Detecting Reverse Engineering with Canaries

Collin Mulliner

Security Researcher and Engineer

collin@3BLabs.com

Twitter: @collinrm

About Me

- Security since ~1994 (hard to tell)
    - BS, MS, PhD in computer science (all focused on security)
- Early work: mobile app development (1997 for PalmOS)
- Worked a lot on MMS (2005-2006) and SMS (2009-2011) security
- Involved in several books on smartphone security
- Worked at mobile device manufacturers and mobile app security team
- Spent (a lot of) time at academic and industry research labs
- Also worked on: Windows desktop app security, anti ransomware, and consumer electronics security
- I find and report issues to manufacturers

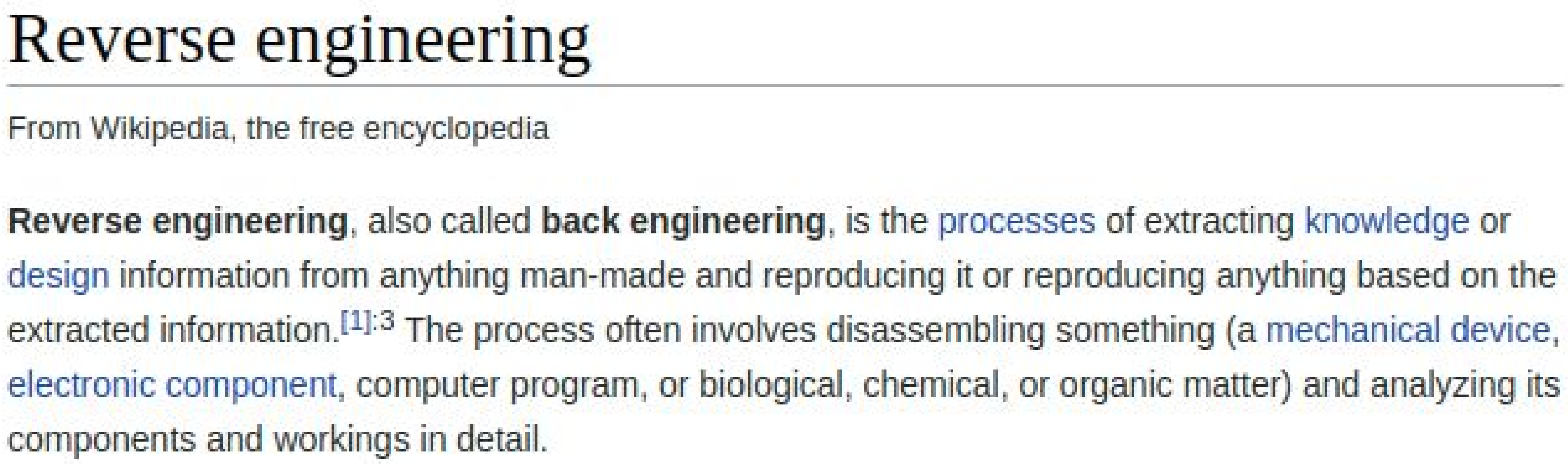
www.3blabs.com

Disclaimer!

- I don’t claim to have invented any of this!
- This talk is about using existing techniques for something newish
- The idea likely is already used by people who don’t talk about it!
- I’m not saying that reverse engineering is bad or malicious by definition!
- Also: I don’t (try/want to) sell you anything!

Goals

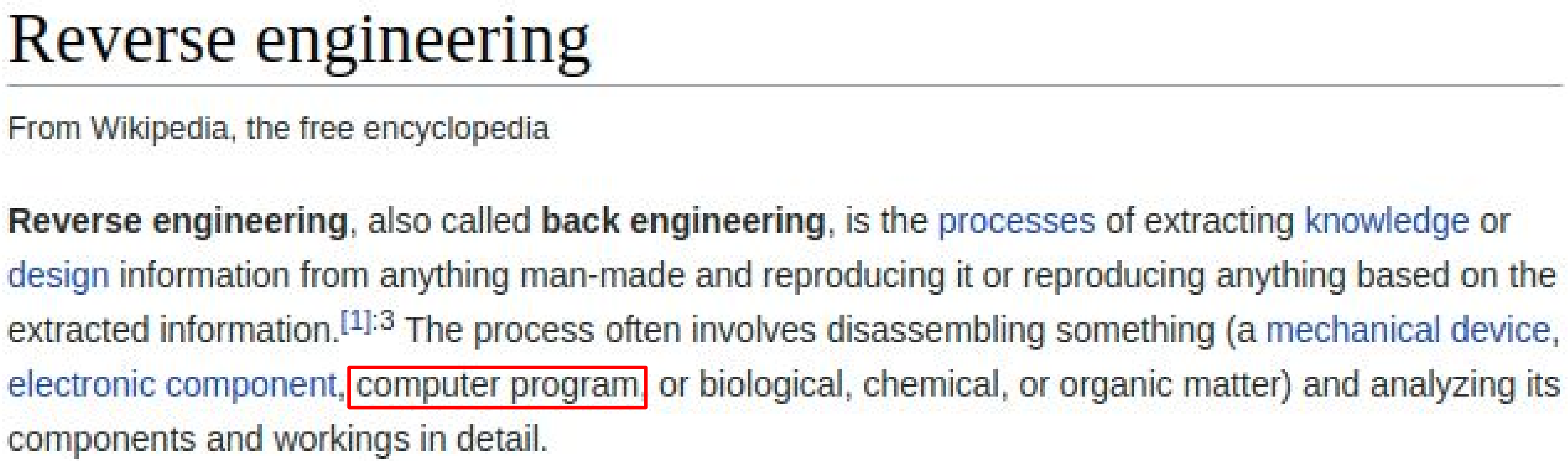
- Reverse Engineering vs Security
- The idea behind RE-Canaries
- How to make RE-Canaries work in the real world
- ~~Fun and Profit~~



Reverse engineering

From Wikipedia, the free encyclopedia

Reverse engineering, also called **back engineering**, is the processes of extracting knowledge or design information from anything man-made and reproducing it or reproducing anything based on the extracted information.[1]:3 The process often involves disassembling something (a mechanical device, electronic component, computer program, or biological, chemical, or organic matter) and analyzing its components and workings in detail.



Reverse engineering

From Wikipedia, the free encyclopedia

Reverse engineering, also called **back engineering**, is the processes of extracting knowledge or design information from anything man-made and reproducing it or reproducing anything based on the extracted information.[1]:3 The process often involves disassembling something (a mechanical device, electronic component, computer program, or biological, chemical, or organic matter) and analyzing its components and workings in detail.

Reverse Engineering Goals

- General understanding of how something works
  - “I just want to know how this works”
- Security
  - Discover vulnerabilities
    - Design
    - Implementation
- Re-implement “proprietary” software components
  - e.g., alternative client for networked service
- Determine possible IP violation (intellectual property)

Reverse Engineering Why Would You Care?

“The first step of attacking a target is recon!” --Brandon Edwards aka DrRaid

- Reverse engineering the target software is mandatory/basic reconnaissance
- Knowing this (about your software) gives you a strategic advantage
  - Prepare countermeasures
  - Collect evidence

Anything (interesting) will be Reverse Engineered

You don't want to find out about it from a talk announcement or the news

Anything (interesting) will be Reverse Engineered

you and your software are being targeted

You don’t want to find out about ~~it~~ from a talk announcement or the news

Software RE Techniques: Static Analysis

- Analyze the software on disk (without executing it)
  - Translate code
    - Assemly
    - High level language pseudocode
- Lot of tools (there is a market for this)
  - Disassembler
  - Binary Diffing
  - Code Fingerprinting
- Program understanding is an entire research field
  - Academic & Industry

Hex-Rays
state-of-the-art code analysis

zynamics BinDiff

IDA Pro
by Ilfak Guilfanov

RE using Dynamic Analysis

- Execute code and observe behavior
  - System calls, network traffic, file system activity
  - Debug or instrument software (monitor code that does not interact with the OS)

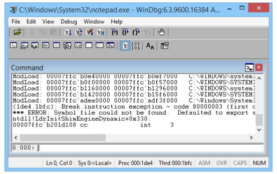
- Again a lot of tooling
  - Specialized debuggers and instrumentation tools
  - Sandboxes to execute code in controlled environment (e.g., malware analysis)
  - also network traffic analysis...

strace

Pin

VMRAY

DEBUGGER


C:\Windows\System32\notepad.exe - WinDbg:6.3.9600.16384 A...
File
Edit
View
Debug
Window
Help
Command

```
ModLoad: 00007ffc`b0e40000 00007ffc`b0ef7000   C:\WINDOWS\system3
ModLoad: 00007ffc`b0f00000 00007ffc`b0f57000   C:\WINDOWS\SYSTEM3
ModLoad: 00007ffc`b1160000 00007ffc`b1296000   C:\WINDOWS\system3
ModLoad: 00007ffc`b1420000 00007ffc`b15f6000   C:\WINDOWS\SYSTEM3
ModLoad: 00007ffc`adea0000 00007ffc`adf3f000   C:\Windows\System3
(1de4.1bfc): Break instruction exception - code 80000003 (first c
*** ERROR: Symbol file could not be found.  Defaulted to export s
ntdll!LdrInitShimEngineDynamic+0x330:
00007ffc`b201d108 cc              int     3
0:000>
```

Ln 0, Col 0 Sys 0:<Local> Proc 000:1de4 Thrd 000:1bfc ASM OVR CAPS NUM

Static vs. Dynamic *(really basic comparison!)*

- Dynamic analysis is much faster → results are not as detailed
  - Can be *detected and *prevented by the analyzed target (*to a certain degree)
    - e.g., vm & sandbox detection, anti-debugging

- Static analysis takes more time → better results
  - Protections such as code obfuscation can make this very hard and time consuming
  - Can't be detected since the target software is not being executed, no network communication

Reverse Engineering is HARD

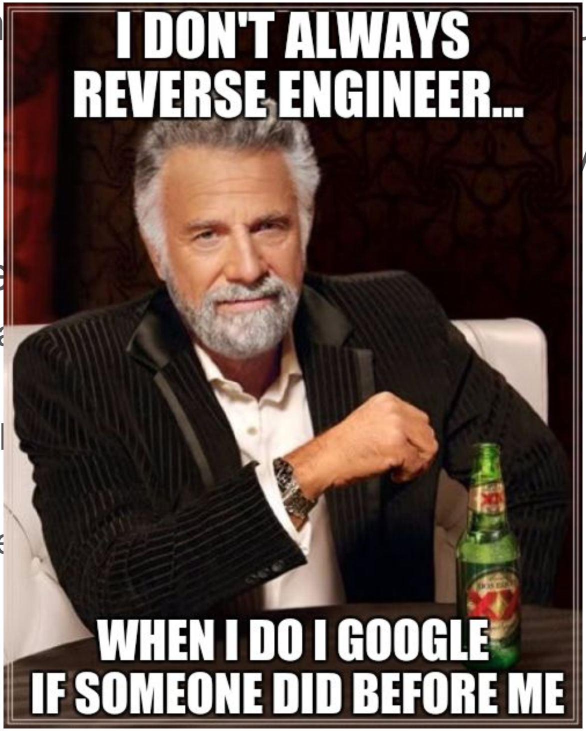
- In reality, both static and dynamic analysis will be used together!
- Very time consuming and tedious, also want to save time → time is money
- People like to cut corners
  - Developer uses a software library
  - Revere Engineer...
    - what libraries are being used?
    - what is this thing?
    - Did somebody already reverse this software?

Reverse Engineering is HARD

- In reality, both static an used together!

- Very time consuming a ve time → time is money

- People like to cut corne
  - Developer uses a softw
  - Revere Engineer...
    - what libraries are l
    - what is this thing?
    - Did somebody alre


I DON'T ALWAYS REVERSE ENGINEER...
WHEN I DO I GOOGLE IF SOMEONE DID BEFORE ME

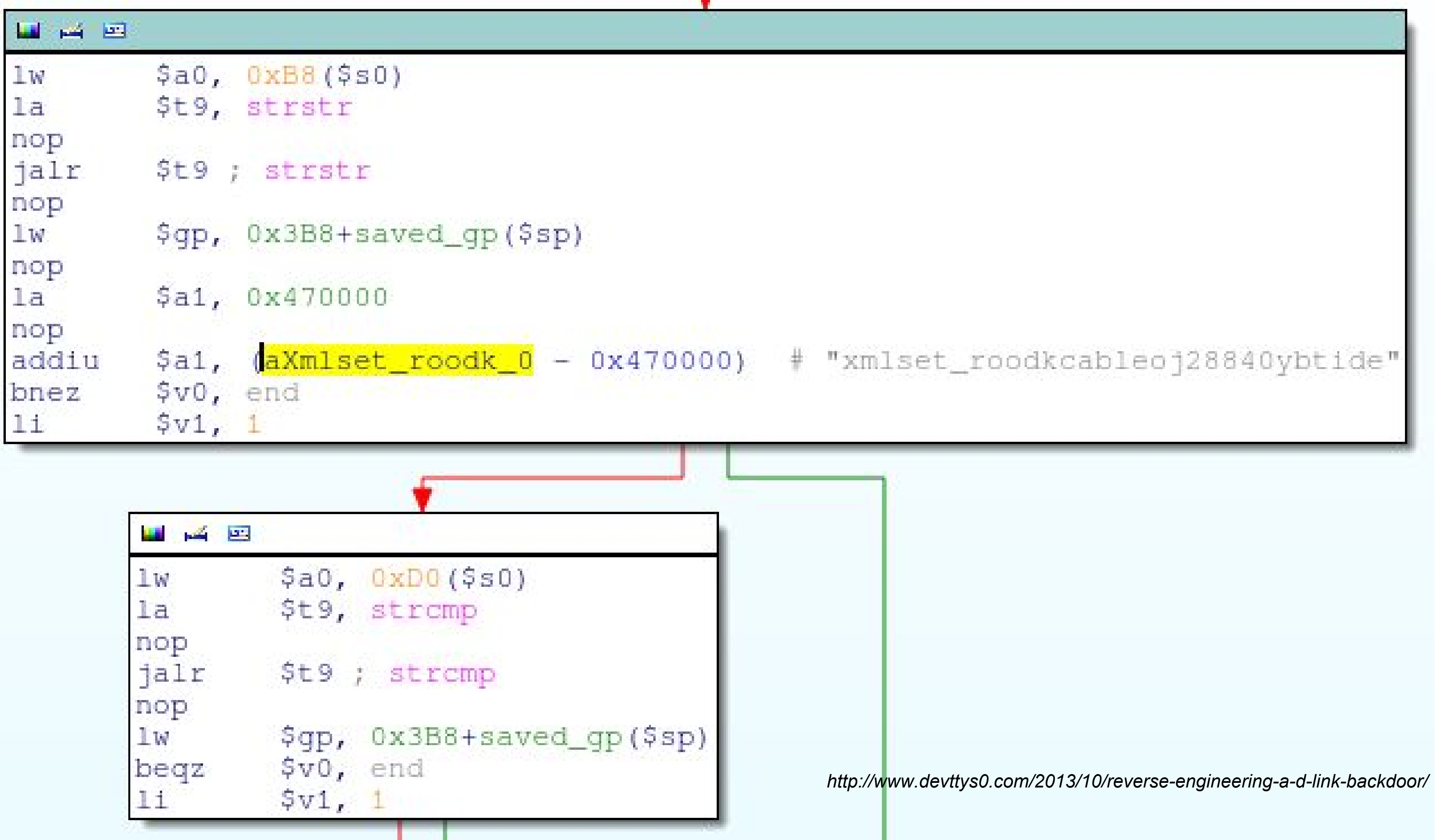
Can we detect the reverse engineer when he is cutting corners?

Reverse Engineering … an Example

...let’s say a wifi-router

- Download firmware update or extract firmware from the device
- Load it into your favorite disassembler

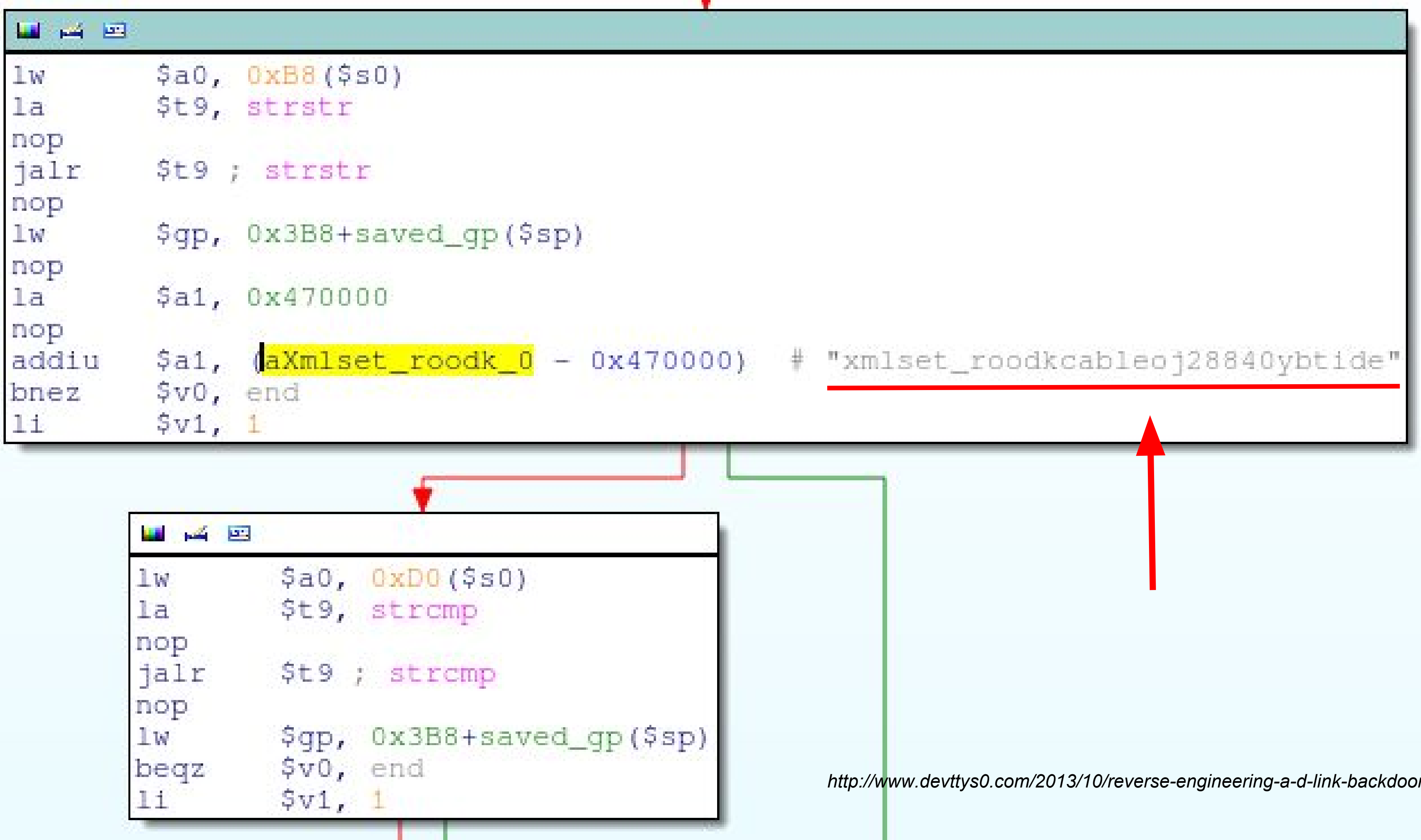
Just an example!



```
lw      $a0, 0xB8($s0)
la      $t9, strstr
nop
jalr    $t9 ; strstr
nop
lw      $gp, 0x3B8+saved_gp($sp)
nop
la      $a1, 0x470000
nop
addiu   $a1, (aXmlset_roodk_0 - 0x470000)  # "xmlset_roodkcableoj28840ybtide"
bnez    $v0, end
li      $v1, 1
```

```
lw      $a0, 0xD0($s0)
la      $t9, strcmp
nop
jalr    $t9 ; strcmp
nop
lw      $gp, 0x3B8+saved_gp($sp)
beqz    $v0, end
li      $v1, 1
```

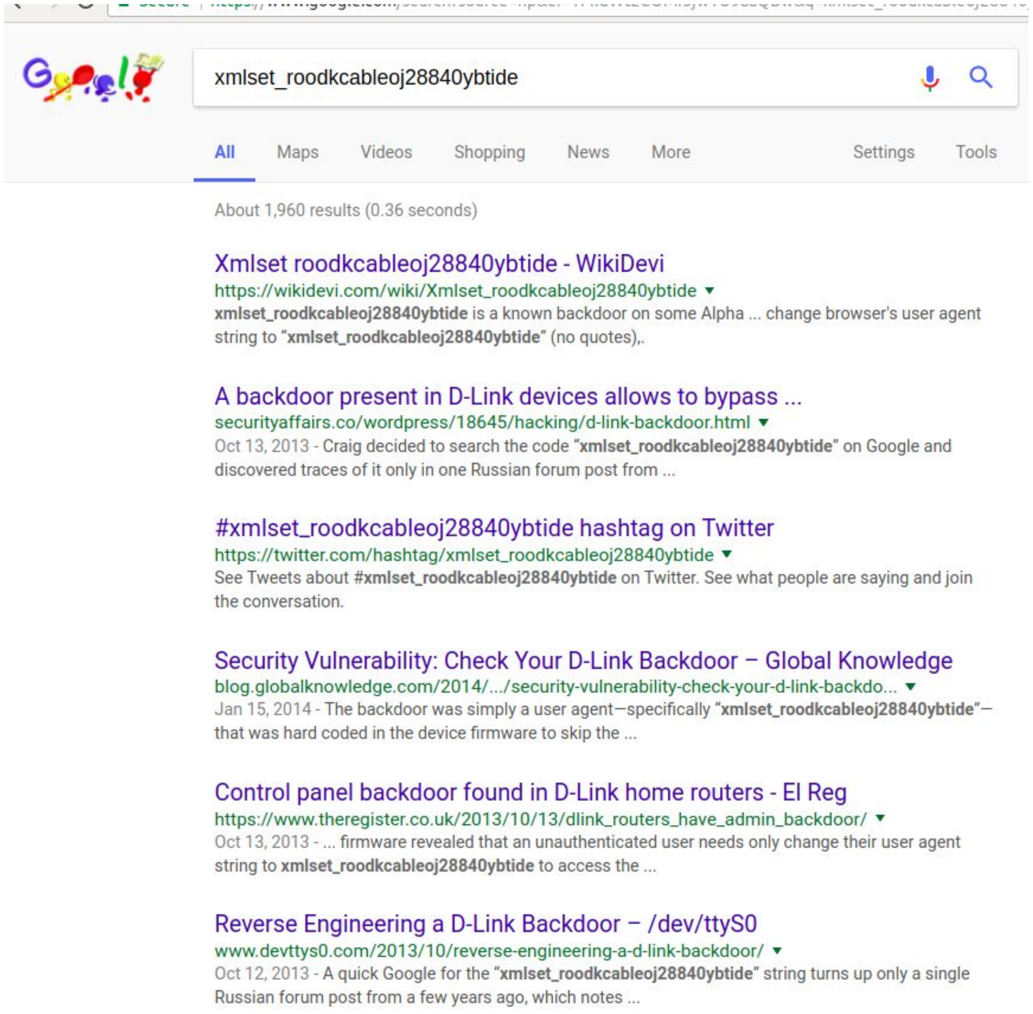
http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/



```
lw      $a0, 0xB8($s0)
la      $t9, strstr
nop
jalr    $t9 ; strstr
nop
lw      $gp, 0x3B8+saved_gp($sp)
nop
la      $a1, 0x470000
nop
addiu   $a1, (aXmlset_roodk_0 - 0x470000)  # "xmlset_roodkcableoj28840ybtide"
bnez    $v0, end
li      $v1, 1
```

```
lw      $a0, 0xD0($s0)
la      $t9, strcmp
nop
jalr    $t9 ; strcmp
nop
lw      $gp, 0x3B8+saved_gp($sp)
beqz    $v0, end
li      $v1, 1
```

http://www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/



xmlset_roodkcableoj28840ybtide

All Maps Videos Shopping News More Settings Tools

About 1,960 results (0.36 seconds)

Xmlset roodkcableoj28840ybtide - WikiDevi
https://wikidevi.com/wiki/Xmlset_roodkcableoj28840ybtide ▾
xmlset_roodkcableoj28840ybtide is a known backdoor on some Alpha ... change browser's user agent string to "**xmlset_roodkcableoj28840ybtide**" (no quotes),.

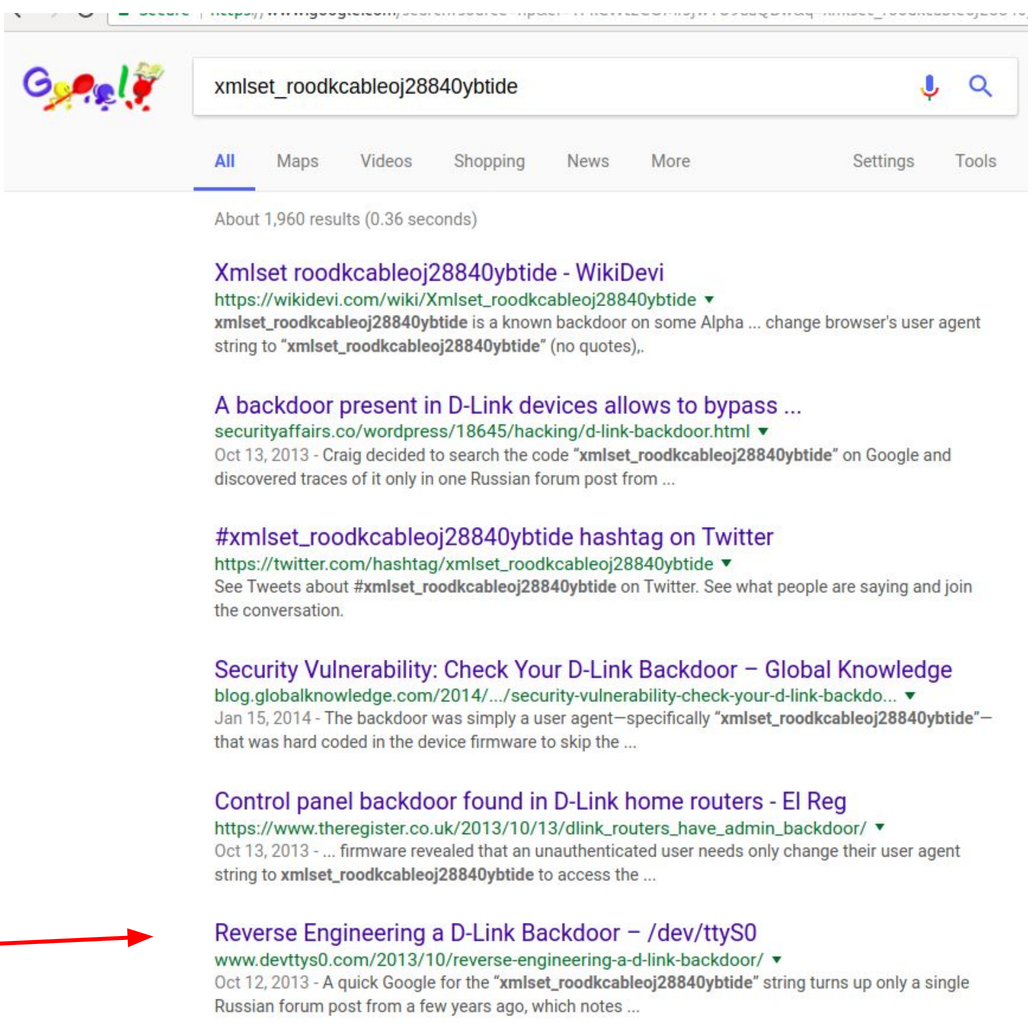
A backdoor present in D-Link devices allows to bypass ...
securityaffairs.co/wordpress/18645/hacking/d-link-backdoor.html ▾
Oct 13, 2013 - Craig decided to search the code "**xmlset_roodkcableoj28840ybtide**" on Google and discovered traces of it only in one Russian forum post from ...

#xmlset_roodkcableoj28840ybtide hashtag on Twitter
https://twitter.com/hashtag/xmlset_roodkcableoj28840ybtide ▾
See Tweets about #**xmlset_roodkcableoj28840ybtide** on Twitter. See what people are saying and join the conversation.

Security Vulnerability: Check Your D-Link Backdoor – Global Knowledge
blog.globalknowledge.com/2014/.../security-vulnerability-check-your-d-link-backdo... ▾
Jan 15, 2014 - The backdoor was simply a user agent—specifically "**xmlset_roodkcableoj28840ybtide**"—that was hard coded in the device firmware to skip the ...

Control panel backdoor found in D-Link home routers - El Reg
https://www.theregister.co.uk/2013/10/13/dlink_routers_have_admin_backdoor/ ▾
Oct 13, 2013 - ... firmware revealed that an unauthenticated user needs only change their user agent string to **xmlset_roodkcableoj28840ybtide** to access the ...

Reverse Engineering a D-Link Backdoor – /dev/ttyS0
www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/ ▾
Oct 12, 2013 - A quick Google for the "**xmlset_roodkcableoj28840ybtide**" string turns up only a single Russian forum post from a few years ago, which notes ...



xmlset_roodkcableoj28840ybtide

All Maps Videos Shopping News More Settings Tools

About 1,960 results (0.36 seconds)

Xmlset roodkcableoj28840ybtide - WikiDevi
https://wikidevi.com/wiki/Xmlset_roodkcableoj28840ybtide
xmlset_roodkcableoj28840ybtide is a known backdoor on some Alpha ... change browser's user agent string to "**xmlset_roodkcableoj28840ybtide**" (no quotes),.

A backdoor present in D-Link devices allows to bypass ...
securityaffairs.co/wordpress/18645/hacking/d-link-backdoor.html
Oct 13, 2013 - Craig decided to search the code "**xmlset_roodkcableoj28840ybtide**" on Google and discovered traces of it only in one Russian forum post from ...

#xmlset_roodkcableoj28840ybtide hashtag on Twitter
https://twitter.com/hashtag/xmlset_roodkcableoj28840ybtide
See Tweets about #**xmlset_roodkcableoj28840ybtide** on Twitter. See what people are saying and join the conversation.

Security Vulnerability: Check Your D-Link Backdoor – Global Knowledge
blog.globalknowledge.com/2014/.../security-vulnerability-check-your-d-link-backdo...
Jan 15, 2014 - The backdoor was simply a user agent—specifically "**xmlset_roodkcableoj28840ybtide**"—that was hard coded in the device firmware to skip the ...

Control panel backdoor found in D-Link home routers - El Reg
https://www.theregister.co.uk/2013/10/13/dlink_routers_have_admin_backdoor/
Oct 13, 2013 - ... firmware revealed that an unauthenticated user needs only change their user agent string to **xmlset_roodkcableoj28840ybtide** to access the ...

Reverse Engineering a D-Link Backdoor – /dev/ttyS0
www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/
Oct 12, 2013 - A quick Google for the "**xmlset_roodkcableoj28840ybtide**" string turns up only a single Russian forum post from a few years ago, which notes ...

www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/

/DEV/TTYS0 Embedded Device Hacking

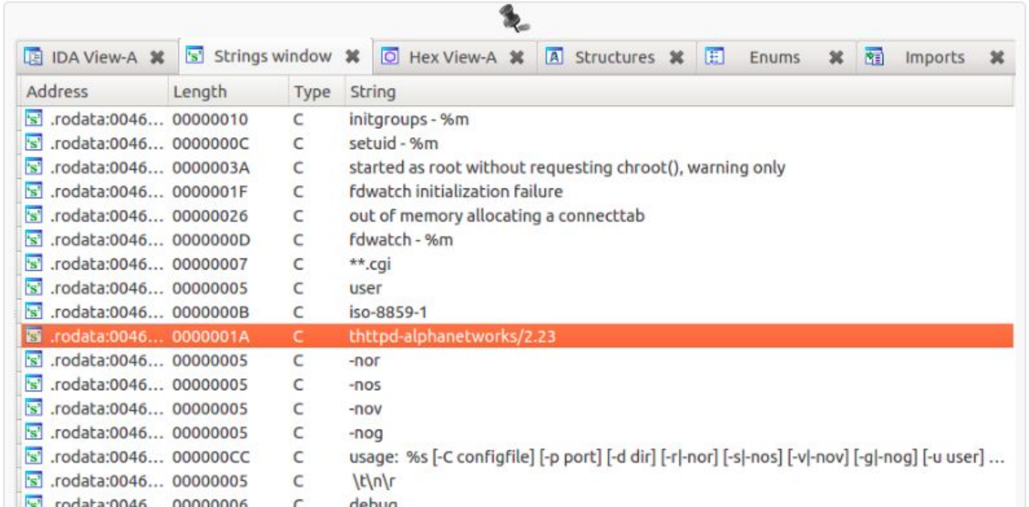
« Vegas, Baby! Some IDA Plugins »

Reverse Engineering a D-Link Backdoor

By Craig | October 12, 2013 | Embedded Systems, Security

All right. It's Saturday night, I have no date, a two-liter bottle of Shasta and my all-Rush mix-tape…let's hack.

On a whim I downloaded firmware v1.13 for the DIR-100 revA. Binwalk quickly found and extracted a SquashFS file system, and soon I had the firmware's web server (/bin/webs) loaded into IDA:



Address	Length	Type	String
.rodata:0046...	00000010	C	initgroups - %m
.rodata:0046...	0000000C	C	setuid - %m
.rodata:0046...	0000003A	C	started as root without requesting chroot(), warning only
.rodata:0046...	0000001F	C	fdwatch initialization failure
.rodata:0046...	00000026	C	out of memory allocating a connecttab
.rodata:0046...	0000000D	C	fdwatch - %m
.rodata:0046...	00000007	C	**.cgi
.rodata:0046...	00000005	C	user
.rodata:0046...	0000000B	C	iso-8859-1
.rodata:0046...	0000001A	C	thttpd-alphanetworks/2.23
.rodata:0046...	00000005	C	-nor
.rodata:0046...	00000005	C	-nos
.rodata:0046...	00000005	C	-nov
.rodata:0046...	00000005	C	-nog
.rodata:0046...	000000CC	C	usage: %s [-C configfile] [-p port] [-d dir] [-r|-nor] [-s|-nos] [-v|-nov] [-g|-nog] [-u user] ...
.rodata:0046...	00000005	C	\t\n\r
.rodata:0046...	00000006	C	debug

Search OK

Recent Posts

- Defcon 24: Blinded By The Light
- Hardware Hacking Workshop is Now Live!
- Binwalk v2.1.1 Stable Release
- What the Ridiculous Fuck, D-Link?!
- Hacking the D-Link DIR-890L

Recent Comments

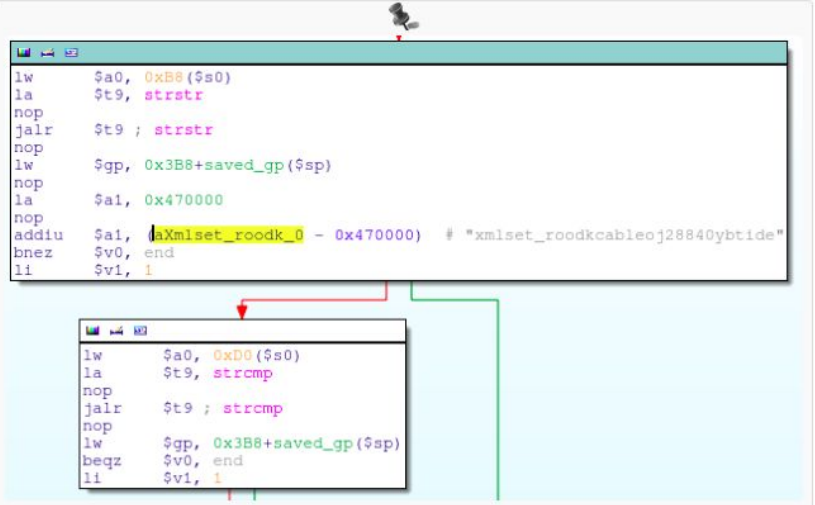
- Al Andrews on Jailbreaking the NeoTV
- dlink modem 8 digit pin – قيمة Rdlj on Reversing D-Link's WPS Pin Algorithm
- 44CON议题《攻击 VxWorks：从石器时代到星际》探究 - 莹莹之色 on Reverse Engineering VxWorks Firmware: WRT54Gv8
- Sergey on Reverse Engineering Serial Ports
- Eileen Eulich on Cracking WPA in 10 Hours or Less

Archives

- August 2016
- January 2016

www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/

It is the final strcmp however, which proves a bit more compelling:



```
lw      $a0, 0xB8($s0)
la      $t9, strstr
nop
jalr    $t9 ; strstr
nop
lw      $gp, 0x3B8+saved_gp($sp)
nop
la      $a1, 0x470000
nop
addiu   $a1, (aXmlset_roodk_0 - 0x470000)  # "xmlset_roodkcableoj28840ybtide"
bnez    $v0, end
li      $v1, 1
```

```
lw      $a0, 0xD0($s0)
la      $t9, strcmp
nop
jalr    $t9 ; strcmp
nop
lw      $gp, 0x3B8+saved_gp($sp)
beqz    $v0, end
li      $v1, 1
```

An interesting string comparison in alpha_auth_check

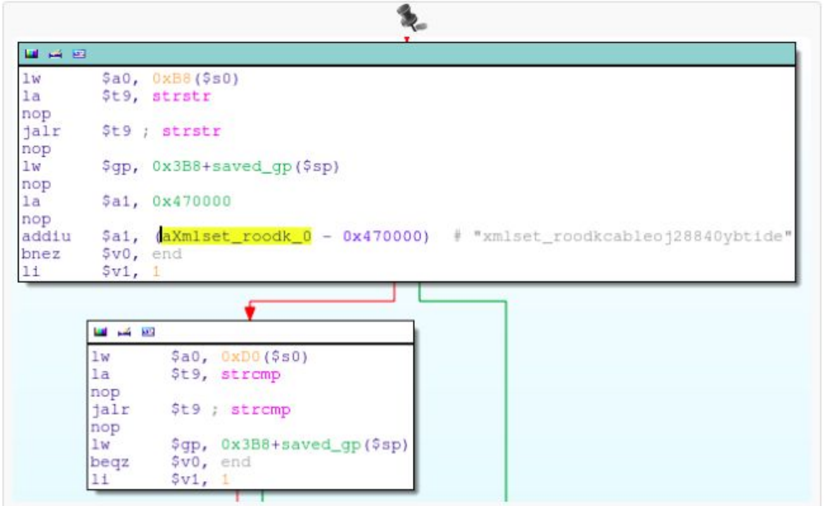
This is performing a strcmp between the string pointer at offset 0xD0 inside the **http_request_t** structure and the string "xmlset_roodkcableoj28840ybtide"; if the strings match, the **check_login** function call is skipped and **alpha_auth_check** returns 1 (authentication OK).

A quick Google for the "xmlset_roodkcableoj28840ybtide" string turns up only a single Russian forum post from a few years ago, which notes that this is an "interesting line" inside the /bin/webs binary. I'd have to agree.

So what is this mystery string getting compared against? If we look back in the call tree, we see that the **http_request_t** structure pointer is passed around by a few functions:

www.devttys0.com/2013/10/reverse-engineering-a-d-link-backdoor/

It is the final strcmp however, which proves a bit more compelling:



```
lw      $a0, 0xB8($s0)
la      $t9, strstr
nop
jalr    $t9 ; strstr
nop
lw      $gp, 0x3B8+saved_gp($sp)
nop
la      $a1, 0x470000
nop
addiu   $a1, (aXmlset_roodk_0 - 0x470000)  # "xmlset_roodkcableoj28840ybtide"
bnez    $v0, end
li      $v1, 1
```

```
lw      $a0, 0xD0($s0)
la      $t9, strcmp
nop
jalr    $t9 ; strcmp
nop
lw      $gp, 0x3B8+saved_gp($sp)
beqz    $v0, end
li      $v1, 1
```

An interesting string comparison in alpha_auth_check

This is performing a strcmp between the string pointer at offset 0xD0 inside the **http_request_t** structure and the string "xmlset_roodkcableoj28840ybtide"; if the strings match, the **check_login** function call is skipped and **alpha_auth_check** returns 1 (authentication OK).

A quick Google for the "xmlset_roodkcableoj28840ybtide" string turns up only a single Russian forum post from a few years ago, which notes that this is an "interesting line" inside the /bin/webs binary. I'd have to agree.

So what is this mystery string getting compared against? If we look back in the call tree, we see that the **http_request_t** structure pointer is passed around by a few functions:

So this guy searched the web...

- ...for this really **unique string**
  - Seems like this was one of the first things he did

- Only **one single search result** at the time
  - This means he would checkout that result for sure!

So this guy searched the web...

- ...for this really **unique string**
  - Seems like this was one of the first things he did
- Only **one single search result** at the time
  - This means he would checkout that result for sure!
- What if this site was run by the manufacturer?
  - If they monitored access to this specific page they know somebody looked at their software

Detecting Reverse Engineering *(side effects)*

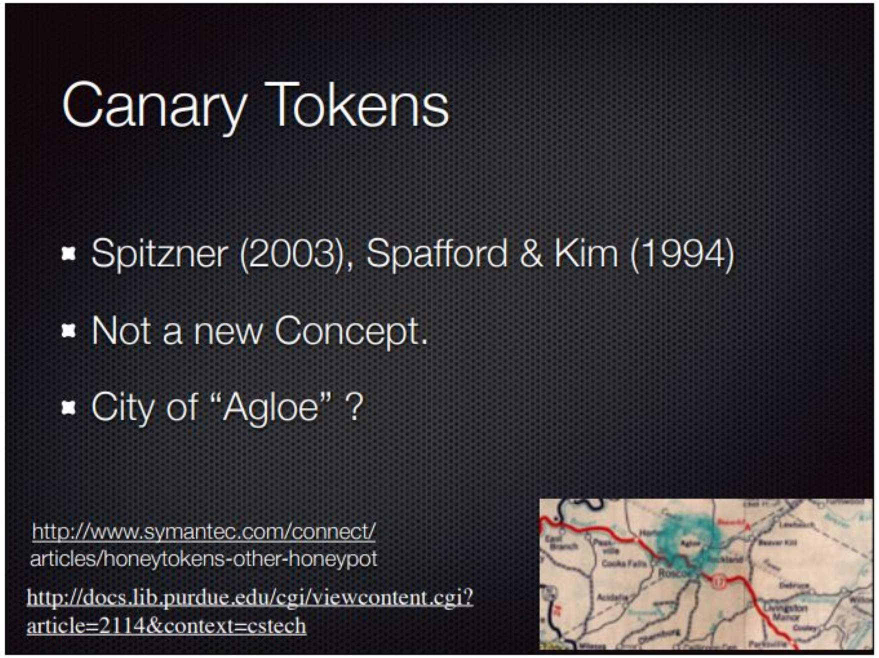
...seems to be absolutely possible

The idea:

Embed Canary Tokens into software to help detect reverse engineering

What are Canary Tokens?

- Unique identifiers that can be embedded in different places
    - If they are touched you get an alert
- OLD concept
    - Fake city in a map to detect if map was copied from you
- Also known as: honeytokens



Canary Tokens

- Spitzner (2003), Spafford & Kim (1994)
- Not a new Concept.
- City of “Agloe” ?

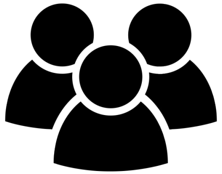
http://www.symantec.com/connect/articles/honeytokens-other-honeypot

http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2114&context=cstech

Bring Back Honeypots by the guys from Thinkst (Black Hat USA 2015)

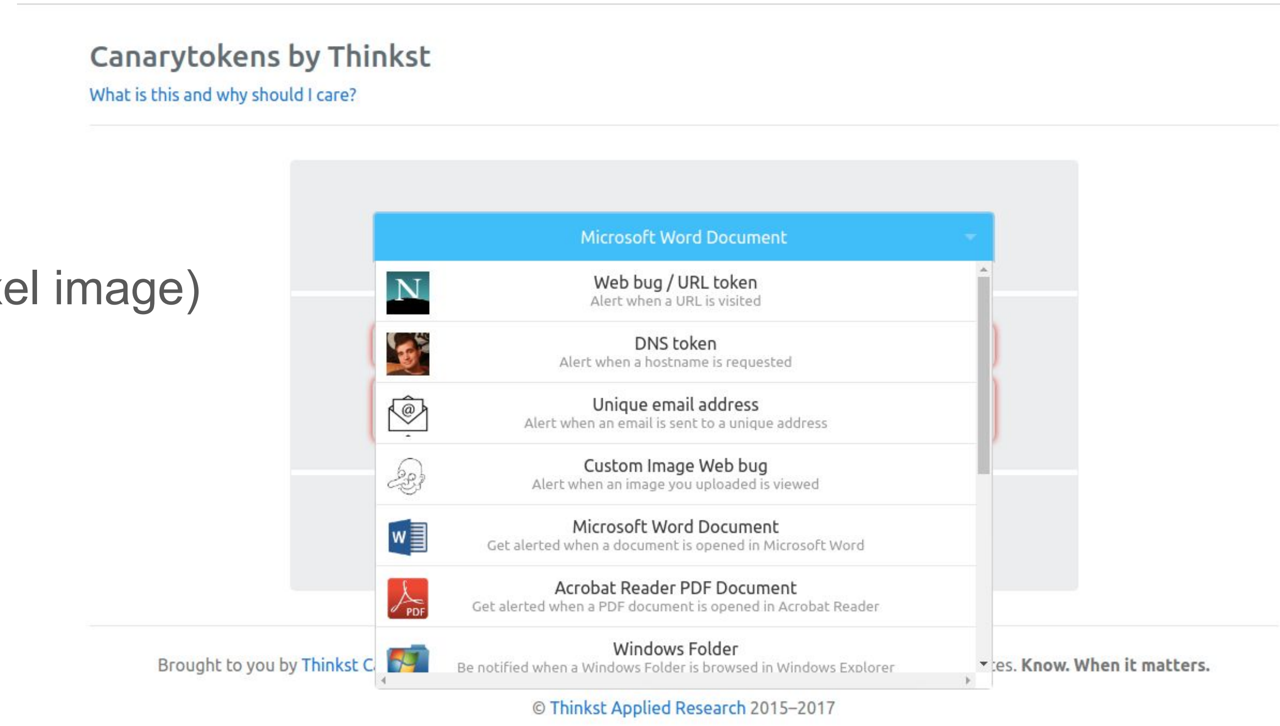
Canary/Honey Tokens already used in lot of Places

- Honeypot/honeynet → scan or login will trigger alarm
- Database row with trigger → read whole database will trigger alarm
- User account with (no privileges) weak password → login will trigger
- Canarytokens.org → open exfiled doc calls home and triggers alarm



Canarytokens.org aka Inspiration for this Work

- URL token
- DNS token
- Web bug (aka 1x1 pixel image)
- Documents...


Canarytokens by Thinkst
What is this and why should I care?

Microsoft Word Document

Web bug / URL token	Alert when a URL is visited
DNS token	Alert when a hostname is requested
Unique email address	Alert when an email is sent to a unique address
Custom Image Web bug	Alert when an image you uploaded is viewed
Microsoft Word Document	Get alerted when a document is opened in Microsoft Word
Acrobat Reader PDF Document	Get alerted when a PDF document is opened in Acrobat Reader
Windows Folder	Be notified when a Windows Folder is browsed in Windows Explorer

Brought to you by Thinkst C… …es. Know. When it matters.

© Thinkst Applied Research 2015–2017

RE-Canary

- Token embedded into software to help detect reverse engineering
    - Unique token (e.g., `SDFDkjfd983743223`)
    - Multiple tokens: tokens in different components of the software
- Hope somebody triggers the canary by activating the token
    - e.g., web search for the token to gain more information about the target software
- Defensive tool to notify and alert you about “attacks”
    - Attacks against you and your software

RE-Canary cont.

- Help you determine how much your adversary might know
- Which of your defenses are gone
  - Time to add new ones?
- Prepare for adversarial action
  - Alert your SecOps people
- Create log trail for legal case
  - Compromise
  - **IP violation** (more than just security!)

RE-Canaries: an Approach to Level the Playing Field

- Dynamic analysis can be detected by the software manufacturer
    - IF software uses network and network access is not restricted (call home, crash report, ...)
        - Can’t be avoided in many circumstances

- Static analysis doesn’t execute software → no network traffic
    - No: logs, crash dumps, ...
    - No interaction with your infrastructure!

- RE-Canaries bring the attacker back to your infrastructure
    - You can observe what he is doing and take defensive action!

RE-Canaries: an Approach to Level the Playing Field

- Dynamic analysis can be detected by the software manufacturer
  - IF software uses network and network access is not restricted (call home, crash report, ...)
    - Can't be avoided in many circumstances

- Static analysis doesn't execute software → no network traffic
  - No: logs, crash dumps, ...
  - No interaction with your infrastructure!

- RE-Canaries bring the attacker back to your infrastructure
  - You can observe what he is doing and take defensive action!

Your adversary is playing a Capture the Flag (CTF) game, that you set up, without knowing it!

RE-Canary: Information Provided *(potential)*

- Action performed (assuming individual tokens for things)
- Date/Time: when did this happen
- IP address (location?)
- Browser information (for web-based canaries)
  - browser, OS, language, …

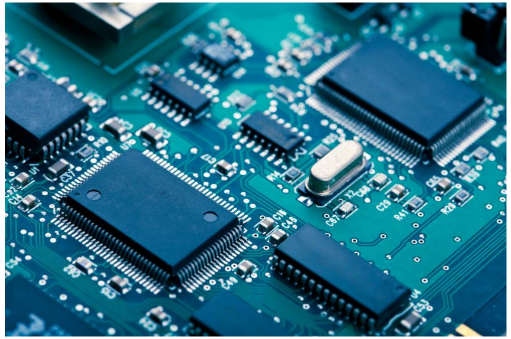
Canaries Invert the Attacker-Defender Problem

- We are used to think like this:
  - Defender has to get everything right 100% of the time
  - Attacker has to get lucky once!

- With canaries:
  - Attacker has to avoid 100% of the canaries
  - Defender has to get lucky once!

*Canaries **detect** and not defend (especially RE canaries!)*

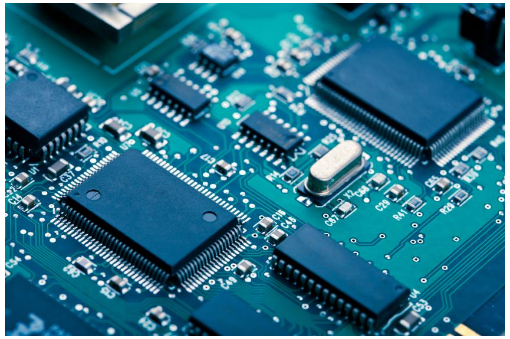
Application Areas



SOTFWARE
SOFTWARE EVERYWHERE

Devices: Consumer Electronics, Medical, Industrial,..

- All the chips run software
  - Software made by the manufacturer, you, or 3rd party
- With source code access
  - Canaries are easily added
- Binary blob
  - Reverse engineer and determine canaries (what would you search for?)
  - Attacker has to go thru the effort of opening device and extracting the software from the chip
  - Your security team gets the vendor blob from your hardware team



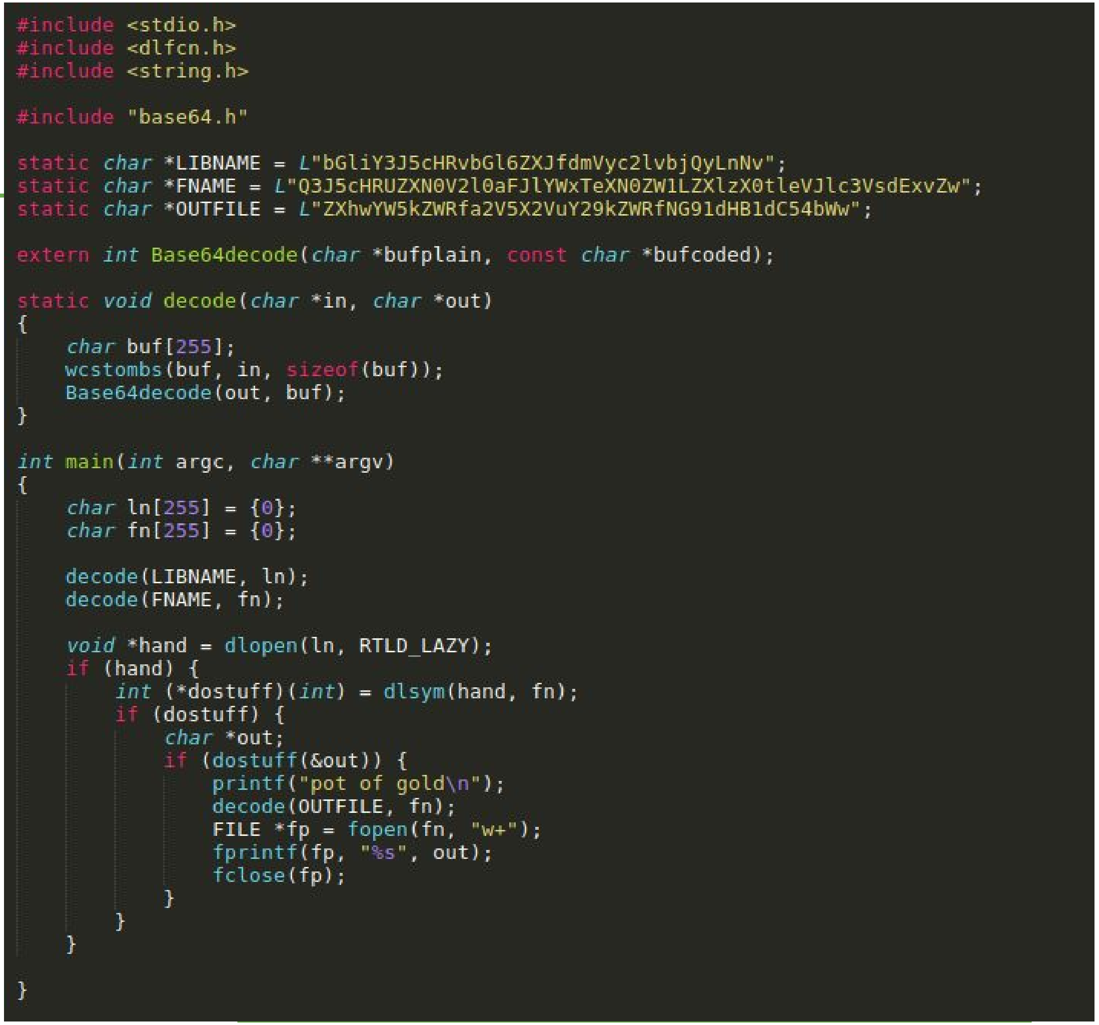
Example

Example

dlopen/dlsym canary

1. try to load library
2. if 1: lookup function
3. if 2: call function
4. if 3: log result

would normally do nothing!



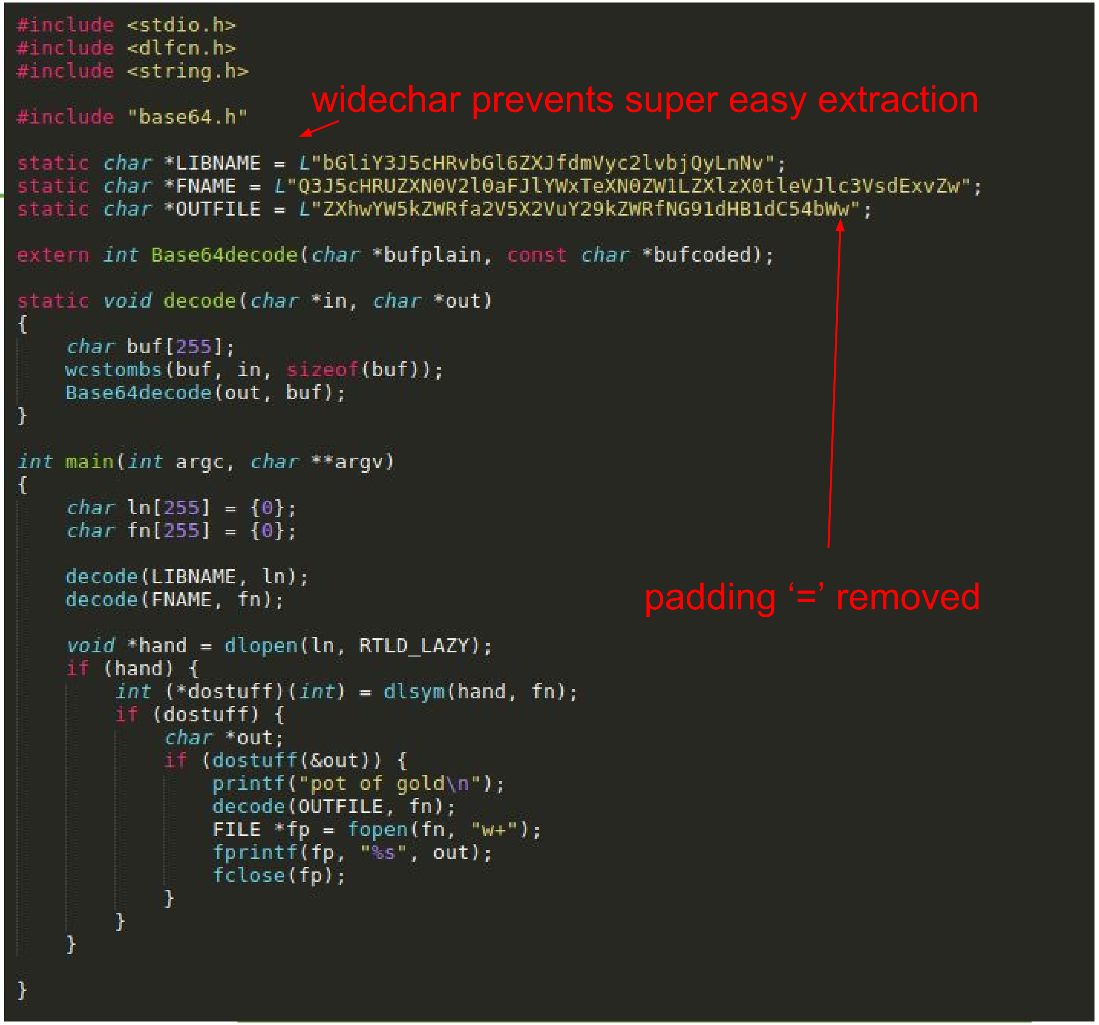
```c
#include <stdio.h>
#include <dlfcn.h>
#include <string.h>

#include "base64.h"

static char *LIBNAME = L"bGliY3J5cHRvbGl6ZXJfdmVyc2lvbjQyLnNv";
static char *FNAME = L"Q3J5cHRUZXN0V2l0aFJlYWxTeXN0ZW1LZXlzX0tleVJlc3VsdExvZw";
static char *OUTFILE = L"ZXhwYW5kZWRfa2V5X2VuY29kZWRfNG91dHB1dC54bWw";

extern int Base64decode(char *bufplain, const char *bufcoded);

static void decode(char *in, char *out)
{
    char buf[255];
    wcstombs(buf, in, sizeof(buf));
    Base64decode(out, buf);
}

int main(int argc, char **argv)
{
    char ln[255] = {0};
    char fn[255] = {0};

    decode(LIBNAME, ln);
    decode(FNAME, fn);

    void *hand = dlopen(ln, RTLD_LAZY);
    if (hand) {
        int (*dostuff)(int) = dlsym(hand, fn);
        if (dostuff) {
            char *out;
            if (dostuff(&out)) {
                printf("pot of gold\n");
                decode(OUTFILE, fn);
                FILE *fp = fopen(fn, "w+");
                fprintf(fp, "%s", out);
                fclose(fp);
            }
        }
    }

}
```

Example

dlopen/dlsym canary

1. try to load library
2. if 1: lookup function
3. if 2: call function
4. if 3: log result

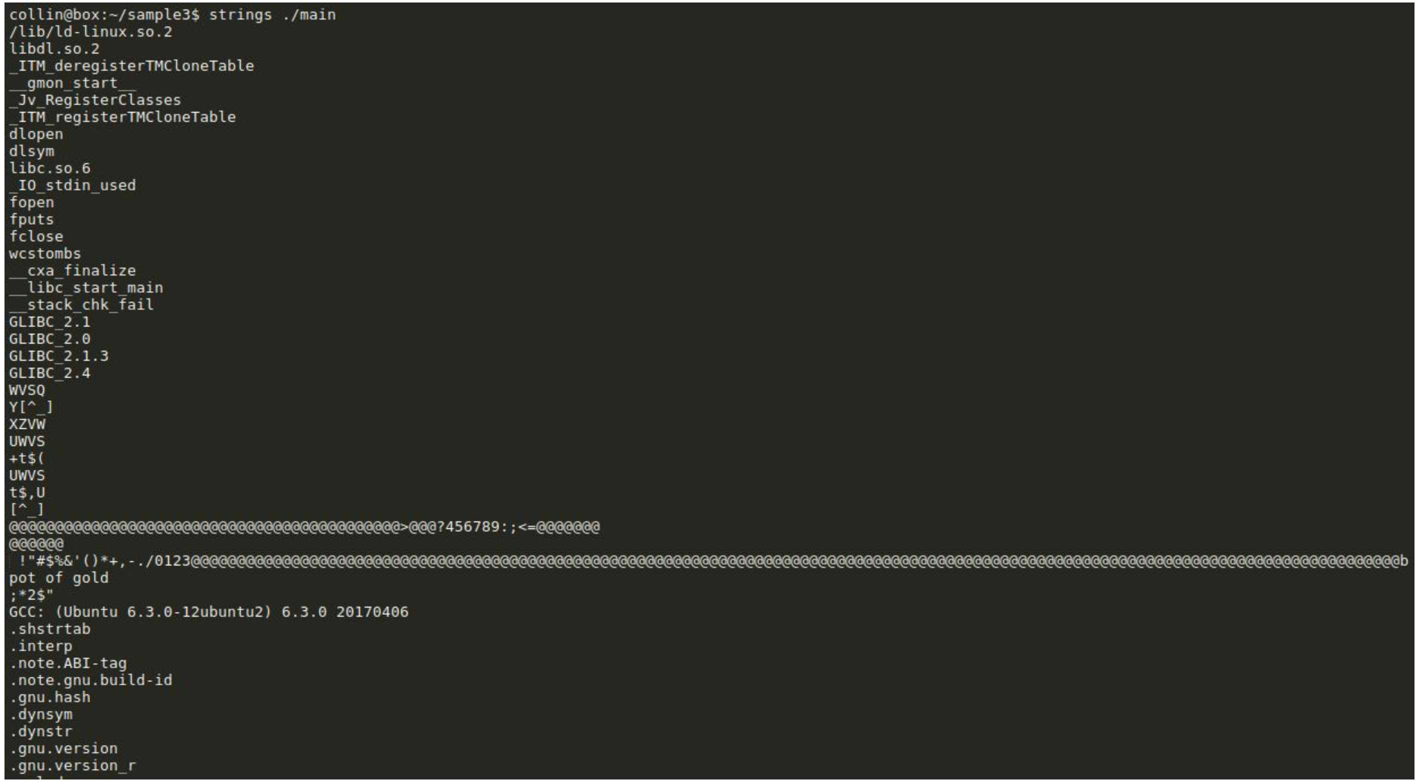
would normally do nothing!



```c
#include <stdio.h>
#include <dlfcn.h>
#include <string.h>

#include "base64.h"

static char *LIBNAME = L"bGliY3J5cHRvbGl6ZXJfdmVyc2lvbjQyLnNv";
static char *FNAME = L"Q3J5cHRUZXN0V2l0aFJlYWxTeXN0ZW1LZXlzX0tleVJlc3VsdExvZw";
static char *OUTFILE = L"ZXhwYW5kZWRfa2V5X2VuY29kZWRfNG91dHB1dC54bWw";

extern int Base64decode(char *bufplain, const char *bufcoded);

static void decode(char *in, char *out)
{
    char buf[255];
    wcstombs(buf, in, sizeof(buf));
    Base64decode(out, buf);
}

int main(int argc, char **argv)
{
    char ln[255] = {0};
    char fn[255] = {0};

    decode(LIBNAME, ln);
    decode(FNAME, fn);

    void *hand = dlopen(ln, RTLD_LAZY);
    if (hand) {
        int (*dostuff)(int) = dlsym(hand, fn);
        if (dostuff) {
            char *out;
            if (dostuff(&out)) {
                printf("pot of gold\n");
                decode(OUTFILE, fn);
                FILE *fp = fopen(fn, "w+");
                fprintf(fp, "%s", out);
                fclose(fp);
            }
        }
    }

}
```

Example… strings *(‘-e L’ will print wchar strings!)*

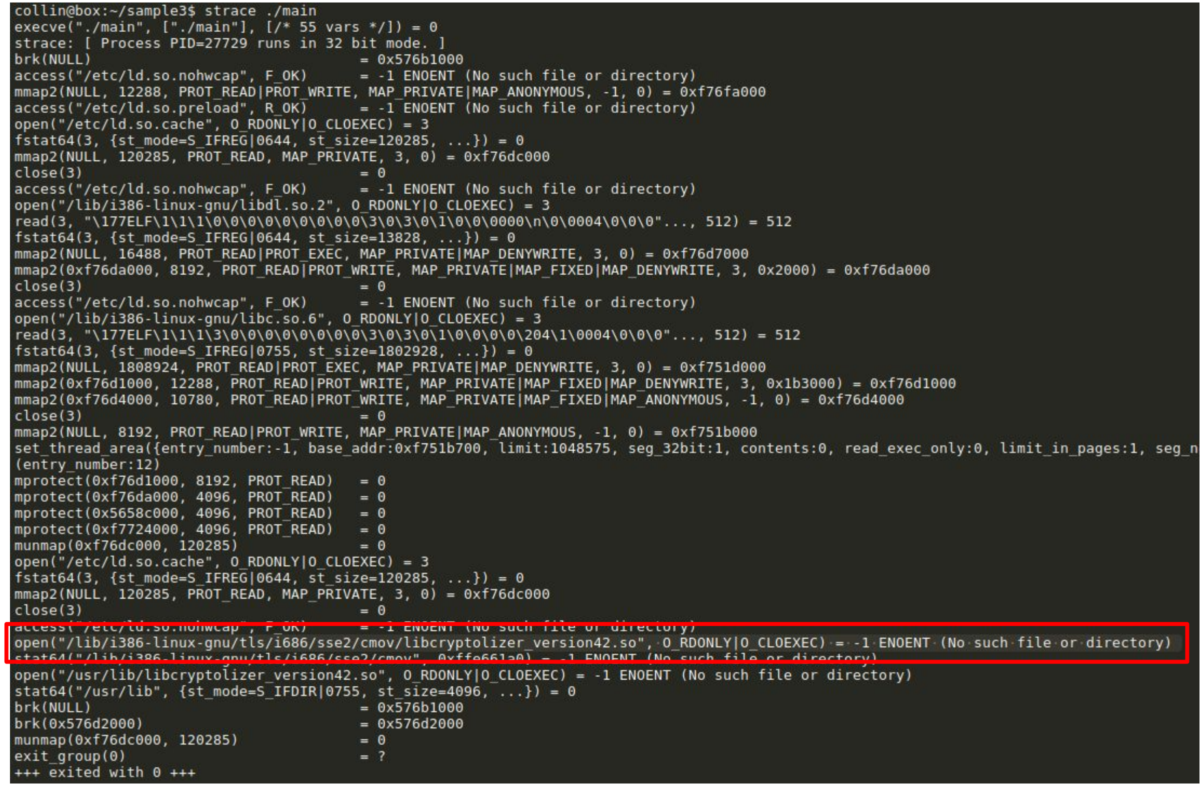
Nothing useful



```
collin@box:~/sample3$ strings ./main
/lib/ld-linux.so.2
libdl.so.2
_ITM_deregisterTMCloneTable
__gmon_start__
_Jv_RegisterClasses
_ITM_registerTMCloneTable
dlopen
dlsym
libc.so.6
_IO_stdin_used
fopen
fputs
fclose
wcstombs
__cxa_finalize
__libc_start_main
__stack_chk_fail
GLIBC_2.1
GLIBC_2.0
GLIBC_2.1.3
GLIBC_2.4
WVSQ
Y[^_]
XZVW
UWVS
+t$(
UWVS
t$,U
[^_]
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@>@@@?456789:;<=@@@@@@@
@@@@@@
 !"#$%&'()*+,-./0123@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@b
pot of gold
;*2$"
GCC: (Ubuntu 6.3.0-12ubuntu2) 6.3.0 20170406
.shstrtab
.interp
.note.ABI-tag
.note.gnu.build-id
.gnu.hash
.dynsym
.dynstr
.gnu.version
.gnu.version_r
```

Example… strace

open("libcryptolizer…")



```
collin@box:~/sample3$ strace ./main
execve("./main", ["./main"], [/* 55 vars */]) = 0
strace: [ Process PID=27729 runs in 32 bit mode. ]
brk(NULL)                               = 0x576b1000
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
mmap2(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xf76fa000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=120285, ...}) = 0
mmap2(NULL, 120285, PROT_READ, MAP_PRIVATE, 3, 0) = 0xf76dc000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/i386-linux-gnu/libdl.so.2", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0000\n\0\0004\0\0\0"..., 512) = 512
fstat64(3, {st_mode=S_IFREG|0644, st_size=13828, ...}) = 0
mmap2(NULL, 16488, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0xf76d7000
mmap2(0xf76da000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2000) = 0xf76da000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/i386-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\1\1\1\3\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0\204\1\0004\0\0\0"..., 512) = 512
fstat64(3, {st_mode=S_IFREG|0755, st_size=1802928, ...}) = 0
mmap2(NULL, 1808924, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0xf751d000
mmap2(0xf76d1000, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b3000) = 0xf76d1000
mmap2(0xf76d4000, 10780, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0xf76d4000
close(3)                                = 0
mmap2(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xf751b000
set_thread_area({entry_number:-1, base_addr:0xf751b700, limit:1048575, seg_32bit:1, contents:0, read_exec_only:0, limit_in_pages:1, seg_n
(entry_number:12)
mprotect(0xf76d1000, 8192, PROT_READ)   = 0
mprotect(0xf76da000, 4096, PROT_READ)   = 0
mprotect(0x5658c000, 4096, PROT_READ)   = 0
mprotect(0xf7724000, 4096, PROT_READ)   = 0
munmap(0xf76dc000, 120285)              = 0
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=120285, ...}) = 0
mmap2(NULL, 120285, PROT_READ, MAP_PRIVATE, 3, 0) = 0xf76dc000
close(3)                                = 0
access("/etc/ld.so.nohwcap", F_OK)      = -1 ENOENT (No such file or directory)
open("/lib/i386-linux-gnu/tls/i686/sse2/cmov/libcryptolizer_version42.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
stat64("/lib/i386-linux-gnu/tls/i686/sse2/cmov", 0xffe661a0) = -1 ENOENT (No such file or directory)
open("/usr/lib/libcryptolizer_version42.so", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
stat64("/usr/lib", {st_mode=S_IFDIR|0755, st_size=4096, ...}) = 0
brk(NULL)                               = 0x576b1000
brk(0x576d2000)                         = 0x576d2000
munmap(0xf76dc000, 120285)              = 0
exit_group(0)                           = ?
+++ exited with 0 +++
```

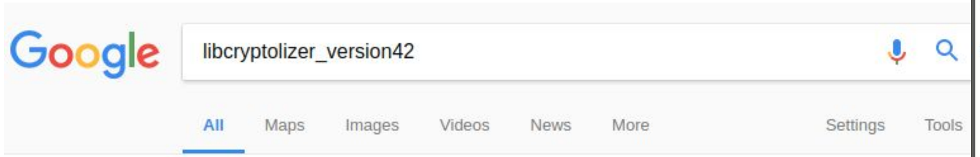
Example… walk-through

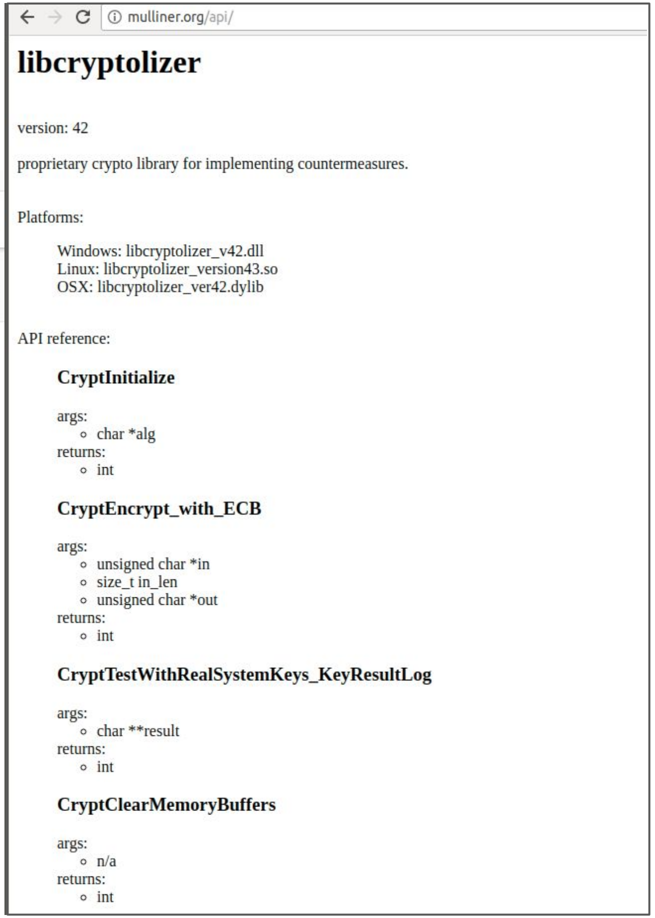
- Output of ‘strings’ is useless (god, we don’t care about you running strings!)
- strace delivers a filename: `libcryptolizer_version42.so`

- Additional dynamic or static analysis delivers two additional tokens
  - Function name: `CryptTestWithRealSystemKeys_KeyResultLog`
  - Output filename: `expanded_key_encoded_4output.xml`

- Analysis takes extra time depending on skill, but time was spent!

Bingo!





libcryptolizer

version: 42

proprietary crypto library for implementing countermeasures.

Platforms:

Windows: libcryptolizer_v42.dll
Linux: libcryptolizer_version43.so
OSX: libcryptolizer_ver42.dylib

API reference:

CryptInitialize

args:
- char *alg

returns:
- int

CryptEncrypt_with_ECB

args:
- unsigned char *in
- size_t in_len
- unsigned char *out

returns:
- int

CryptTestWithRealSystemKeys_KeyResultLog

args:
- char **result

returns:
- int

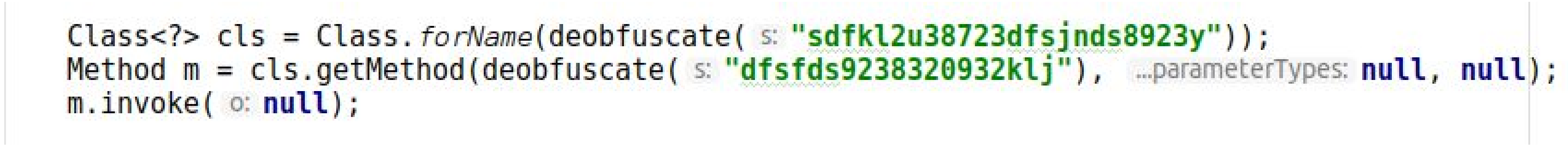
CryptClearMemoryBuffers

args:
- n/a

returns:
- int

Example 2

- Java reflection is often used for obfuscation
- Use it as a canary



```
Class<?> cls = Class.forName(deobfuscate( s: "sdfkl2u38723dfsjnds8923y"));
Method m = cls.getMethod(deobfuscate( s: "dfsfds9238320932klj"),  ...parameterTypes: null, null);
m.invoke( o: null);
```

- Put in try-catch block and do nothing if class/method not found
- Unique class/method name as the RE-Canary token

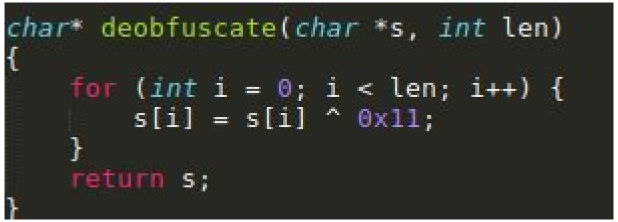
Some Words on String Obfuscation

- Problem: many things rely on strings, strings show a lot about functionality
- Goal: make strings "go away"

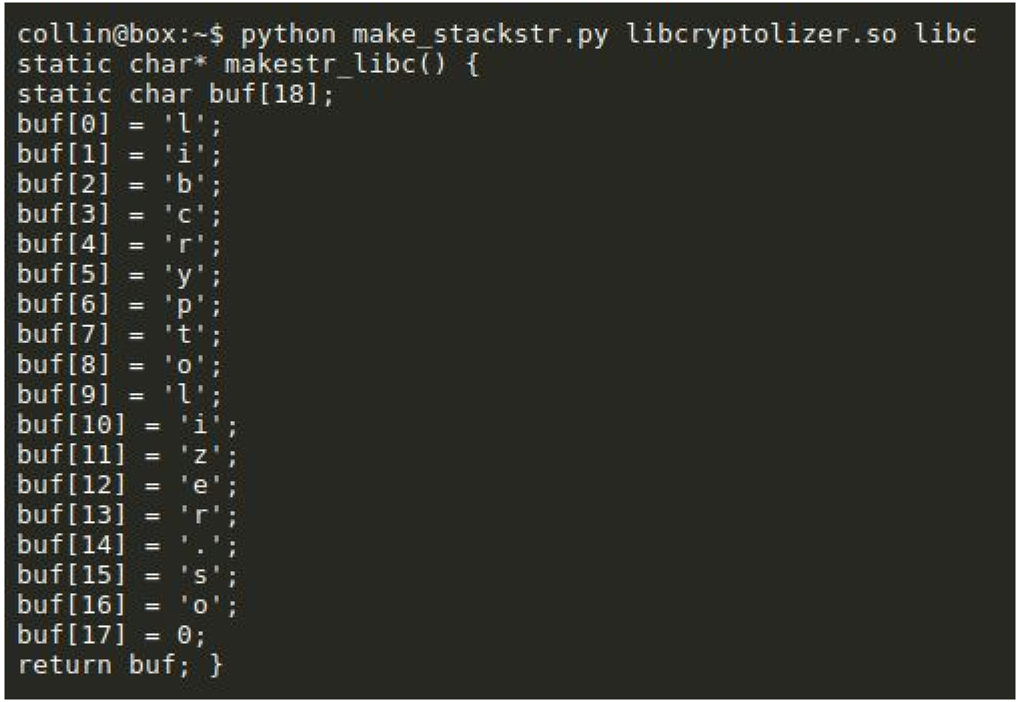
- Many many obfuscation techniques
  - **Encrypt strings in binary and decrypt at runtime:**
    - send(sock, STRING); → send(sock, decrypt(ENC_STRING, KEY));
  - Transform (e.g., base64, ROT13, 1-byte XOR)
  - Build string: string is stored as code, function builds and returns string
    (Malware provides a lot of interesting examples for this)

String Obfuscation cont.

- Real simple string obfuscation
  - Ideas taken from malware
- Can be automated during build



```c
char* deobfuscate(char *s, int len)
{
    for (int i = 0; i < len; i++) {
        s[i] = s[i] ^ 0x11;
    }
    return s;
}
```



```
collin@box:~$ python make_stackstr.py libcryptolizer.so libc
static char* makestr_libc() {
static char buf[18];
buf[0] = 'l';
buf[1] = 'i';
buf[2] = 'b';
buf[3] = 'c';
buf[4] = 'r';
buf[5] = 'y';
buf[6] = 'p';
buf[7] = 't';
buf[8] = 'o';
buf[9] = 'l';
buf[10] = 'i';
buf[11] = 'z';
buf[12] = 'e';
buf[13] = 'r';
buf[14] = '.';
buf[15] = 's';
buf[16] = 'o';
buf[17] = 0;
return buf; }
```

Word on Code Obfuscation

- Code obfuscation is commonly done to protect IP and to harden security code
  - Many products out there
- Compiler extension obfuscates code without developer doing anything special
  - Configure obfuscation level (blow up binary size by x%)
- Obfuscated code hardens string obfuscation
- Add RE-Canary to code that gets obfuscated to detect people breaking the obfuscation!

RE-Canaries Measure your Adversary’s Progress

- Multiple canary layers (obfuscation and hardening)
  - Canary from \`strings software.exe\` vs encrypted canary extracted from obfuscated code

- Time between alerts from different canaries

```
Canary 9879873232 triggered at 01/03/2017
Canary 8789722322 triggered at 04/13/2017
Canary 7672672643 triggered at 04/26/2017
```

- Maybe create canaries that look different depending on discovery technique
  - Different tools, dynamic vs static analysis, ...

RE-Canary Types...

I think we covered String Canaries!

API-Endpoints / URL Canaries

u = URL(“https://api.mulliner.net/service/auth_no_2fa”)

Anybody who hits the specific path triggers the canary

Hostname (DNS) Canary

u = URL("https://ds3klfjd3s4f3ldsdnal.3blabs.com")

Resolving the IP address for the host will trigger the canary

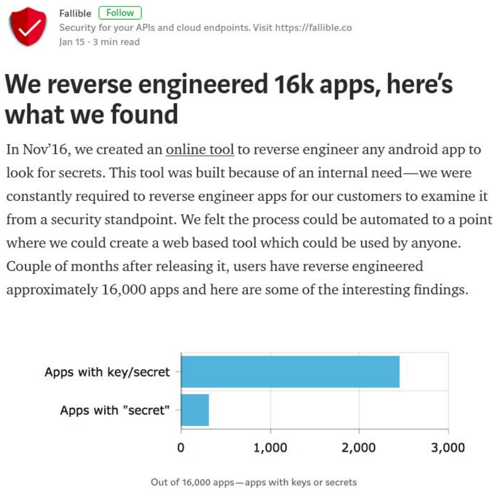
Symbol Canary

- Shared libraries have exported symbols
- Add unique symbol
- Similar to dlopen/reflection idea but actually adding it to your shared library
- Webpage with symbol (special version of library documentation)
- C++ name mangling ;-)

```
_ZNK3MapI10StringName3RefI8GDScriptE10ComparatorIS0_E16DefaultAllocatorE3hasERKS0_
```

(AWS) Credentials (in Mobile Apps)

- Hardcoded credentials are (sadly) a common thing
  - Obfuscate them to not be publicly shamed for doing it!

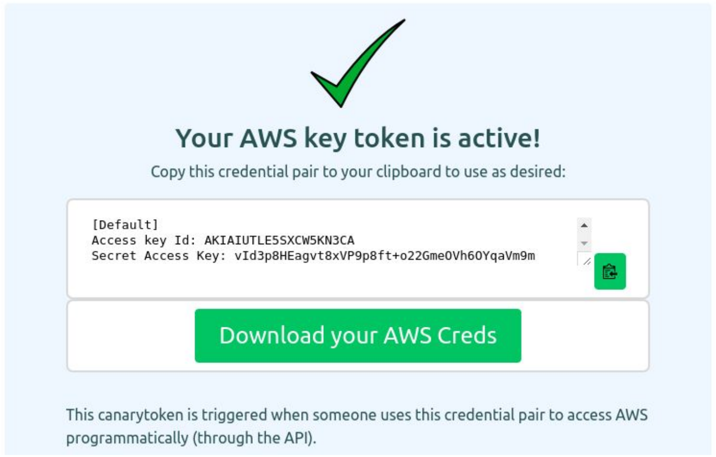


Fallible Follow
Security for your APIs and cloud endpoints. Visit https://fallible.co
Jan 15 · 3 min read

We reverse engineered 16k apps, here's what we found

In Nov'16, we created an online tool to reverse engineer any android app to look for secrets. This tool was built because of an internal need—we were constantly required to reverse engineer apps for our customers to examine it from a security standpoint. We felt the process could be automated to a point where we could create a web based tool which could be used by anyone. Couple of months after releasing it, users have reverse engineered approximately 16,000 apps and here are some of the interesting findings.

Out of 16,000 apps—apps with keys or secrets



Your AWS key token is active!

Copy this credential pair to your clipboard to use as desired:

```
[Default]
Access key Id: AKIAIUTLE5SXCW5KN3CA
Secret Access Key: vId3p8HEagvt8xVP9p8ft+o22GmeOVh6OYqaVm9m
```

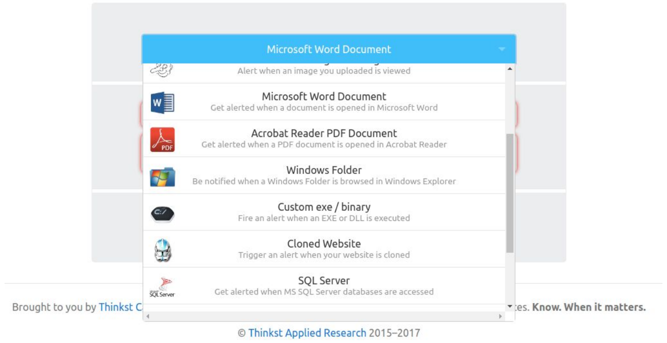
This canarytoken is triggered when someone uses this credential pair to access AWS programmatically (through the API).

canarytokens.org

Resource Canary

- Application packages contain resources alongside with code
  - e.g., APK, Jar
- Add file-based canary to app package
  - PDF, docx, html

- Poking around and opening files will trigger the canary


Microsoft Word Document

Alert when an image you uploaded is viewed

Microsoft Word Document
Get alerted when a document is opened in Microsoft Word

Acrobat Reader PDF Document
Get alerted when a PDF document is opened in Acrobat Reader

Windows Folder
Be notified when a Windows Folder is browsed in Windows Explorer

Custom exe / binary
Fire an alert when an EXE or DLL is executed

Cloned Website
Trigger an alert when your website is cloned

SQL Server
Get alerted when MS SQL Server databases are accessed

Brought to you by Thinkst C ... es. Know. When it matters.

© Thinkst Applied Research 2015–2017

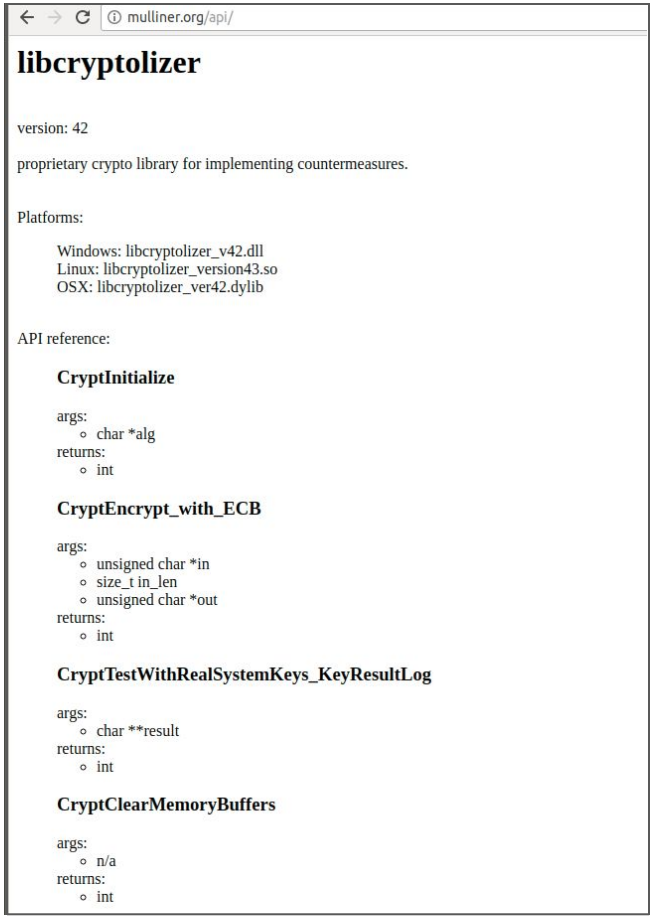
RE-Canary Properties

- Strings → web search
  - Via intermediate (search engine), could avoid triggering your alarm (e.g., view cached page)
- API-endpoint, URL, credentials
  - Connect to (YOUR) infrastructure and therefore guaranteed to trigger your alarms
- Canary files (e.g., documents from canarytokens.org)
  - Require network connectivity to trigger alarm (can be avoided easily)
    but adds another layer of obfuscation

Canary Trigger Deployment

Webpages...

- Fake API documentation
- Basically a site that just contains tokens



mulliner.org/api/

libcryptolizer

version: 42

proprietary crypto library for implementing countermeasures.

Platforms:

Windows: libcryptolizer_v42.dll
Linux: libcryptolizer_version43.so
OSX: libcryptolizer_ver42.dylib

API reference:

CryptInitialize

args:
- char *alg

returns:
- int

CryptEncrypt_with_ECB

args:
- unsigned char *in
- size_t in_len
- unsigned char *out

returns:
- int

CryptTestWithRealSystemKeys_KeyResultLog

args:
- char **result

returns:
- int

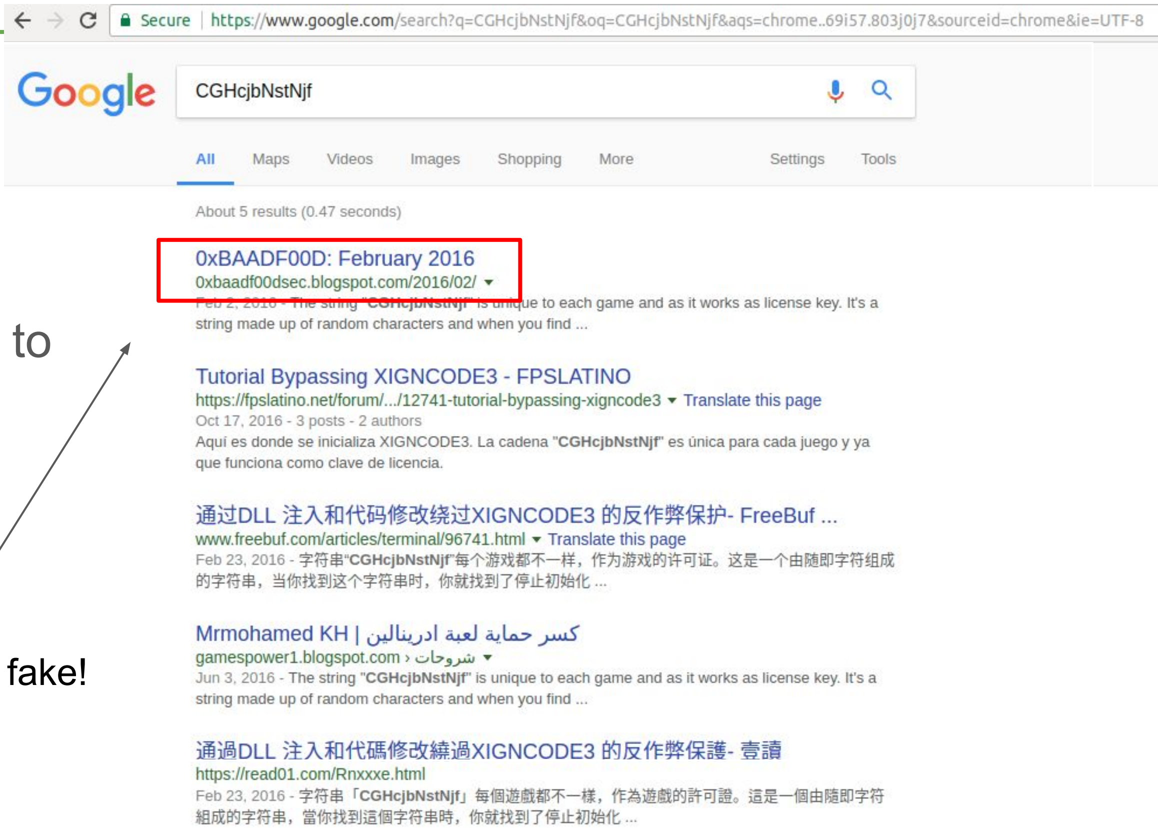
CryptClearMemoryBuffers

args:
- n/a

returns:
- int

Fake "Blog"

- Sites like blogspot help you to mask your identity (not important for canaries)
- Load content from your site to alert you
  - image
  - javascript



Secure | https://www.google.com/search?q=CGHcjbNstNjf&oq=CGHcjbNstNjf&aqs=chrome..69i57.803j0j7&sourceid=chrome&ie=UTF-8

CGHcjbNstNjf

About 5 results (0.47 seconds)

0xBAADF00D: February 2016
0xbaadf00dsec.blogspot.com/2016/02/
Feb 2, 2016 - The string "CGHcjbNstNjf" is unique to each game and as it works as license key. It's a string made up of random characters and when you find ...

Tutorial Bypassing XIGNCODE3 - FPSLATINO
https://fpslatino.net/forum/.../12741-tutorial-bypassing-xigncode3 Translate this page
Oct 17, 2016 - 3 posts - 2 authors
Aquí es donde se inicializa XIGNCODE3. La cadena "CGHcjbNstNjf" es única para cada juego y ya que funciona como clave de licencia.

通过DLL 注入和代码修改绕过XIGNCODE3 的反作弊保护- FreeBuf ...
www.freebuf.com/articles/terminal/96741.html Translate this page
Feb 23, 2016 - 字符串"CGHcjbNstNjf"每个游戏都不一样，作为游戏的许可证。这是一个由随即字符组成的字符串，当你找到这个字符串时，你就找到了停止初始化 ...

Mrmohamed KH | كسر حماية لعبة ادرينالين
gamespower1.blogspot.com › شروحات
Jun 3, 2016 - The string "CGHcjbNstNjf" is unique to each game and as it works as license key. It's a string made up of random characters and when you find ...

通過DLL 注入和代碼修改繞過XIGNCODE3 的反作弊保護- 壹讀
https://read01.com/Rnxxxe.html
Feb 23, 2016 - 字符串「CGHcjbNstNjf」每個遊戲都不一樣，作為遊戲的許可證。這是一個由隨即字符組成的字符串，當你找到這個字符串時，你就找到了停止初始化 ...

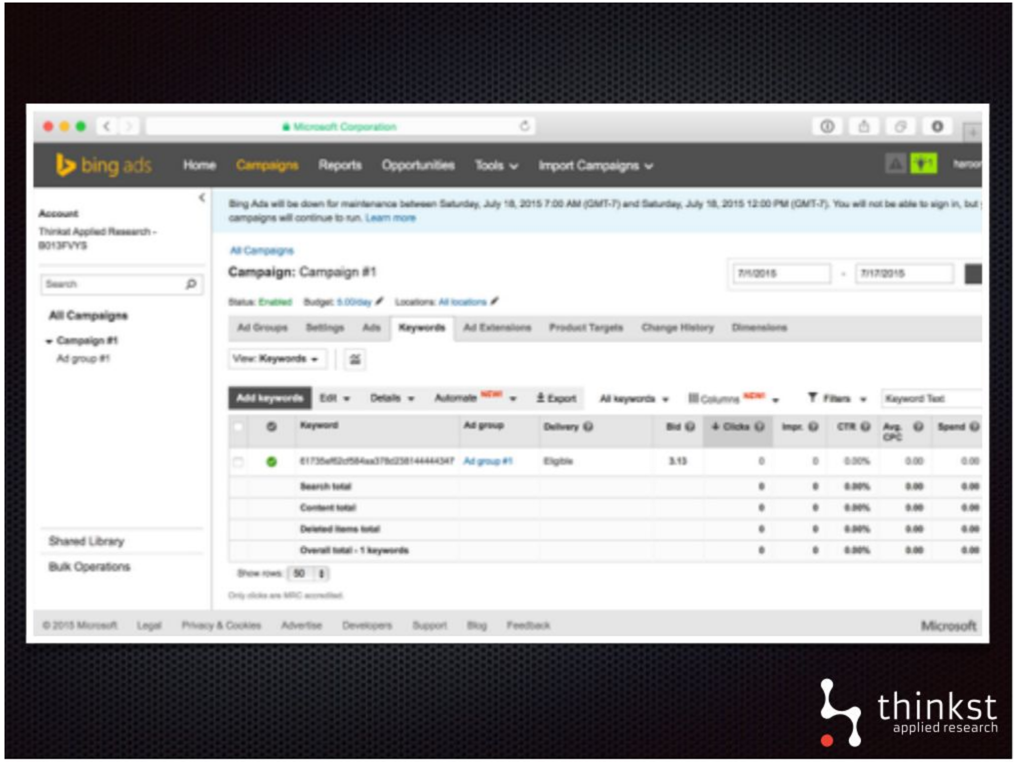
Not actually fake!

The Canarytokens.org Way

- Monitor your DNS
- Add “canaried” API-endpoint for your web services
  - Speak to your development team :)
- Create canary user account
  - No privileges!

Adwords

- Register token as an adword
- Automatic notification if searched for (without running a token page)



Microsoft Corporation

Account
Thinkst Applied Research - B013FVYS

Bing Ads will be down for maintenance between Saturday, July 18, 2015 7:00 AM (GMT-7) and Saturday, July 18, 2015 12:00 PM (GMT-7). You will not be able to sign in, but campaigns will continue to run. Learn more

All Campaigns

Campaign: Campaign #1

7/1/2015 - 7/17/2015

Status: Enabled Budget: 5.00/day Locations: All locations

		Keyword	Ad group	Delivery	Bid	Clicks	Impr.	CTR	Avg. CPC	Spend
☐	●	61735ef62cf584aa378d238144444347	Ad group #1	Eligible	3.13	0	0	0.00%	0.00	0.00
		Search total				0	0	0.00%	0.00	0.00
		Content total				0	0	0.00%	0.00	0.00
		Deleted items total				0	0	0.00%	0.00	0.00
		Overall total - 1 keywords				0	0	0.00%	0.00	0.00

Show rows: 50

Only clicks are MRC accredited.

© 2015 Microsoft Legal Privacy & Cookies Advertise Developers Support Blog Feedback Microsoft

Bring Back Honeypots by the guys from Thinkst (Black Hat USA 2015)

Adding RE-Canaries to your Software

RE-Canaries and your Software

- More than one canary, different modules
  - You want to bait your adversary more than once

- Components
  - Security/protection/licensing
  - Secret sauce (your IP)
  - Places where they don’t stick out too much

- Don’t obfuscate your canaries if that makes them look super special
  - If the rest of your strings are plain text your canaries should be plain text too
  - People are lazy but not stupid

RE-Canaries and your Software Lifecycle

- Every release with a fresh set of canaries → detect what version is analyzed
  - Could be easy to diff, depending on the obfuscation
- Provide individual builds to customers
  - Individual canary → who of them loves your software just a bit too much
- Don’t change the canary at download (don’t binary patch app on the way out)
  - Will break integrity checks (such as file hash) also easy to find and diff
- Automate
  - Record canaries information at build time and “deploy” them at a later time
  - Tasks likely done by different teams

Software Already in the Field?

- You know your software (you have the source!)
- Determine what could make a good canary
- If you have any kind of obfuscation or hardening don’t forget to pick a canary inside the obfuscated parts
- Look at your software with a disassembler!

Dead Canaries

- Canary that constantly triggers
  - Was to easy to find (try to avoid this in the first place!)
  - Collision with something that is not a canary

- Just kill it!
  - Don’t alert, just ignore it → your ops people will thank you

Further Ideas and Notes...

Avoiding Detection via RE-Canaries *(attacker's view)*

- Hiding your actions is pretty hard to impossible
  - ~~Don't search~~, only use specific search engine, only look at results and don't visit result page
  - Non web-based tokens make this hard (e.g., API-endpoint canary or credential canary)

- Hide origin of search (VPN, Tor, etc..)
  - Easy, likely done but you can get lucky

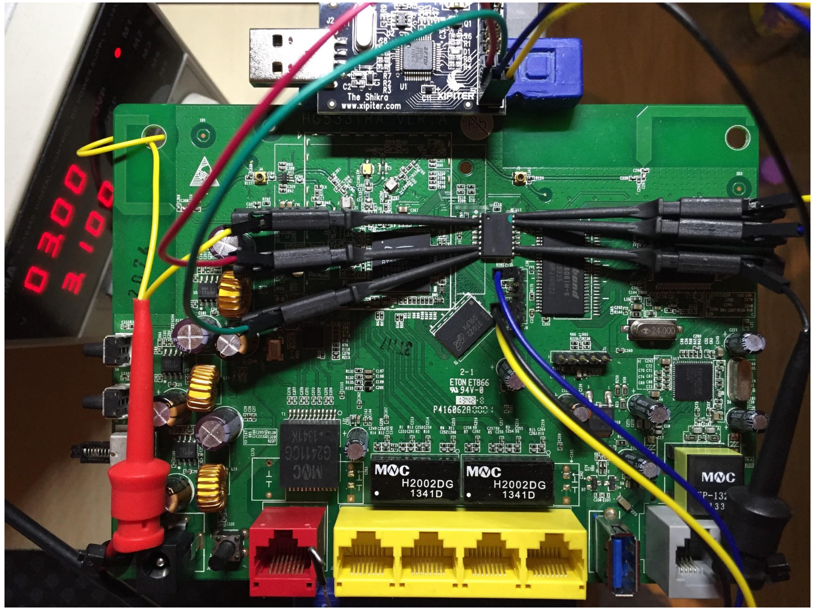
Building RE-Canaries for 3rd-Party Software

- You probably use and rely on software made by someone else
- You would be negatively impacted if that software is targeted
  - I does have bugs! But not necessarily known bugs!
- You could create 3rd-party RE-canaries for that software to get notified if somebody is looking at it very closely
  - You will need to find interesting unique strings in that software
- You should probably search for those strings before creating a token website

Firmware and RE-Canaries

- Extracting or dumping the firmware often is the hardest part!

- No need to obfuscate!
  - Just add some “unique strings”



http://jcjc-dev.com/2016/06/08/reversing-huawei-4-dumping-flash/

Security Nihilism

[1]

Ve believe zat nothing...
ist secure enough vor ze real world, Lebowski!

[1] Flickr user Joe Goldberg CC BY-SA 2.0

YAHOO!

18

Building Security at Scale - Yahoo (Alex Stamos, 2014)

Security Nihilism

- This is just a cute trick and not a security solution!

“Non-obvious protections can increase the chance of catching an attacker in time.” --Alex Stamos (2014)

- Adversaries know about this technique and will not fall for it!
    - Do they really?
- So you are saying your adversary is just going to stop searching the web?

  😂 😂 😂

“Forcing an adversary to expend resources and risk detection is a valid goal.” --Alex Stamos (2014)

Summary and Conclusions

- RE-Canaries can help you to keep up with your adversaries
  - No rocket science, this is easy stuff!
- Universal application
  - Desktop apps, mobile, embedded software and firmware
- Canaries provide information, they don’t add protection or security
  - You have to have a plan what to do when a canary is triggered
- Help to level playing field: brings the attacker to the environment you control
  - Inverts the Attacker-Defender Problem (you just need to get lucky once!)

Thanks & Acknowledgements

Haroon Meer and the guys at Thinkst (creators of http://canarytokens.org)

Ben Nell, Chris Rohlf, and Haroon Meer for various interesting discussions

Q & A

Thank you!

Collin Mulliner

www.3BLabs.com

References

Canarytokens by Thinkst: https://canarytokens.org

Bring back the Honeypots: http://thinkst.com/stuff/bh2015/thinkst_BH_2015_notes.pdf

Building Security at Scale: https://www.slideshare.net/astamos/security-at-scale-lessons-from-six-months-at-yahoo

String deobfuscation: https://www.fireeye.com/blog/threat-research/2016/06/automatically-extracting-obfuscated-strings.html

RE Canaries: http://www.mulliner.org/blog/blosxom.cgi/security/re_canary.html

Related:

A Bodyguard of Lies: The Use of Honey Objects in Information Security: http://www.arijuels.com/wp-content/uploads/2013/09/SACMATabstract.pdf

Honey Encryption: Security Beyond the Brute-Force Bound: https://eprint.iacr.org/2014/155.pdf

Honeywords: Making Password-Cracking Detectable: http://www.arijuels.com/wp-content/uploads/2013/09/JR13.pdf