Sandboxing is (the) shit !

Jonathan Brossard (Toucan System)

Who am I ?

- Security researcher, publishing since 2005.

- Past research : vulnerabilities in BIOSes, Microsoft Bitlocker, Truecrypt, McAfee Endpoint (Defcon 2008), PMCMA debugger (Blackhat USA 2011), « Rakshasa » supply chain backdoor PoC (Blackhat 2012), 2 SAP notes (2013).

- Speaker/trainer at HITB, CCC, Ruxcon...

- Co-founder of the Hackito Ergo Sum and NoSuchCon research conferences (France).

Disclaimer : contains research

This was supposed to be a short research on finding/exploiting a few cool low level bugs in sandboxes.

It ended up leading to more questions than answers on my understanding of what the industry is doing in the AV/sandbox space.

If you have better understanding, I'd really like if you took the time to explain me

(endrazine@gmail.com,+PGP).

Disclaimer (rephrased)

WTF is the AV industry doing ? Well, I'm not so sure I understand anymore ...

What's hot in the AV industry in 2013 ?

AV industry : 2013 trends

- Desktop AV is essentially a thing of the past

- Focus moves technologies hopefully able to
« detect 0days »[1] like <u>sandboxing</u>.

=> The new cool thing is emulation and sandboxing.

[1] Don't laugh yet.

How it all started... (/story telling)

CVE-2013-0640
(Adobe Sandbox bypass)

Adobe Reader and Acrobat 9.x before 9.5.4, 10.x before 10.1.6, and 11.x before 11.0.02 allow remote attackers to execute arbitrary code or cause a denial of service (memory corruption) via a crafted PDF document, as exploited in the wild in February 2013.

CVE-2013-0640
(Adobe Sandbox bypass)

Their « analysis » :

Here is the sequence of the ROP shellcode:

msvcr100!fsopen()

msvcr100!write()

mvvcr100!fclose()

kernel32!LoadLibraryA()

kernel32!Sleep()

Upon loading the malicious library, it will enter a long sleep and ensure that the thread has not crashed because the whole stack in the thread is already manipulated for creating a ROP chain.

Their « analysis » :

Here is the sequence of the ROP shellcode:

msvcr100!fsopen()

msvcr100!write()

mvvcr100!fclose()

kernel32!LoadLibraryA()

kernel32!Sleep()

Upon loading the malicious library, <u>it will enter a long sleep and ensure that the thread has not crashed because the whole stack in the thread is already manipulated for creating a ROP chain.</u>

Their « analysis » :

Here is the sequence of the ROP shellcode:

msvcr100!fsopen()

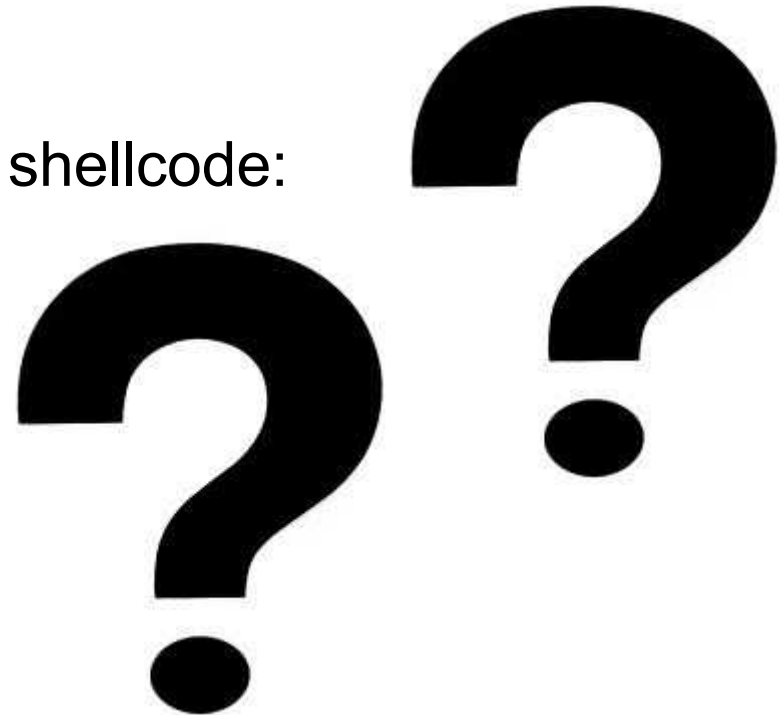msvcr100!write()

mvvcr100!fclose()

kernel32!LoadLibraryA()

kernel32!Sleep()

Upon loading the malicious library, <u>it will enter a long sleep and ensure that the thread has not crashed because the whole stack in the thread is already manipulated for creating a ROP chain.</u>

=> In trivial english, this is called bullshitting. They clearly have no idea what the exploit is trying to do here.

What I believe really happens in this case (wild guess)

Sleep 5 minutes to attempt bypass sanboxing detection :)

After all, it's a hardened exploit, found in the wild and the first of its kind to bypass Adobe sandboxing technology...

Limits of such technologies (imho)

- Good at finding artefacts (it's still « something »).

- Pretty bad at understanding what is actually happening inside the exploit.

That being said...

The raise of sandboxes...

The raise of sandboxes...

The raise of sandboxes...

The raise of sandboxes...

Note to self : I don't find quite reasonable to add to your corporate network something nobody really understands.

Note : lack of third party assessment

Jonathan Brossard
@endrazine

Dear @FireEye , is there a chance you will let
independant researchers look at your
appliance ? Lack of transparency=not good :)
/cc @taviso

← Reply   🗑 Delete   ★ Favorite   ••• More

Note : lack of third party assessment

the grugq
@thegrugq
Following

@endrazine @HaifeiLi @taviso surprise plays a part in security, if the adversary cannot examine your defences before encountering them...

Reply Retweet Favorite More

Note : lack of third party assessment

the grugq
@thegrugq
Following

@HaifeiLi @endrazine @taviso if you want to remain effective as a security vendor, having working tech basically requires keeping it secret.

Reply   Retweet   Favorite   More

Note : lack of third party assessment

The whole concept of sandboxing vendors is to not have the perceived enemy take a look at the technology. Ok, agreed.

Note : lack of third party assessment

It also means no third party assessment has been done by the security community...

In real life, having software due dilligence done by the community has proved to be a good thing for the quality of the said software.

See similar requests from Tavis Ormandi and Pipacs to have a look at Bromium's technology...

Note : well, they're not Bromium clients, so we have a problem... as an industry, really.

Note 2 : Afaik, Bromium has researchers like Nergal and Jarred Demott. Who of this caliber works for FireEye really ?

grsecurity
@grsecurity

A petition to get a Bromium demo for @taviso and Pipacs. Pipacs has agreed to use a 10 year old copy of IDA to handicap his breaking it.

← Reply   ↻ Retweet   ★ Favorited   ••• More

31
RETWEETS

6
FAVORITES

8:40 PM - 6 Aug 13

Reply to @grsecurity @taviso

Kostya Kortchinsky @crypt0ad   6 Aug
@grsecurity @taviso can I play too??
Details

Room for problems (research leads)

Room for problems part I : general design/architecture

What's the trend, perceived objective like

Current « genious » idea :

- Correlate/share more data to create information asymetry.

- To do that, most (all I've seen allow it, at least in non default mode) solutions now <u>allow a malware to connect back to the internet[*]</u>.

[*] Idea being to correlate DNS/binary checksum informations over « campaigns » of attacks in the time.

A few facts on this...

- The whole corporate strategy over the past 15 years has been to segregate LANS, DMZs and the internet.

- Now you give a temporary shell to the attacker at network perimeter (proxies, mail gateways, wherever such sandboxing solutions exist)...

- … and what happens to your DNS ? To your http proxy cache ?

The Katsuni-Kaminsky attack (having it both ways)

- Attacker can run a malware inside a sandbox.

- Sandbox allows attacker to connect back to the internet.

- Corporate DNS server is used as a recursive DNS server.

- Attacker has it both ways and can synchronize arbitrary spoofed packets emission from both inside and outside the network.

=> That's gonna be very « safe » for sure...

Not to mention...

- What happens if the malware, from inside the sandbox manages to attack other networks (say crowdstrike !) ?

- What happens if the malware can send back a modified copy of itself to the same network (smtp?) for more analysis, and more sandbox cpu time ?

=> It's all about implementation details really.

Is this « wormable » (yet) ?

- FireEye claims to work with 30 % of the TOP 100 Companies. That makes it not so hard to find...

- Their strategy is to synchronize malware information sharing... (allow exploits to drop exes on the sandboxes... ?!?)

- They cover you « 360 », from mail gateways to http proxy file downloads, etc.

- Ok, so how exactly do you prevent one malware to get endless free execution time inside your different sandboxes around the world ?

Room for problems part II : Turning lame bugs into sandbox Oracles

The problem

- Many online malware scanning engines run qemu (+ some various instrumentation and automation custom software)

- User doesn't get to see anything from the scanning process.

- Now, what if an attacker has a qemu lame DoS or endless loop PoC ?

Turning lame bugs into sandbox Oracles

(aka : hacking online malware analysis tools... hrm)

Room for problems part III : Such bugs do happen...

What degrees do projects like Xen or qemu really have in terms of security ?

A fair question is the relative maturity of such technology, not when it comes to support legacy Oses or current Oses, but hostile malware trying to hurt them.

Exempli gratia : typical bugs reported (complexity, software maturity : format strings/symlinks or complex overflows ?)

xen-unstable.hg

log

log
graph
tags
branches
changeset
browse

age	author	revision	description
18 hours ago	Olaf Hering	27594:2c8d4cf6d99f	unmodified_drivers: enable unplug per default default tip
18 hours ago	Matthew Daley	27593:c77b00504c33	gnttab: remove unused shared header lookup
18 hours ago	Dario Faggioli	27592:f47399ef59f8	sched_credit: filter node-affinity mask against online cpus
19 hours ago	Jan Beulich	27591:f6e6813de0f0	x86_emulate: fold wide reads
19 hours ago	Jan Beulich	27590:f14569c8d9f2	x86_emulate: fix flag setting for 8-bit signed multiplication
19 hours ago	Jan Beulich	27589:b9147356be4f	x86_emulate: PUSH <mem> must read source operand just once
19 hours ago	Jan Beulich	27588:e5a327a1adf6	x86_emulate: MOVSXD must read source operand just once
19 hours ago	Jan Beulich	27587:fe2fc2d6f4af	x86: fix dependencies of emulator test
19 hours ago	Jan Beulich	27586:d99cbeae68e1	x86/HVM: properly handle MMIO reads and writes wider than a machine word
37 hours ago	Ian Campbell	27585:1ca80f00207c	tools: print xm deprecation warning to stderr
37 hours ago	Tim Deegan	27584:ec815275b09a	x86: mark BUG()s and assertion failures as terminal.
2 days ago	George Dunlap	27583:e40e0d8ad3f4	VMX: fix failure path in construct_vmcs
2 days ago	George Dunlap	27582:620e3224306f	x86/HVM: fix failure path in hvm_vcpu_initialise
3 days ago	Ian Campbell	27581:aad17180d4d0	tools: disable xend build by default
3 days ago	Ian Campbell	27580:c59fb63abd4b	docs: fix documentation index for hypercalls
3 days ago	Tim Deegan	27579:6f2d9ab0a018	passthrough/amd: Shuffle declaration.
3 days ago	Tim Deegan	27578:a892aea40e0f	passthrough/amd: Missing 'break'
3 days ago	Tim Deegan	27577:49c74a84158b	passthrough/amd: Drop unnecessary lock lookup.
3 days ago	Andre Przywara	27576:5f57bd039986	ARM: parse separate DT properties for different commandlines
3 days ago	Julien Grall	27575:08af4f786b8d	xen/arm: Check if the device is available before using it
3 days ago	Julien Grall	27574:3d1bc43bfdef	xen/dts: replace get_val by dt_next_cell
3 days ago	Julien Grall	27573:5fdd1eed65c7	xen/dts: device_get_reg: cells are 32-bit big endian value
3 days ago	Julien Grall	27572:202bc504049d	xen/dts: Clean up the exported API for device tree
3 days ago	Julien Grall	27571:400e098bb92b	xen/arm: exynos5: Blacklist MCT device
3 days ago	Julien Grall	27570:83c96cea43b5	xen/arm: vexpress: Blacklist a list of board specific devices
3 days ago	Julien Grall	27569:2b027af299c1	xen/arm: Remove devices used by Xen from dom0 device tree
3 days ago	Julien Grall	27568:bdc3d187f55e	xen/arm: Add new platform specific callback device_is_blacklist
3 days ago	Julien Grall	27567:4dc18d1dd8ea	xen/arm: Create a fake timer node in dom0 device tree
3 days ago	Julien Grall	27566:089ee397ca5f	xen/arm: Create a fake GIC node in dom0 device tree
3 days ago	Julien Grall	27565:c4bf014cdf7f	xen/arm: Create a fake cpus node in dom0 device tree
3 days ago	Julien Grall	27564:91f495f44063	xen/arm: Create a fake PSCI node in dom0 device tree
3 days ago	Julien Grall	27563:344668a7901a	xen/arm: Don't map disabled device in DOM0
3 days ago	Julien Grall	27562:d598a6f942bf	xen/arm: Build DOM0 FDT by browsing the device tree structure
3 days ago	Julien Grall	27561:83d8c6627844	xen/dts: dt_find_interrupt_controller: accept multiple compatible strings
3 days ago	Julien Grall	27560:2dfc1641fe3e	xen/arm: Use dt_device_match to avoid multiple if conditions
3 days ago	Julien Grall	27559:bcad6b37c262	xen/video: hdlcd: Use early_printk instead of printk
3 days ago	Julien Grall	27558:1c1b9d0708e0	xen/video: hdlcd: Convert the driver to the new device tree API
3 days ago	Julien Grall	27557:afb84d62af5d	xen/dts: Check the CPU ID is not greater than NR_CPUS
3 days ago	Julien Grall	27556:191527726eae	xen/dts: Check "reg" property length in process_multiboot_node
3 days ago	Julien Grall	27555:af61f9f00abc	xen/dts: Remove device_get_reg call in process_cpu_node
3 days ago	Julien Grall	27554:f7c0208c457c	xen: Use the right string comparison function in device tree
3 days ago	Julien Grall	27553:58fcf834d122	xen/dts: Add new helpers to use the device tree
3 days ago	Julien Grall	27552:fa2d8d29bb4d	xen/dts: Don't add a fake property "name" in the device tree
3 days ago	Julien Grall	27551:fc755ee667bf	xen/dts: Prefix device tree macro by dt_
3 days ago	Julien Grall	27550:9e3cbbd58671	xen/dts: Constify device_tree_flattened

x86_emulate: MOVSXD must read source operand just once (Xen Unstable, 21/09/2013)

```
1.109          if ( (rc != X86EMUL_OKAY) ||
1.110               (regs.eax != 0x0000ffff) ||
1.111               ((regs.eflags&0x240) != 0x200) ||
1.112 -              (regs.eip != (unsigned long)&instr[2]) )
1.113 +              (regs.eip != (unsigned long)&instr[2 + (ctxt.addr_size
1.114          goto fail;
1.115      printf("okay\n");
1.116
1.117 @@ -821,6 +877,8 @@ int main(int argc, char **argv)
1.118          if ( j == 2 ) break;
1.119          memcpy(res, blowfish32_code, sizeof(blowfish32_code));
```

Ok, so you're at miscomputing EIP on basic instructions such as MOVSXD on 32b...

Note : this is totally exploitable imho btw.

How does that rank compared to real cpu bugs ?

Intel® Pentium® Processor
Invalid Instruction Erratum Overview

Invalid Instruction Errata Home   **Software Vendor Statements**

Software Backgrounder   **Erratum Technical Description**
Updated Nov. 20 1997   Updated Nov. 20 1997

Intel Identifies Workaround for the "Invalid Operand with Locked Compare Exchange 8Byte (CMPXCHG8B) Instruction" Erratum

Erratum Overview

On Friday, November 7th 1997, a number of reports were posted to the Internet implying the possibility of a new erratum on the Pentium® processors and Pentium® processors with MMX™ technology. An erratum is a design defect or error which may cause a product to deviate from published specifications. Based on the Internet reports our engineering team quickly jumped on this issue. Once we were able to reproduce the behavior we confirmed that an erratum does exist which is now named the "Invalid Operand with Locked CMPXCHG8B instruction" erratum. We were also able to identify the following:

That was in 1997... The instruction is far less used (CMPXCHG8B)
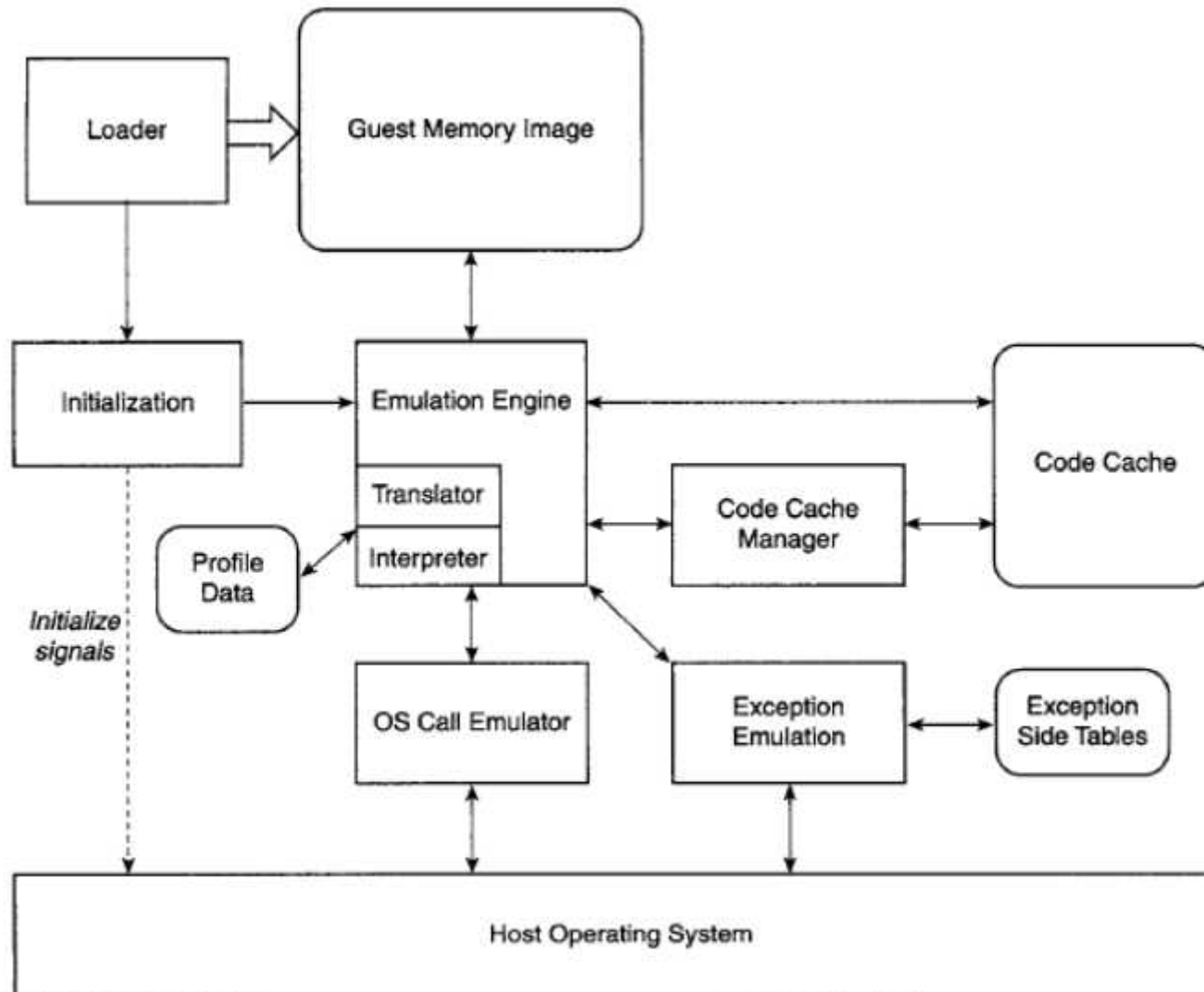
Qemu internals

Qemu goals

- (Fast) binary translation

- Binary code is translated into an IR, which is then executed by a virtual cpu, independently of host OS/arch.

- Super generic, super fast, super portable, super cool. Truely impressive.

- Super secure for the purpose of malware analysis though ?
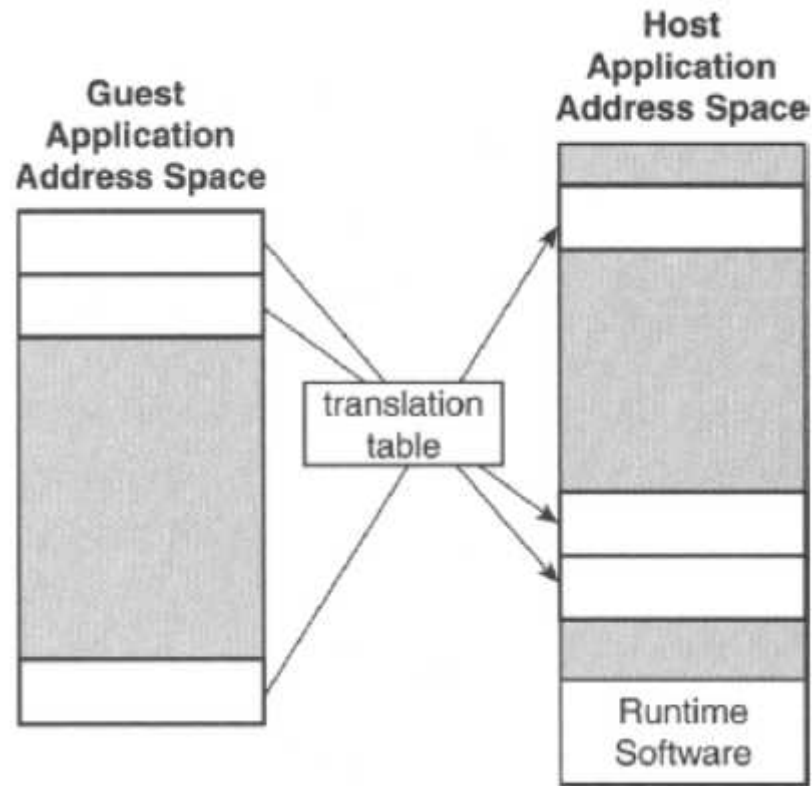
Qemu supports two modes

- System (full) virtualization

- Kernel emulation (wine/Windows, linux).

While many of current implementations are likely to be using the first flavour, we'll focus on the later one, which promises great speed enhancements, and a much greater attack surface... ;)
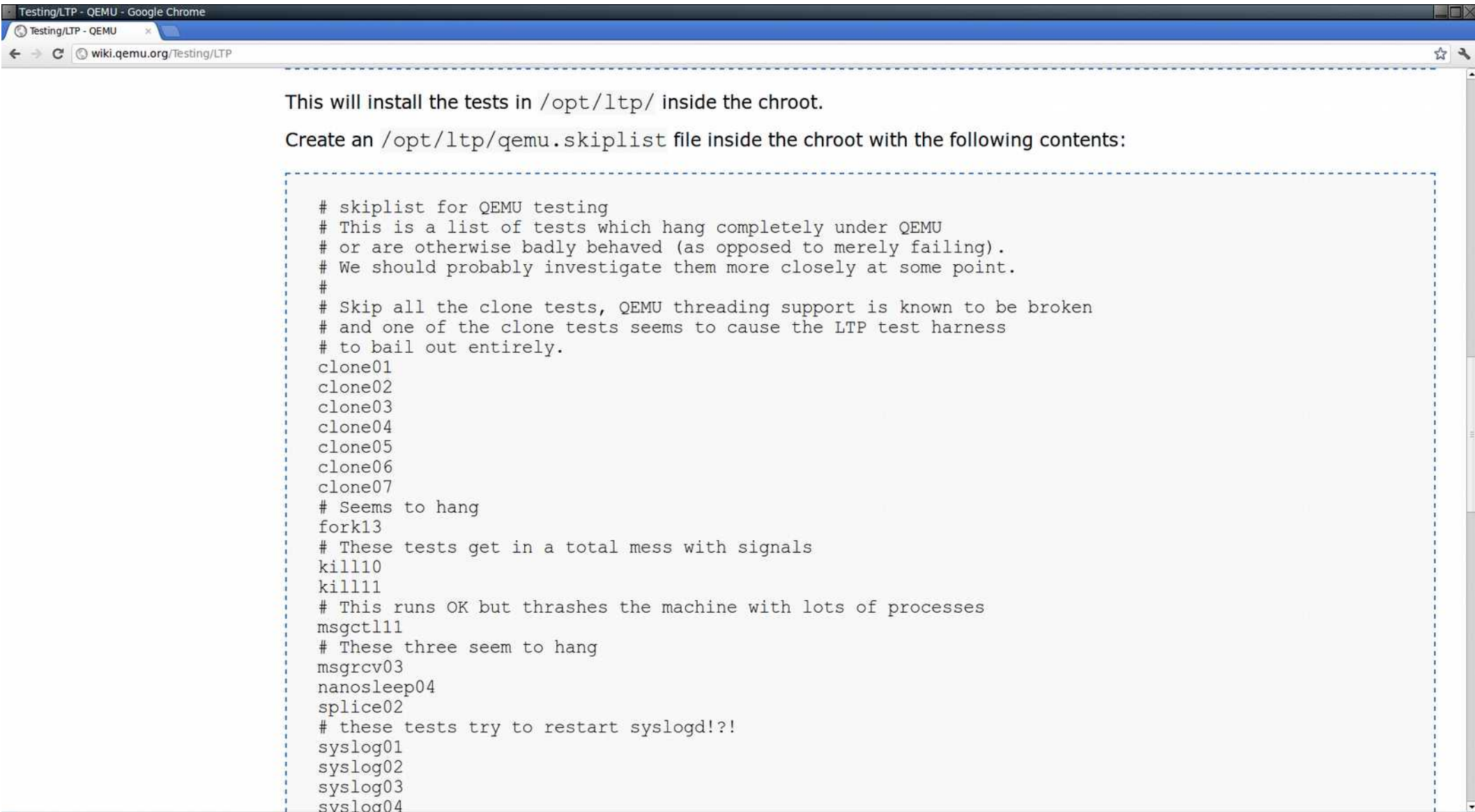
Qemu architecture

Understanding binary translation & memory sharing

Party time !!

Qemu regression tests...

Demos

Conclusion

Interresting technologies. Cool hacking tools, usefull for researchers.

There is room for massive security problems, the devil being in the details. Imho, not ready for Enterprise grade deployment.

No such a thing as third party assessment afaict.

From a strict game theory pov, your best interrest is probably to have <u>others</u> use that, but stay away from such technologies...

The Katsuni-Grothendieck theorem applied to silver bullet 0day sandboxing :

« You can't have it both ways ! »

Thanks for inviting me.

Questions ?