOORS – WINDOWS APPLICATION

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

```
loc_18003582C:
lea     r8, seed          ; [REDACTED]
lea     rcx, [rsp+428h+Destination] ; Destination
mov     edx, 3E8h         ; SizeInBytes
call    strcpy_s
lea     rdx, [rsp+428h+Destination]
xor     ecx, ecx
call    FUNC_generateSuperSecretPassword
lea     rdx, aEnterPasswordF ; "Enter password fo
lea     rcx, [rsp+428h+Destination]
call    FUNC_callChkPassWord
xor     ecx, ecx
test    eax, eax
```



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

```
1 const char *__fastcall generateSecretPassword(const char *a1, int a2)
2 {
3     const char *v2; // rbp
4     int v3; // ebx
5     struct _FILETIME v4; // r11
6     __int64 v6; // rax
7     struct _FILETIME SystemTimeAsFileTime; // [rsp+20h] [rbp-28h]
8     struct _SYSTEMTIME SystemTime; // [rsp+28h] [rbp-20h]
9
10    v2 = a1;
11    v3 = a2;
12    GetSystemTimeAsFileTime(&SystemTimeAsFileTime);
13    v4 = SystemTimeAsFileTime;
14    if ( strcmp("██████", v2) )
15        *v2 = 0;
16    if ( v3 == -1 )
17    {
18        v6 = -604800000000i64;
19 LABEL_9:
20        *(_QWORD *)&v4 += v6;
21        goto LABEL_10;
22    }
23    if ( v3 )
24    {
25        if ( v3 != 1 )
26            return v2;
27        v6 = 604800000000i64;
28        goto LABEL_9;
29    }
30 LABEL_10:
31    SystemTimeAsFileTime = v4;
32    FileTimeToSystemTime(&SystemTimeAsFileTime, &SystemTime);
33    return sub_180008750(v2, SystemTime.wDay + 100 * (SystemTime.wMonth + 100 * (unsigned int)SystemTime.wYear));
```

DEVELOPER BACKDOORS – WINDOWS APPLICATION

- Since the password didn't change that often, we pulled it out from RCX

WinDbg:10.0.10240.9 AMD64

```

00000000`0014c670  00000000 00000000 00000000 00000000  .....
00000000`0014c680  00000000 00000000 00000000 00000000  .....
00000000`0014c690  bfa465c5 0000507c 00000000 00000000  .e..|P.....
0:000> g

```

```

                                mov     rbp,rcx
0:000> dc rcx
00000000`0014c620  706e6a78 00006f53 00000000 00000000  xjnpSo.....
00000000`0014c630  00000000 00000000 00000000 00000000  .....
00000000`0014c640  00000000 00000000 00000000 00000000  .....
00000000`0014c650  00000000 00000000 00000000 00000000  .....
00000000`0014c660  00000000 00000000 00000000 00000000  .....
00000000`0014c670  00000000 00000000 00000000 00000000  .....
00000000`0014c680  00000000 00000000 00000000 00000000  .....
00000000`0014c690  bfa465c5 0000507c 00000000 00000000  .e..|P.....
0:000> g
ModLoad: 00007ffc`53490000 00007ffc`53526000  C:\WINDOWS\System32\TextInputFramework.dll
ModLoad: 00007ffc`54b00000 00007ffc`54e1e000  C:\WINDOWS\System32\CoreUIComponents.dll
ModLoad: 00007ffc`613a0000 00007ffc`6147a000  C:\WINDOWS\System32\CoreMessaging.dll
ModLoad: 00007ffc`63200000 00007ffc`63231000  C:\WINDOWS\SYSTEM32\ntmarta.dll
ModLoad: 00007ffc`5ff00000 00007ffc`6004d000  C:\WINDOWS\SYSTEM32\wintypes.dll
ModLoad: 00000000`04140000 00000000`0428d000  C:\WINDOWS\SYSTEM32\wintypes.dll
ModLoad: 00007ffc`63af0000 00007ffc`63b56000  C:\WINDOWS\system32\mswsock.dll

```

BUSY Debuggee is running...

DEVELOPER BACKDOORS – WINDOWS APPLICATION

- We are greeted with a console once we've entered the password

Dyalog APL/W-64 Version 16.0.33583

Unicode Edition

Tue Nov 5 15:57:24 2019

ws saved Wed Aug 23 21:57:06 2017

```

1+1
2
VALUE ERROR
seFnsConfigure[3] :GoTo End×\ #.xsErr xs→0 #.progObjLoad #.mSE_CNF

```

Ready...

The screenshot shows a SQL query execution window. The query is:

```
dbSelect tUSERLOGONS
```

The results are displayed in a table with the following columns:

- ID
- USERNAME
- LOGON_TIMESTAMP

The table contains 16 rows of data. The first row is highlighted in blue. The window has a menu bar (File, Edit, View, Windows, Session, Log, Action, Options, Tools, Help) and a toolbar with various icons. The status bar at the bottom shows "Ready..." and "SI:13".

ID	USERNAME	LOGON_TIMESTAMP
1	345	26 1 0
1	990	25 1 0
1	2100	26 1 0
1	277	26 1 0
1	2094	26 1 0
...		
0	3571	20190130.681597 0
		20190201.530093 136
		20190212.586817 20190212.683229 118
		20190212.68978 20190212.697002 121
		20190212.705313 20190212.706632 122
		20190212.831111 20190212.904549 127
		20190220.379294 0 300
		20190311.417269 20190311.54485 515
		20190311.649236 20190311.650394 572
		20190311.712002 20190311.773565 578
		20190311.774329 20190311.822211 582
		20190322.476551 20190322.590984 177
		20190326.379363 20190326.439259 1079
		20190403.351296 20190403.388252 11325
		20190403.406192 20190403.439711 11326
		20190417.362813 20190417.397535 54
		20190417.402836 20190417.437014 56
		20190418.394063 20190418.421366 54
		20190418.675046 20190418.707222 102
		20190422.46875 20190422.554444 145
		20190503.45478 20190503.489757 2
		20190515.641424 20190515.727535 261
		20190515.713519 20190515.804468 271
		20190607.398021 20190607.530961 406
		20190607.61419 20190607.661887 411
		20190607.666921 20190607.729988 413
		20190611.380706 20190611.382593 169
		20190611.443356 20190611.476898 178
		20190614.604815 20190614.605289 166
...		

-
- File Edit View Windows Session Log Action Options Tools Help
- WS Object Tool Font size 16
- ...
- ```
dbSelect tUSERLOGONS
```
- 0 3571 [REDACTED] 20190130.681597 0 [REDACTED] 233 [REDACTED]
- ...
- Ready...



# 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



# PLACEHOLDER

- Handover to Max
- Transition slide, to be edited out



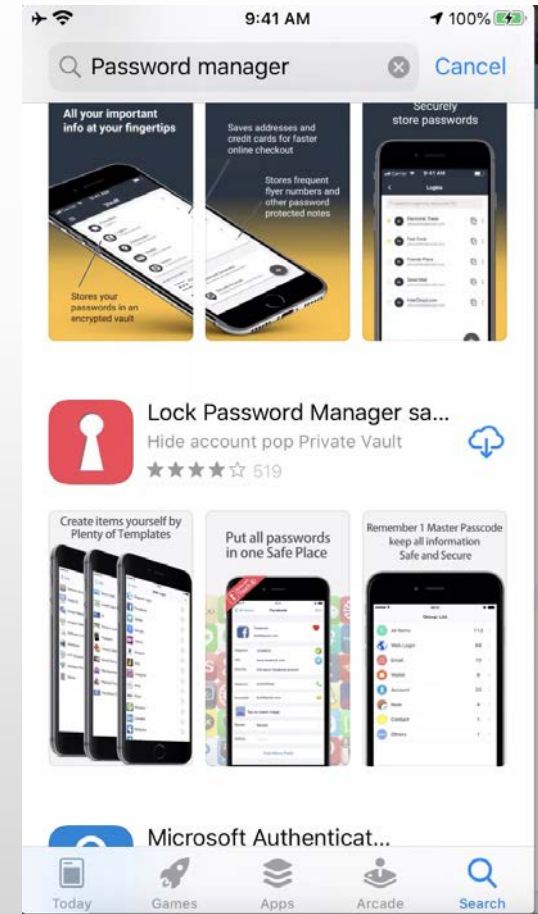
# DEVELOPER BACKDOORS – IOS APPLICATION

## 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

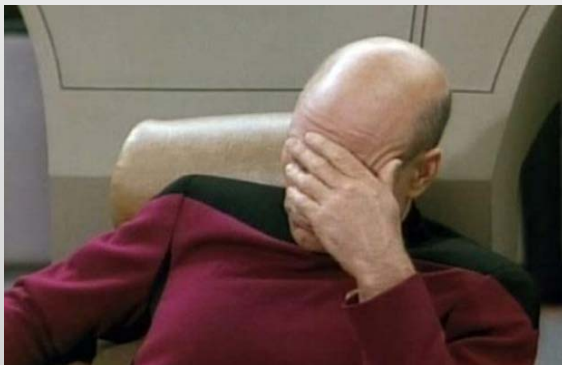




# DEVELOPER BACKDOORS – IOS APPLICATION

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
```



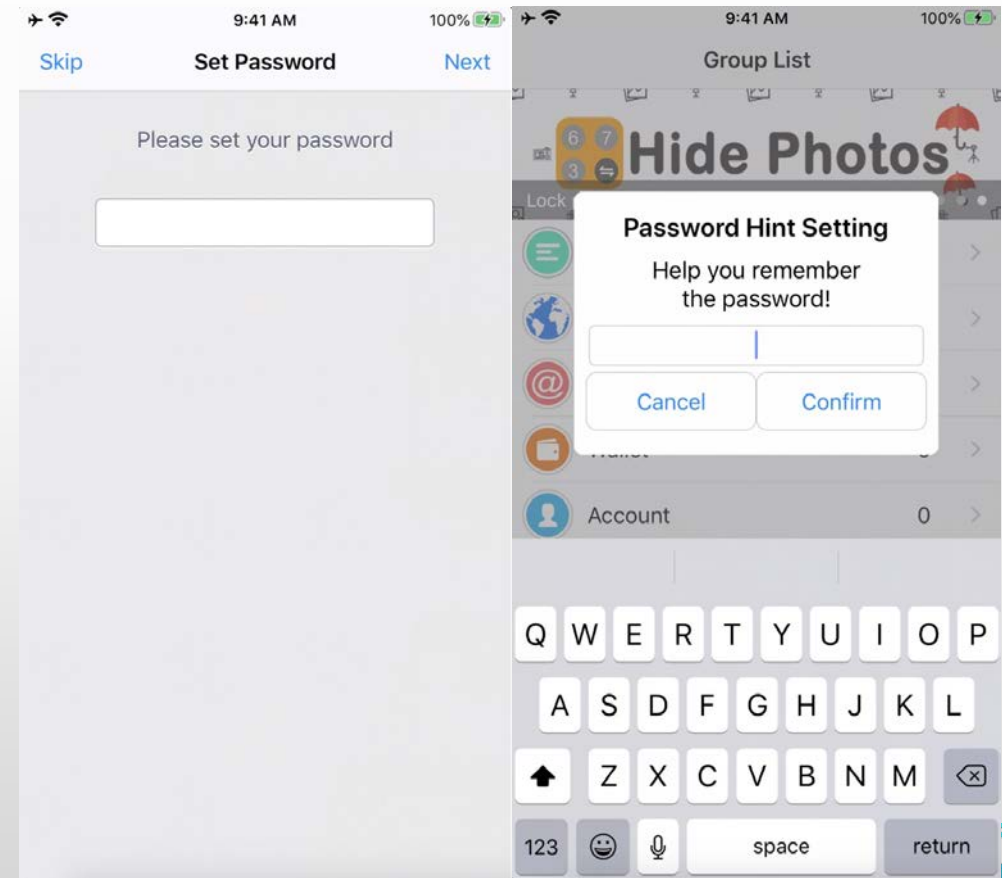
# DEVELOPER BACKDOORS – IOS APPLICATION

- 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.



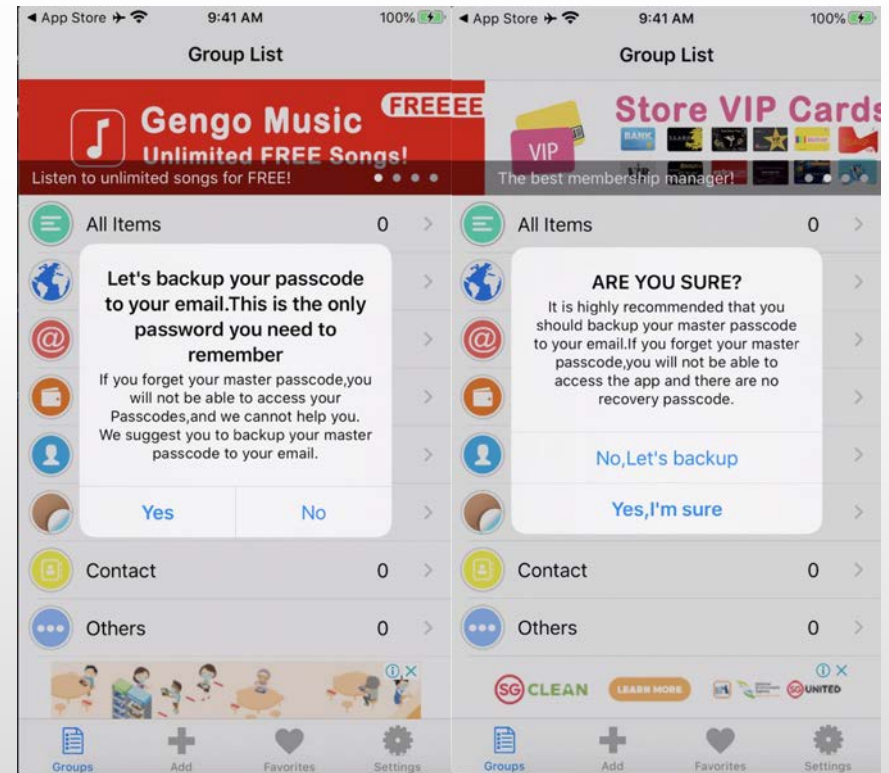
# DEVELOPER BACKDOORS – IOS APPLICATION

- 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

## Explanation 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; HLogDebug(@"isUseSecureClearData return value: = %hu", r); return r; }
- (unsigned char)PAPasscodeViewControllerDidEnterPasscode:(id)arg1 enterPasscode:(id)arg2 { %log; unsigned char r = %orig;
HLogDebug(@"PAPasscodeViewControllerDidEnterPasscode return value: = %hu", 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; HLogDebug(@" = %lld", r); return r; }
- (void)setDateFlag { %log; %orig; }
%end
```



# DEVELOPER BACKDOORS – IOS APPLICATION

## 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]
[PasscodeViewController return value: = 0]
[<NBAppDelegate: 0x2812eefc0> PAPasscodeViewController:<PAPasscodeViewController: 0x106148a00> didFailToEnterPasscode:2]
[<NBAppDelegate: 0x2812eefc0> IsSuccessLogin:0]
[<NBAppDelegate: 0x2812eefc0> isUseSecureClearData]
[isUseSecureClearData return value: = 0]
```



# DEVELOPER BACKDOORS – IOS APPLICATION

## 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]
[PasscodeViewController return value: = 0]
[<NBAppDelegate: 0x2812eefc0> PAPasscodeViewController:<PAPasscodeViewController: 0x106148a00> didFailToEnterPasscode:2]
[<NBAppDelegate: 0x2812eefc0> IsSuccessLogin:0]
[<NBAppDelegate: 0x2812eefc0> isUseSecureClearData]
[isUseSecureClearData return value: = 0]
```



# DEVELOPER BACKDOORS – IOS APPLICATION

- Compare the input password against the master password set
  - Returns true (value 1), else return false (value 0)

```
50 v20 = objc_msgSend(v19, "stringForKey:", CFSTR("kloginPassword"));
51 v21 = (void *)objc_retainAutoreleasedReturnValue(v20);
52 if ((unsigned int)objc_msgSend(v21, "isEqualToString:", v8))
53 {
54 -[NBAppDelegate setDateFlag](v5, "setDateFlag");
55 v22 = 1;
56 -[NBAppDelegate IsSuccessLogin:](v5, "IsSuccessLogin:", 1LL); |
57 }
58 else
59 {
60 if (!(unsigned int)objc_msgSend(v8, "isEqualToString:", CFSTR(".*#06#")))
61 {
62 v22 = 0;
63 goto LABEL_10;
64 }
65 objc_msgSend(v19, "setBool:forKey:", 0LL, CFSTR("kpasswordon"));
66 objc_msgSend(v19, "setObject:forKey:", &stru_1004D5820, CFSTR("kloginPassword"));
67 objc_msgSend(v19, "synchronize");
68 v22 = 1;
69 }
```





# DEVELOPER BACKDOORS – IOS APPLICATION

- On right password entry,
  - Calls [NBAppDelegate setDateFlag]
  - Calls [NBAppDelegate isSuccessLogin] passing 1 as argument

```
50 v20 = objc_msgSend(v19, "stringForKey:", CFSTR("kloginPassword"));
51 v21 = (void *)objc_retainAutoreleasedReturnValue(v20);
52 if ((unsigned int)objc_msgSend(v21, "isEqualToString:", v8))
53 {
54 -[NBAppDelegate setDateFlag](v5, "setDateFlag");
55 v22 = 1;
56 -[NBAppDelegate isSuccessLogin:](v5, "IsSuccessLogin:", 1LL); |
57 }
58 else
59 {
60 if (!(unsigned int)objc_msgSend(v
61 {
62 v22 = 0;
63 goto LABEL_10;
64 }
65 objc_msgSend(v19, "setBool:forKey:");
66 objc_msgSend(v19, "setObject:forKey:");
67 objc_msgSend(v19, "synchronize");
68 v22 = 1;
69 }
```

## Assembly Code for Correct Password

```
ADRP X8, #selRef_setDateFlag@PAGE
LDR X1, [X8,#selRef_setDateFlag@PAGEOFF] ; char *
MOV X0, X21 ; void *
BL _objc_msgSend ; -[NBAppDelegate 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 ; -[NBAppDelegate 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...



# DEVELOPER BACKDOORS – IOS APPLICATION

- On wrong password entry,
  - Checks if user input password was equal to `*#06#*`

## Assembly Code for Wrong Password

```
50 v20 = objc_msgSend(v19, "stringForKey:", CFSTR("kloginPassword"));
51 v21 = (void *)objc_retainAutoreleasedReturnValue(v20);
52 if ((unsigned int)objc_msgSend(v21, "isEqualToString:", v8))
53 {
54 -[NBAppDelegate setDateFlag](v5, "setDateFlag");
55 v22 = 1;
56 -[NBAppDelegate IsSuccessLogin:](v5, "IsSuccessLogin:", 1LL);
57 }
58 else
59 {
60 if (!(unsigned int)objc_msgSend(v8, "isEqualToString:", CFSTR(".*#06#*")))
61 {
62 v22 = 0;
63 goto LABEL_10;
64 }
65 objc_msgSend(v19, "setBool:forKey:", 0LL, CFSTR("kpasswordon"));
66 objc_msgSend(v19, "setObject:forKey:", &stru_1004D5820, CFSTR("kloginPassword"));
67 objc_msgSend(v19, "synchronize");
68 v22 = 1;
69 }
```

```
loc_10007F244 ; ".*#06#*"
ADRP X2, #cfstr_06@PAGE
ADD X2, X2, #cfstr_06@PAGEOFF ; ".*#06#*"
MOV X0, X20 ; void *
MOV X1, X26 ; char *
BL _objc_msgSend
CBZ W0, loc_10007F2D0
```

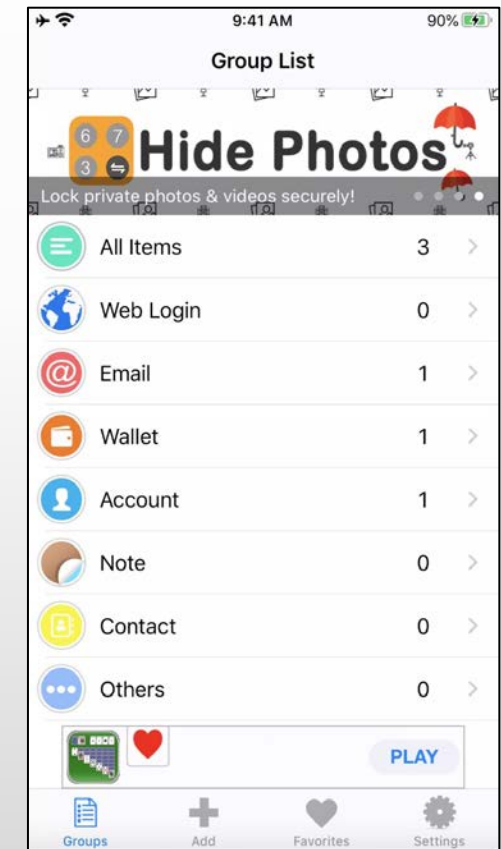
```
loc_10007F2D0
MOV W24, #0
```



# 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

```
50 v20 = objc_msgSend(v19, "stringForKey:", CFSTR("kloginPassword"));
51 v21 = (void *)objc_retainAutoreleasedReturnValue(v20);
52 if ((unsigned int)objc_msgSend(v21, "isEqualToString:", v8))
53 {
54 -[NBAppDelegate setDateFlag](v5, "setDateFlag");
55 v22 = 1;
56 -[NBAppDelegate IsSuccessLogin:](v5, "IsSuccessLogin:", 1LL); |
57 }
58 else
59 {
60 if (!(unsigned int)objc_msgSend(v8, "isEqualToString:", CFSTR("*#06#")))
61 {
62 v22 = 0;
63 goto LABEL_10;
64 }
65 objc_msgSend(v19, "setBool:forKey:", 0LL, CFSTR("kpasswordon"));
66 objc_msgSend(v19, "setObject:forKey:", &stru_1004D5820, CFSTR("kloginPassword"));
67 objc_msgSend(v19, "synchronize");
68 v22 = 1;
69 }
```





# DEVELOPER BACKDOORS – IOS APPLICATION

- 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

```
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("kTouchIDLaunch"));
objc_msgSend(v17, "synchronize");
v58 = 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!!!

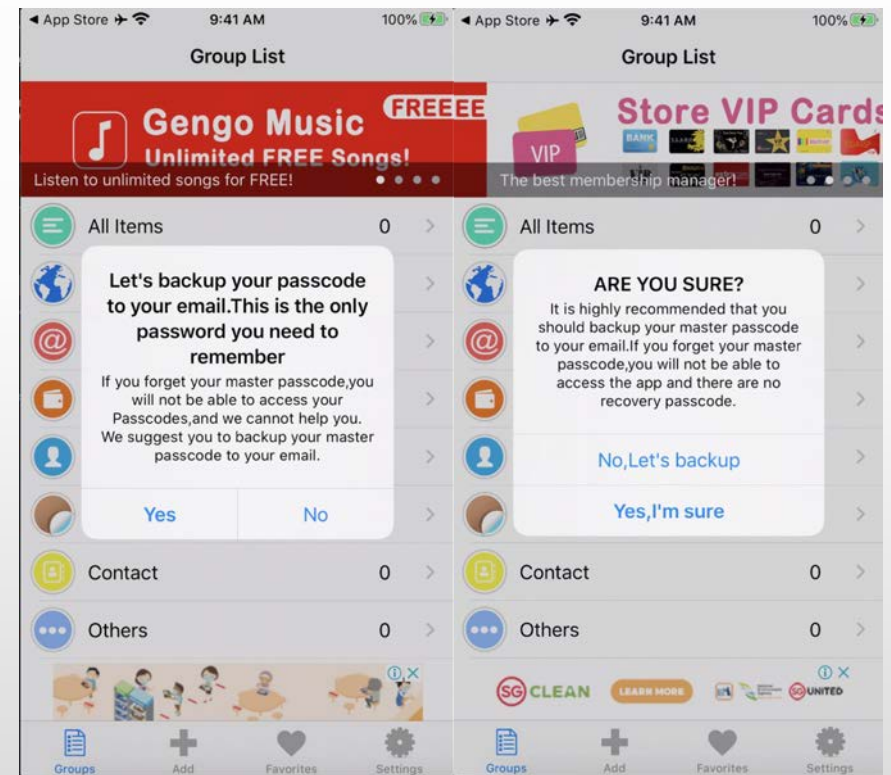
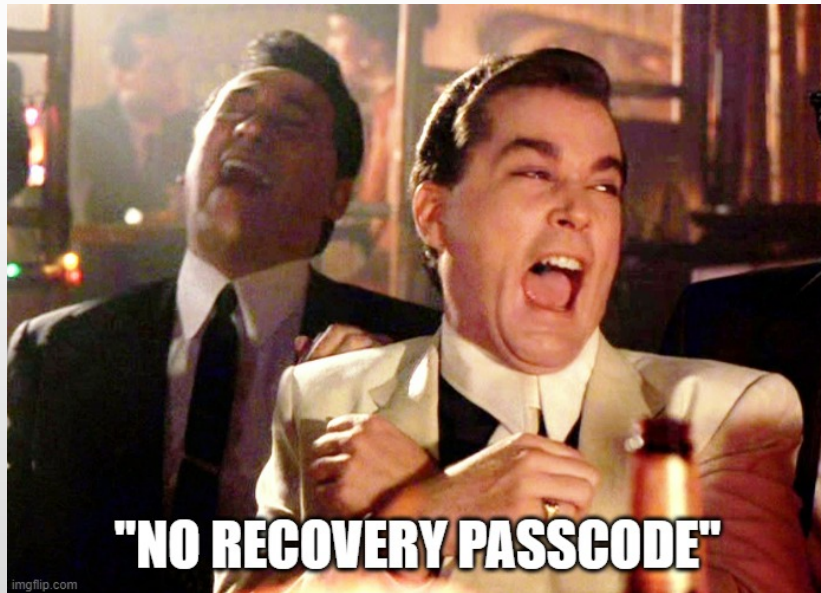
```
iPhone:/ root# find / -iname group.com.estill.passwordmgr.plist
find: './.fseventsd': Operation not permitted
find: '/private/var/.fseventsd': Operation not permitted
/private/var/mobile/Containers/Shared/AppGroup/5B449D23-B5F9-4D00-B470-B0C663DE1DD4/Library/Preferences/group.com.estill.passwordmgr.plist
iPhone:/ root#
```

| group.com.estill.passwordmgr.plist                |            |                        |
|---------------------------------------------------|------------|------------------------|
| group.com.estill.passwordmgr.plist > No Selection |            |                        |
| Key                                               | Type       | Value                  |
| ▼ Root                                            | Dictionary | (5 items)              |
| kTouchIdCancel                                    | Boolean    | NO                     |
| kTouchIDAllowed                                   | Boolean    | NO                     |
| kloginPassword                                    | String     | MySuperSecureP@sswOrd! |
| kTouchIDLauch                                     | Boolean    | NO                     |
| kpasswordon                                       | Boolean    | YES                    |



# DEVELOPER BACKDOORS – IOS APPLICATION

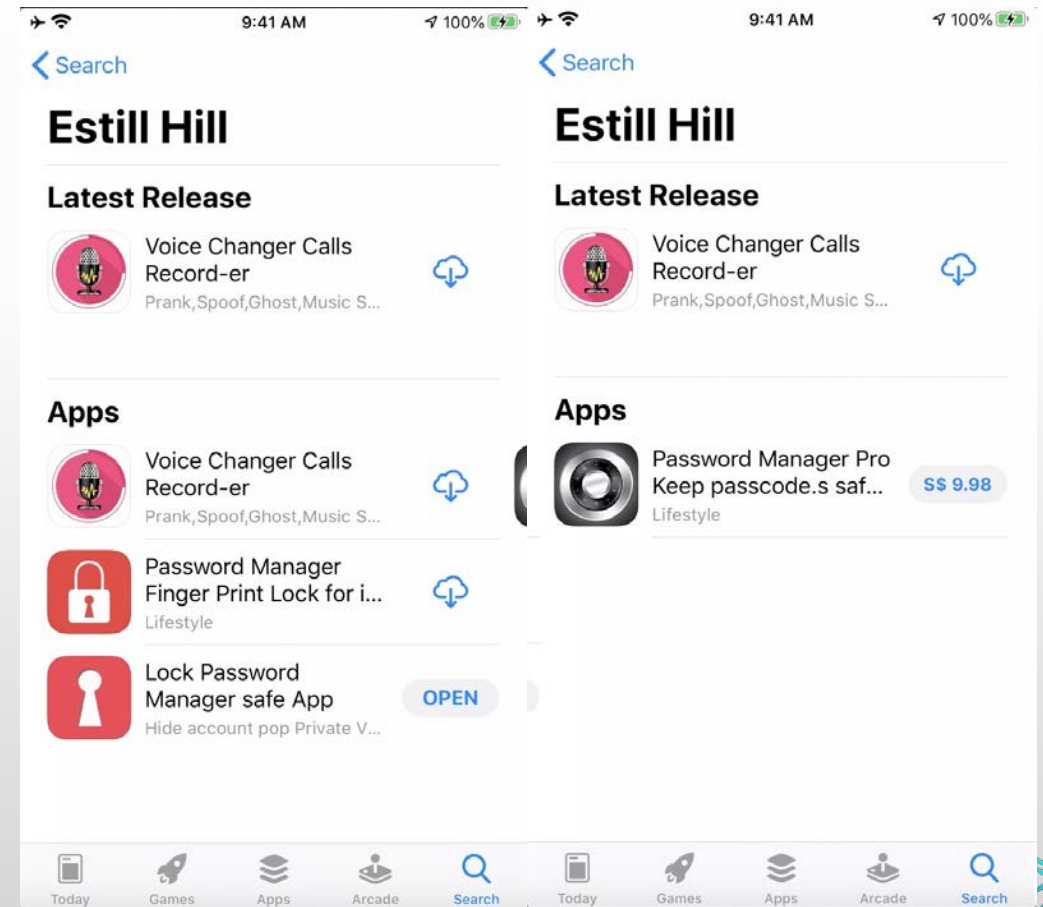
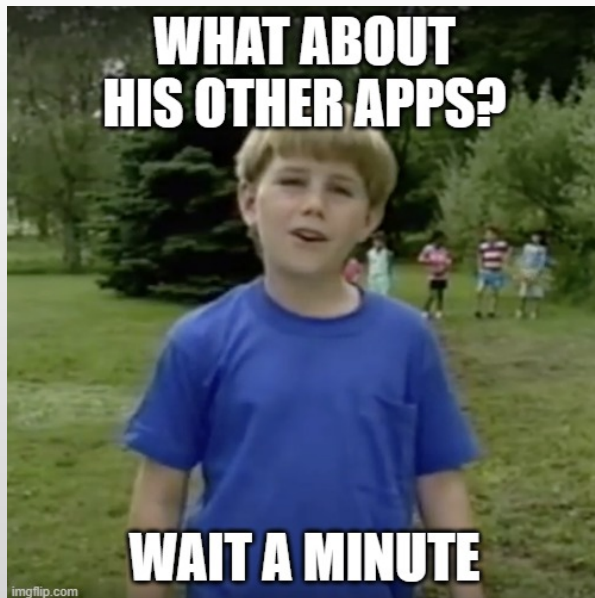
- So much for telling users to backup their master password





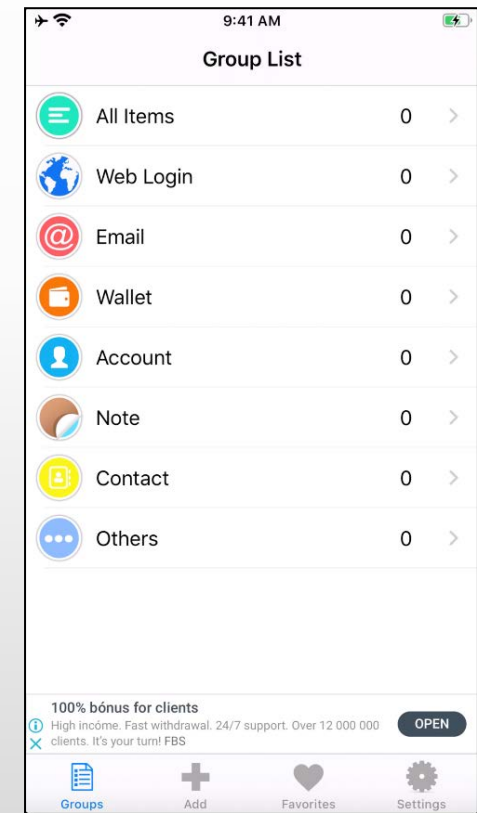
# DEVELOPER BACKDOORS – IOS APPLICATION

- 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?

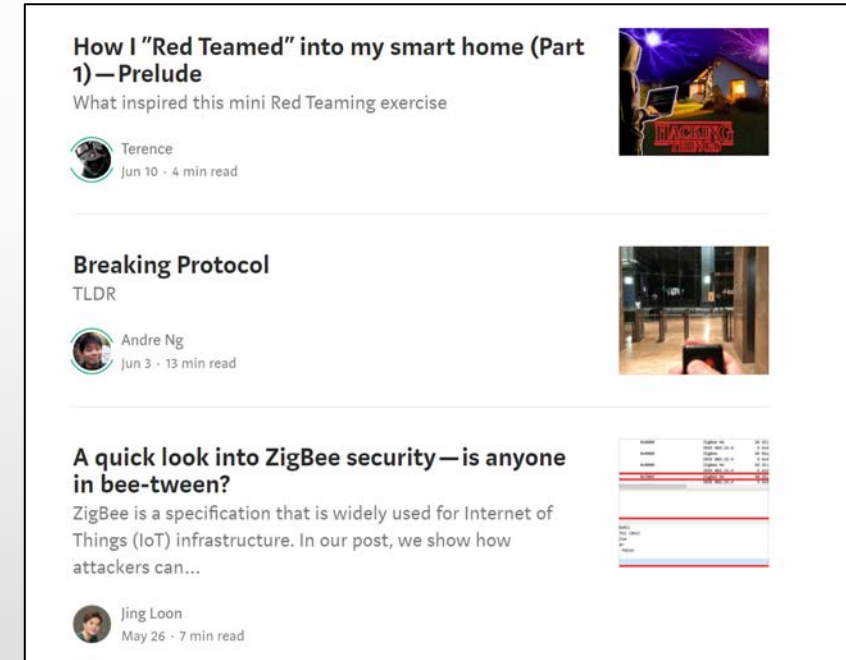
- Reported to developer multiple times since 2 Jan 2020. Since there was no reply from developer, I also tried to reach out to Apple. To date, the vulnerable app is still available on the AppStore.



# 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!
  - 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
  - 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

 angelystor & max-chee

 @angelystor



CYBER SECURITY  
GROUP