AMD DATACENTER SOLUTIONS

Enhance your Cloud Security with AMD EPYC™ Hardware Memory Encryption

White Paper
October, 2018

Introduction

Consumers and enterprises are becoming increasingly concerned about the security of their digital data, and with just cause. The Common Vulnerabilities and Exposures (CVE) database, which catalogs more than 100,000 known exploits, is now growing by more than a thousand new entries per month. Table 1 below gives the year-by-year totals. [1]

Year	New CVE Entries
2015	6487
2016	6447
2017	7613
2018	**9428** (as of 7/9/2018)

Table 1. New CVE Entries by Year. Note the sharp increase in 2018.
https://nvd.nist.gov/vuln/search

This sense of heightened concern is particularly relevant given the ongoing migration of end user data into the cloud. Information which was safely guarded on-premises may now cross multiple networks, storage systems and geographies. All of which present potential vectors of attack. When discussing strategies to effectively combat these threats, security experts often categorize data into its three different states:

Data In Flight	Data At Rest	Data In Use

Each has unique requirements for protection and are discussed below.

[1] For a truly harrowing experience, follow the live feed for a constant stream of newly discovered issues: https://twitter.com/CVEnew

Data in Flight

Data moving through the network - also called Data in *Motion or Data in Transition.*

Considering the three states of data, perhaps data in flight security is most the widely understood and applied. Critical transactions are commonly routed through public infrastructure or even open WIFI connections. HTTPS web connections have been available for more than 20 years to provide secure transactions, but for many years, adoption was slow. This trend has dramatically reversed in the last few years. Google regularly measures the number of page loads occurring via HTTP vs HTTPS. The year-over-year data for May is shown in Table 2. Note the clear trend toward secure connections.

Date	% of page loads over HTTPS
May 16, 2015	47%
May 14, 2016	60%
May 13, 2017	73%
May 12, 2018	84%

Table 2. HTTPS Encryption by Chrome Platform
https://transparencyreport.google.com/https/overview?hl=en

In addition to the increased use of secure connections, enterprises are using stronger encryption to protect those secure links. SSLlabs regularly conducts a survey of the 150,000 most popular SSL and TLS enabled websites. At the last reporting, 100% use the equivalent of RSA 2048 or above, with a growing number using the equivalent of RSA 4096 and above.

Date	< 2048	2048	3072	4096
April 22, 2012	16.7%	81.3%	0.1%	2.0%
April 9, 2013	6.7%	91.1%	0.1%	2.2%
April 5, 2014	0.6%	96.8%	0.1%	2.5%
April 4, 2015	0.1%	95.7%	0.0%	3.5%
April 5, 2016	0.1%	93.8%	0.0%	4.4%
April 2, 2017	0.0%	92.8%	0.1%	4.9%
April 3, 2018	0.0%	91.7%	0.0%	5.5%

Table 3. Key Strength Distribution
https://www.ssllabs.com/ssl-pulse/

Data at Rest

Data in persistent storage (disk).

Once the data has been securely transmitted through the network, it must be properly secured against unauthorized access within the datacenter. Storage drives provide a very high value target for potential data thieves. Data at rest protection requires securing the contents of the data as they reside on the disk. Many different technologies exist to perform this function, but they commonly fit into the three categories shown below.

Category	Definition	Common Examples
Individual File Encryption	Individual sensitive files are encrypted by the end user	VeraCrypt, AxCrypt, GNU PG, 7-Zip,
Filesystem Encryption	Files and folders are encrypted by the filesystem. Software drivers perform the encryption	NTFS with EFS (Windows), eCryptfs (Linux), PEFS (FreeBSD)
Disk Encryption	The entire disk is encrypted. Software drivers or dedicated hardware perform the encryption	Bitlocker (Windows), dm-crypt (Linux)

Table 4. Data at Rest Encryption

Regardless of the implementation, at some point the data will be decrypted for use. This requires a suitable key managed with appropriate user access control. Secure key stores such as a TPM can provide hardened storage for key management.

Data at rest protection in the cloud

Data protection can be a tricky challenge for the cloud tenant who wants to defend data stored in a third-party datacenter. A cloud VM can be configured to use software-based disk encryption such as bitlocker or dm-crypt to protect a virtual drive. When protecting a boot drive, the end user provides a password or pin at boot time. A data drive might require a password on first use. For complete security, all these password exchanges and their associated memory need to be hidden from the datacenter operator[2].

[2] See the whitepaper "Solving the Cloud Trust Problem with WinMagic and AMD EPYC Hardware Memory Encryption".

Data in Use

Data in memory

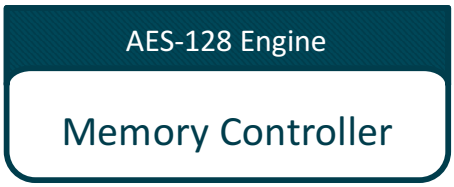
Even in the most secure data at rest scenario, at some point the data will be decrypted and loaded into memory. Unfortunately, much like how unsecured WIFI is accessible to all users in radio range, most data in memory is completely available to datacenter administrators[3], successful hackers, or intruders with physical access to the machine

AMD Hardware Memory Encryption

Much like the cryptographic solutions offered for **Data in Flight** and **Data at Rest**, AMD EPYC Hardware Memory Encryption provides encryption for **Data in Use**. AMD EPYC™ is an SOC microprocessor which introduces two new hardware security components

1) **AES-128 hardware encryption engine**

   This is embedded in the memory controller. When a key is provided, data written to main memory is encrypted. Data read from main memory is decrypted. Because memory controllers are inside the EPYC SOC, the memory data lines that leave the processor contain encrypted data.



2) **AMD Secure Processor**

   This dedicated security processor provides cryptographic functionality for secure key generation and key management.

AMD EPYC processors combine these components to offer two new security features:

- **Secure Memory Encryption (SME), and**
- **Secure Encrypted Virtualization (SEV)**

Both provide encryption for Data in Use and require no application changes for the end user.

[3] Appendix 1 shows a command line based attack that can be run by an administrator on the host hypervisor system. In the example, the rogue admin develops a single command line to monitor and steal credit card numbers directly from the target VM memory.

AMD Secure Memory Encryption (SME) – Single Key Protection

AMD Secure Memory Encryption uses a single key to encrypt all of system memory. The key is generated by the AMD Secure Processor at boot. An attacker with physical access may try to perform a cold boot attack, which involves freezing and then stealing DIMMs directly from a machine[4]. With SME enabled, the attacker won't be able to read the encrypted data.
SME requires enablement in the system BIOS or operating system. When enabled in the BIOS, memory encryption is transparent and can be run with any operating system.

AMD Secure Encrypted Virtualization (SEV) – Many Key Protection

AMD Secure Encrypted Virtualization uses keys to cryptographically isolate individual virtual machines and the hypervisor from one another. The keys are managed by the AMD Secure Processor. An attacker with hypervisor administrator access or a compromised VM account may try to read the memory of other virtual machines. With SEV, the attacker sees only encrypted data. SEV requires enablement in the guest and hypervisor operating system. The guest changes allow the VM to indicate which pages in memory should be encrypted. The hypervisor changes use hardware virtualization instructions and communication with the AMD Secure processor to securely load the appropriate keys in the memory controller.

	Secure Memory Encryption	**Secure Encrypted Virtualization**
Feature	Single key encryption	One key per VM/Hypervisor
Key(s)	Generated at Boot	Managed by AMD Secure Processor and Hypervisor
Enabled by	BIOS or Operating System	Hypervisor and Guest OS
Application Changes Required?	None	None

Table 5. AMD EPYC Hardware Memory Encryption

[4] Princeton researchers demonstrate cold boot attacks: https://www.youtube.com/watch?v=Ej-Nr79bVjg

Implementation and Conclusion

The enablement required to support AMD EPYC Hardware Memory encryption has accepted in to the Linux kernel versions 4.14 – 4.16

- 4.14 – Secure Memory Encryption (only required when not enabled in BIOS)
- 4.15 – Guest VM - AMD Secure Encrypted Virtualization
- 4.16 – Host Hypervisor - AMD Secure Encrypted Virtualization.

Visit the links below for more detail including system setup instructions

- https://developer.amd.com/amd-secure-memory-encryption-sme-amd-secure-encrypted-virtualization-sev/
- https://github.com/AMDESE/AMDSEV

By providing fast and stable Data-In-Use encryption capability, AMD EPYC Hardware Memory Encryption helps users compliment their Data-in-Flight and Data-at-Rest strategies to provide advanced protection for their valuable data.

Appendix 1. Administrator scraping memory of guest VM

The steps below show an administrator viewing guest memory on a Linux based system.

1) List all processes running on the system to identify the target VM (here 40057).

```
> ps ax
   PID     COMMAND
 40057     qemu-system-x86_64 -enable-kvm -cpu EPYC -smp 4,max …
     …
```

2) List memory regions allocated to the target VM. The guest VM memory is identified by its 2GB size.

```
> cat /proc/40057/maps
7f5a4fe00000-7f5acfe00000 …
…
```

3) Calculate the page offset corresponding to the VM memory region. Assume **4096** bytes per page.

7f5a4fe00000 (hex) = 140,025,863,864,320 (decimal)

140,025,863,864,320 (bytes) / 4,096 (bytes per page) = **34,186,001,920** (pages)

4) Read **1** page of memory.

```
> dd if=/proc/40057/mem bs=4096 skip=34186001920 count=1 | hd
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000040  00 02 00 c0 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00001000
```

5) Search 2GB (**524288 pages**) of memory for desired secrets. Note the credit card numbers reported below.

```
> dd if=/proc/40057/mem of=fifo bs=4096 skip=34186001920
count=524288 & grep -a -o -b credit_card.\{0,20\}' fifo
1543313168:credit_card=4111111111111111
1828370416:credit_card=5500000000000004
```

DISCLAIMER

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of non-infringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale.