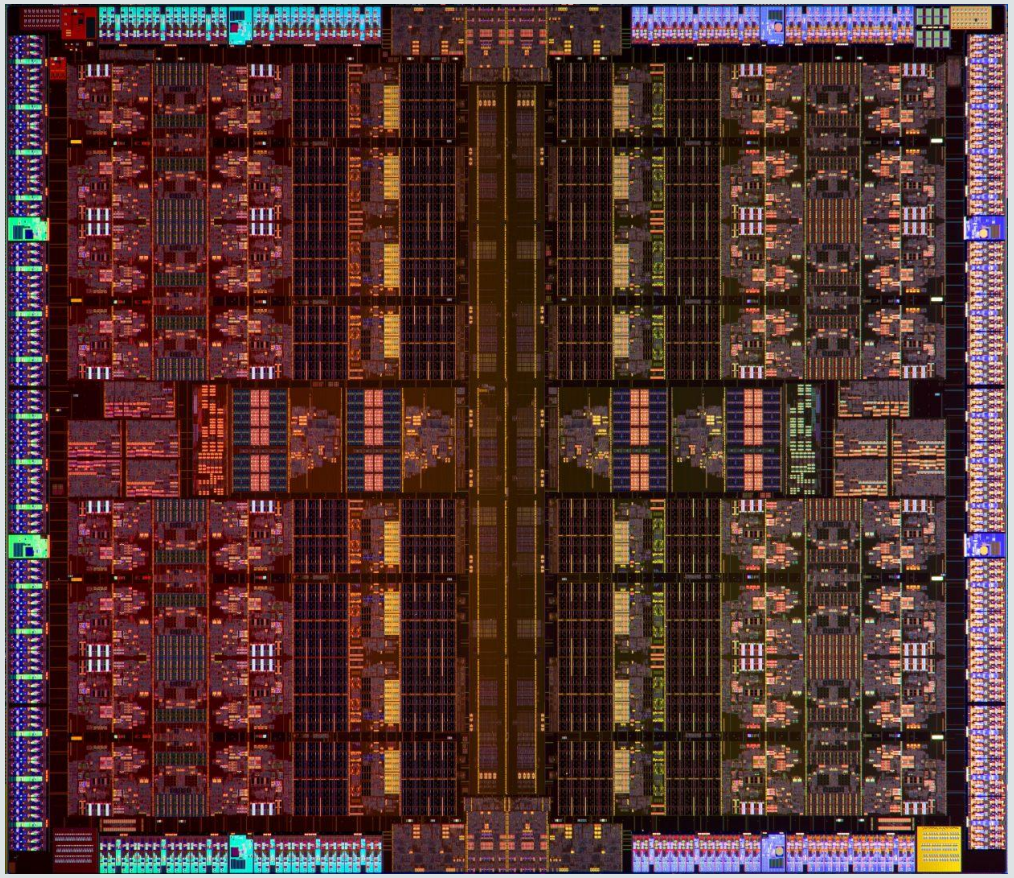

ORACLE®

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Mit Silicon Secured Memory Heartbleed und Co. vorbeugen

Franz Haberhauer
Chief Technologist
Systems Sales Consulting Northern Europe

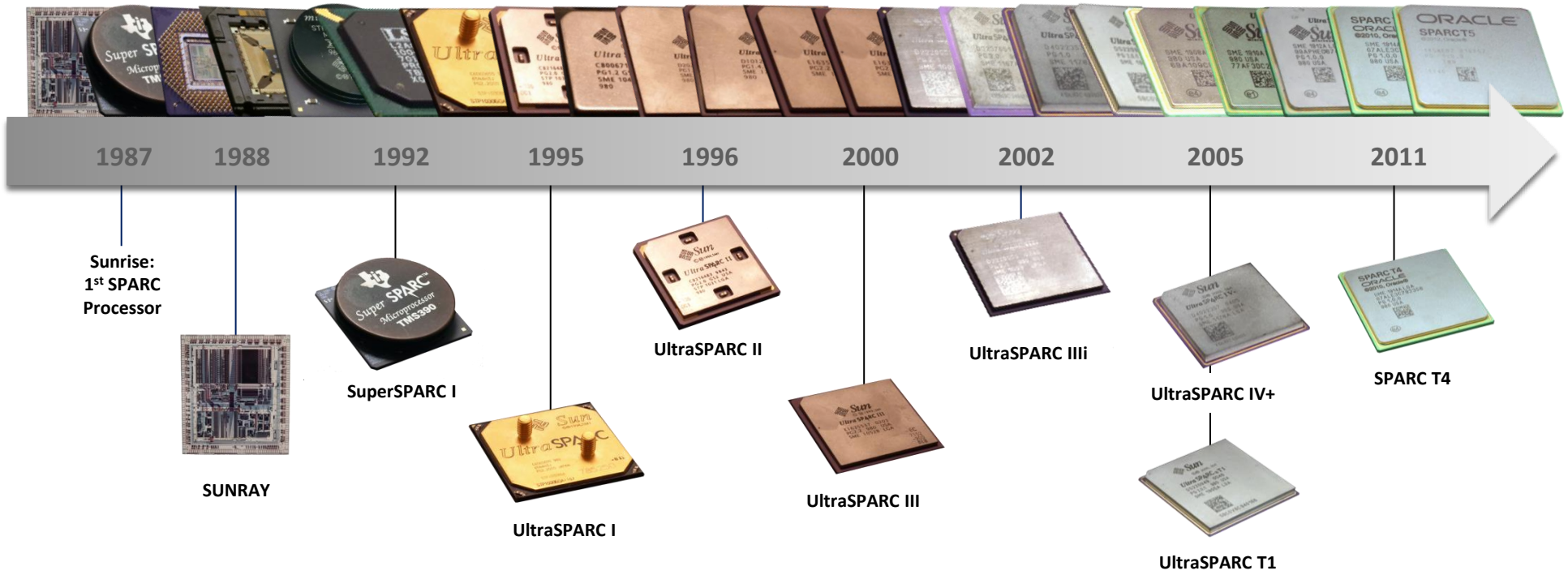


ORACLE

Chip Advances in the Last Decade

- Focus on better/faster general purpose chip
- More CPU cores per chip
- Memory & PCI interfaces, GPU moved on-chip
- Improved pipelines, branch prediction, cache coherency, reliability, clock rates, power management etc.
- **New Functionality**: vector processing/SIMD, virtualization, **encryption**
  - Encryption on-chip is 10X faster and frees CPU cores to do other work
  - Database optimizations on chip are analogous

2012 – 25 Years of SPARC Processors



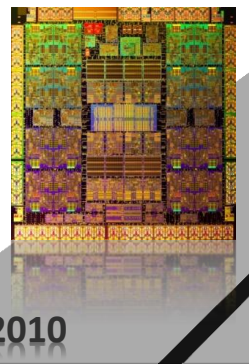
Anniversary Video: http://www.youtube.com/watch?v=IKB9zV8TXuQ
Infographic: http://www.oracle-downloads.com/sparc25info/

SPARC @ Oracle

7 Processors in 6 Years

Including Software in Silicon

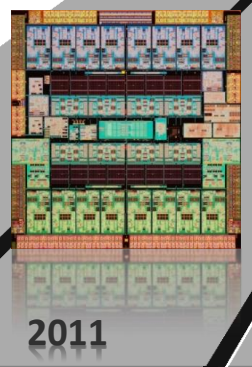
- Silicon Secured Memory
- DB Query Acceleration
- Inline Decompression
- More….



2010

SPARC T3

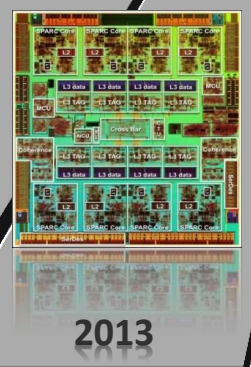
16 x 2nd Gen cores
4MB L3 Cache
1.65 GHz



2011

SPARC T4

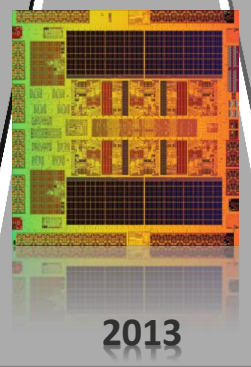
8 x 3rd Gen Cores
4MB L3 Cache
3.0 GHz



2013

SPARC T5

16 x 3rd Gen Cores
8MB L3 Cache
3.6 GHz



2013

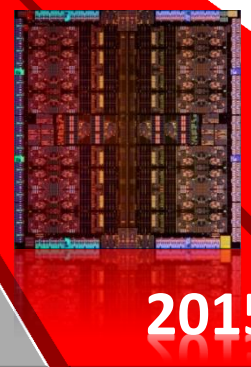
SPARC M5

6 x 3rd Gen Cores
48MB L3 Cache
3.6 GHz

2013

SPARC M6

12 x 3rd Gen Cores
48MB L3 Cache
3.6 GHz

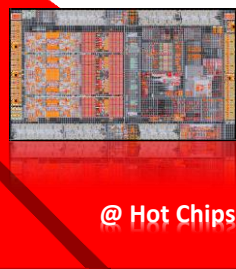


2015

SPARC M7

32 x 4th Gen Cores
64MB L3 Cache
4.1 GHz

Current
T7-/M7-Servers



@ Hot Chips 2015

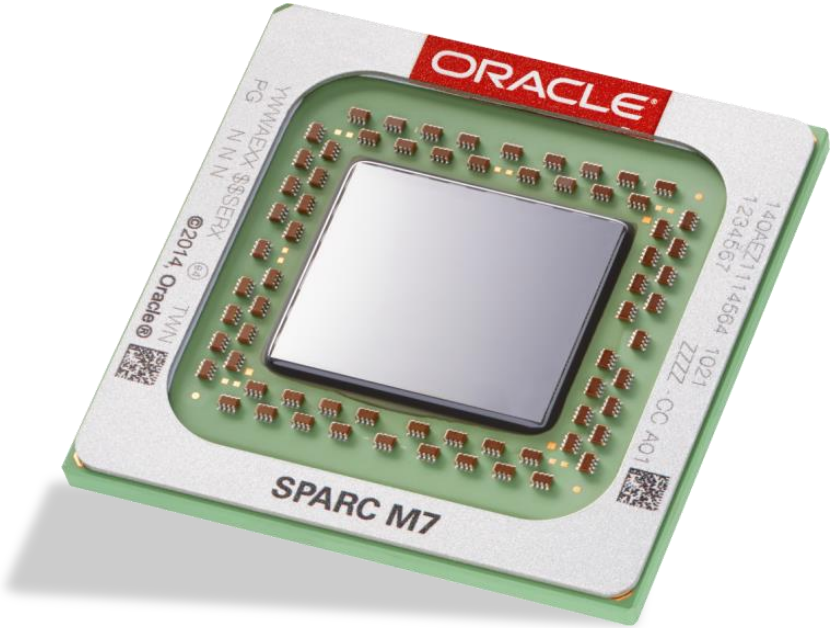
'SONOMA'

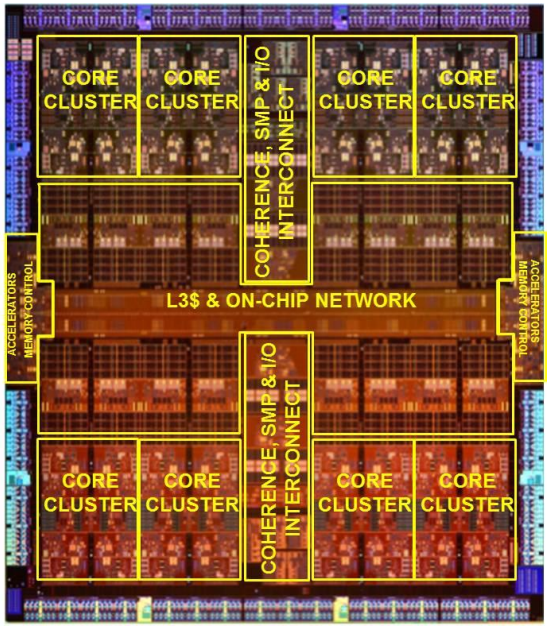
8 x 4th Gen Cores
IB

Scale-Out
Servers TBA.

ORACLE

It Does not Take Much Die to Make a Difference
SPARC M7





2x-3x More Throughput Performance (16 -> 32 Cores) **with** **30 to 40% More Single Thread Performance** **and** **Over 2x More Encryption Bandwidth** **Plus**

Software in Silicon:
Security in Silicon
SQL in Silicon
Capacity in Silicon

< 1% of Die

ORACLE

SPARC T7 and M7 Systems





	T7-1	T7-2	T7-4	M7-8	M7-16
Processors	1	2	2 or 4	Up to 8 [1]	Up to 16 [2]
Max Cores	32	64	128	256	512
Max Threads	256	512	1,024	2,048	4,096
Max Memory [3]	.5 TB	1 TB	2 TB	4 TB	8 TB
Form Factor	2U	3U	5U	Rack / 10U	Rack
Domaining	LDOMs	LDOMs	LDOMs	LDOMs, PDOMs [1]	LDOMs, PDOMs [2]

(1) Factory configured with one (up to 8 processors) or two (up to 4 processors each) static physical domains
(2) 1, 2, 3 or 4 reconfigurable physical domains
(3) Maximum memory capacity is based on 32 GB DIMMs

http://oracle.com/m7infowall -> White Paper M7 Systems Architecture
https://blogs.oracle.com/bestperf

ORACLE

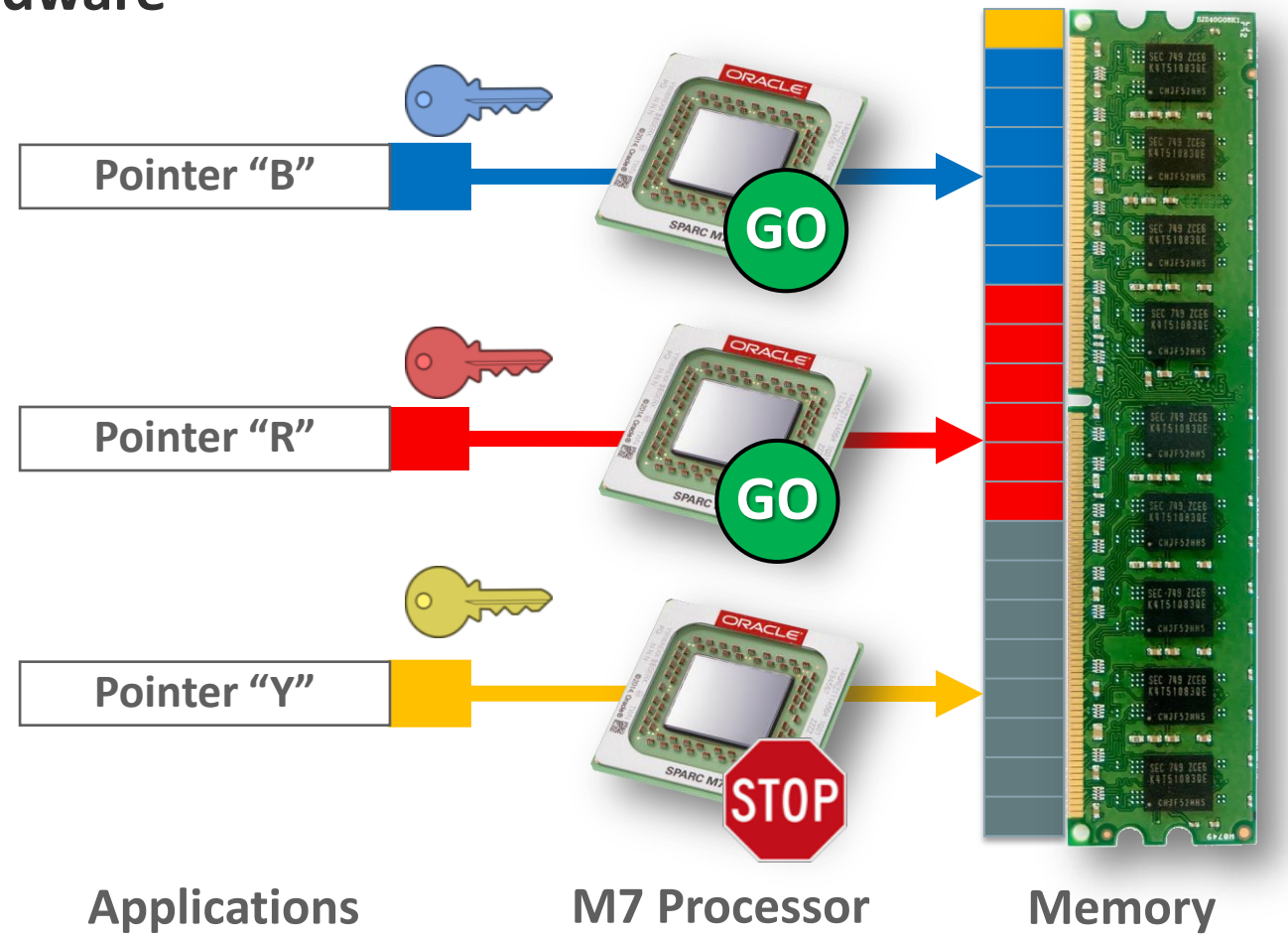
Silicon Secured Memory

Application Data Integrity (ADI)

Oracle M7 Silicon Secured Memory

Always-On Memory Protection in Hardware

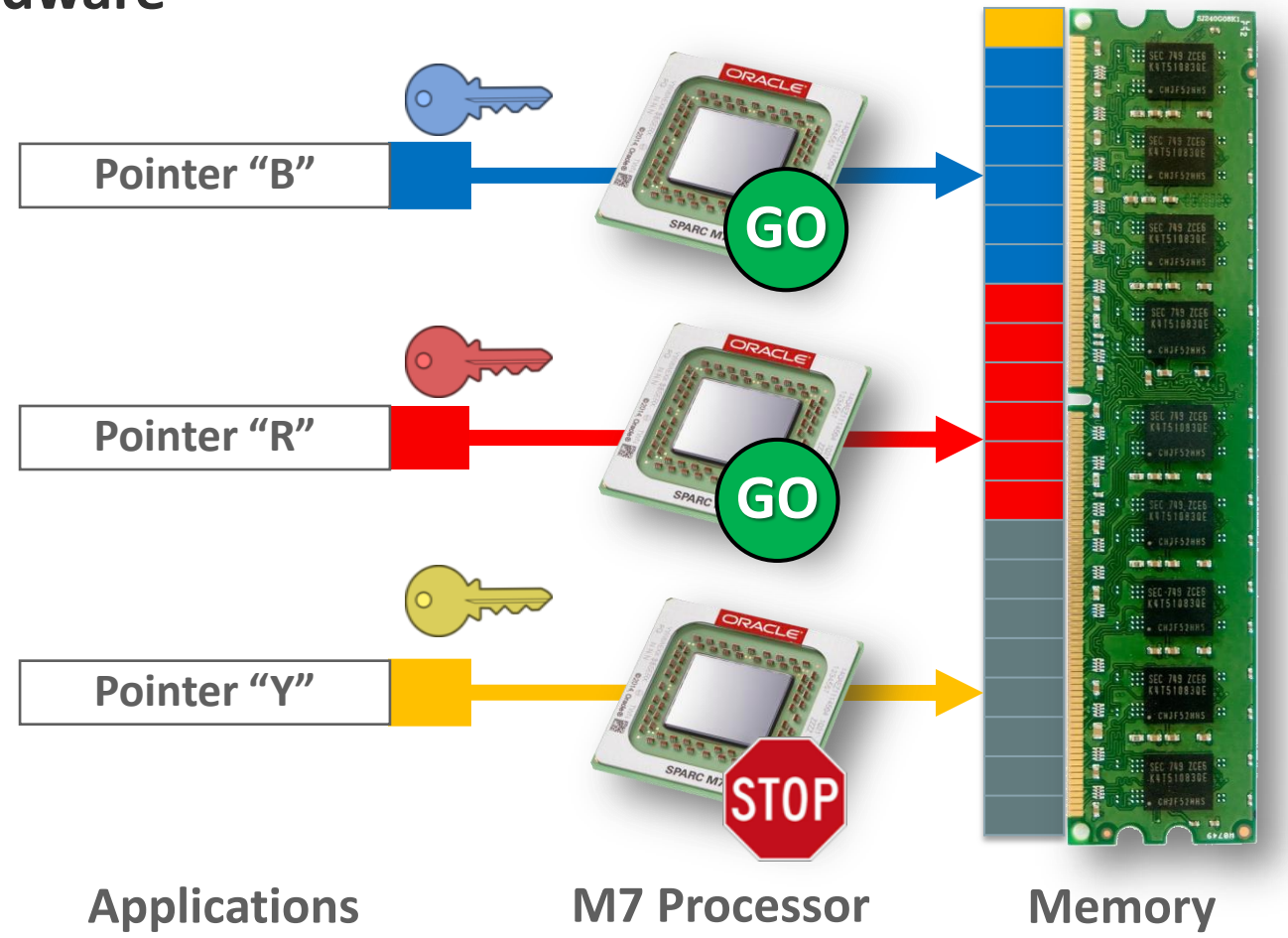
- **Protects data in memory**
- Hidden “color” bits added to *pointers* (key) and content (lock)
- Pointer color (key) must match content color or program is aborted
  - Set on *memory allocation*, changed on *memory free’*
  - Protects against *access off end of structure*, *stale pointer* access and malicious attacks



Oracle M7 Silicon Secured Memory

Always-On Memory Protection in Hardware

- **Protects data in memory**
- Hidden “color” bits added to *pointers* (key), and content (lock)
- Pointer color (key) must match content color or program is aborted
  - Set on *memory allocation*, changed on *memory free’*
  - Protects against *access off end of structure*, *stale pointer* access and malicious attacks
- **Extremely efficient for software development**

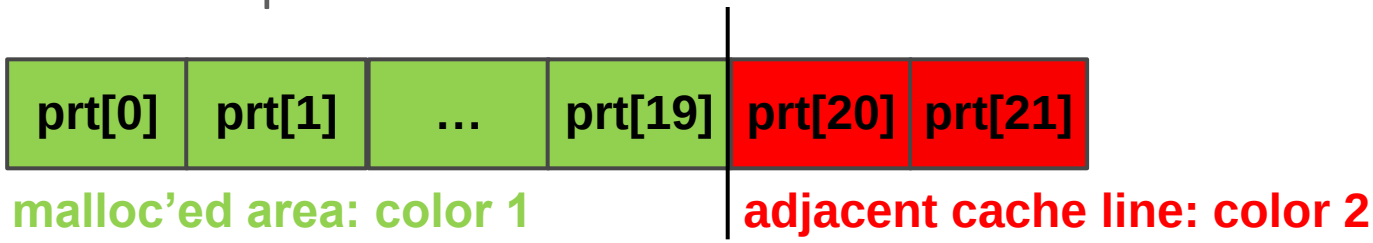


Linear Buffer Overflows

- ADI is really great at detecting **linear overflows**
- The attacker controls the size of the buffer being written, but not the starting address

```
char *ptr;
ptr = malloc(20);
strcpy(ptr, argv[1]); /* argv could be bigger than 20 chars */
```

  - The overflowed memory is **adjacent** to the buffer. Other live buffers, free buffers and potentially metadata may become corrupted
  - As long as the buffer adjacent to the one allocated for *ptr* has a different ADI color, any attempt to overflow will trap



A Couple of Famous Examples: Heartbleed & Venom

Silicon Secured Memory Protection From Read and Write Attacks

Buffer Over-Read Attack

Buffer Over-Write Attack

ORACLE

Heartbleed - Impacted Websites Using OpenSSL

Heartbeat request sent to victim

Type	Payload_size	Payload
HB_REQUEST	65535	Hello

Payload_size does not match Payload

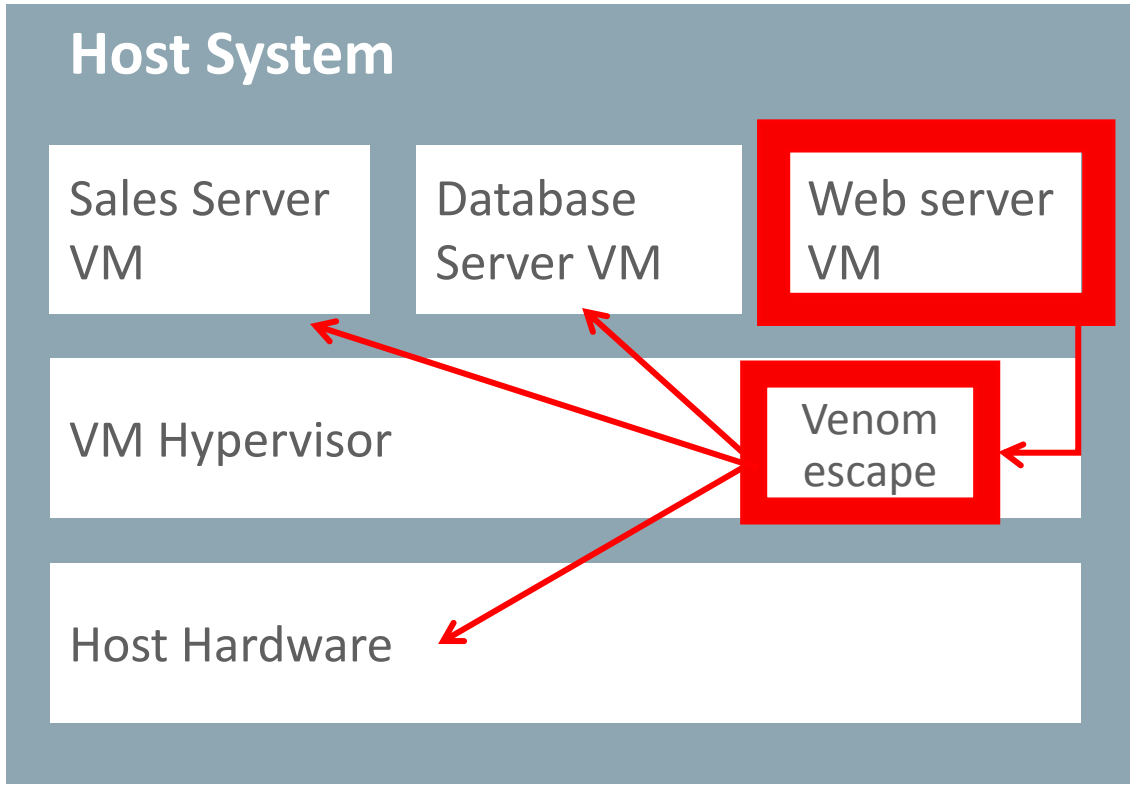
Victim responds with requested payload size (64K bytes)

Type	Payload_size	Payload
HB_RESPONSE	65535	Hello ……

Unauthorized data returned to requestor

ORACLE

Venom Vulnerability - Impacted Servers Using QEMU



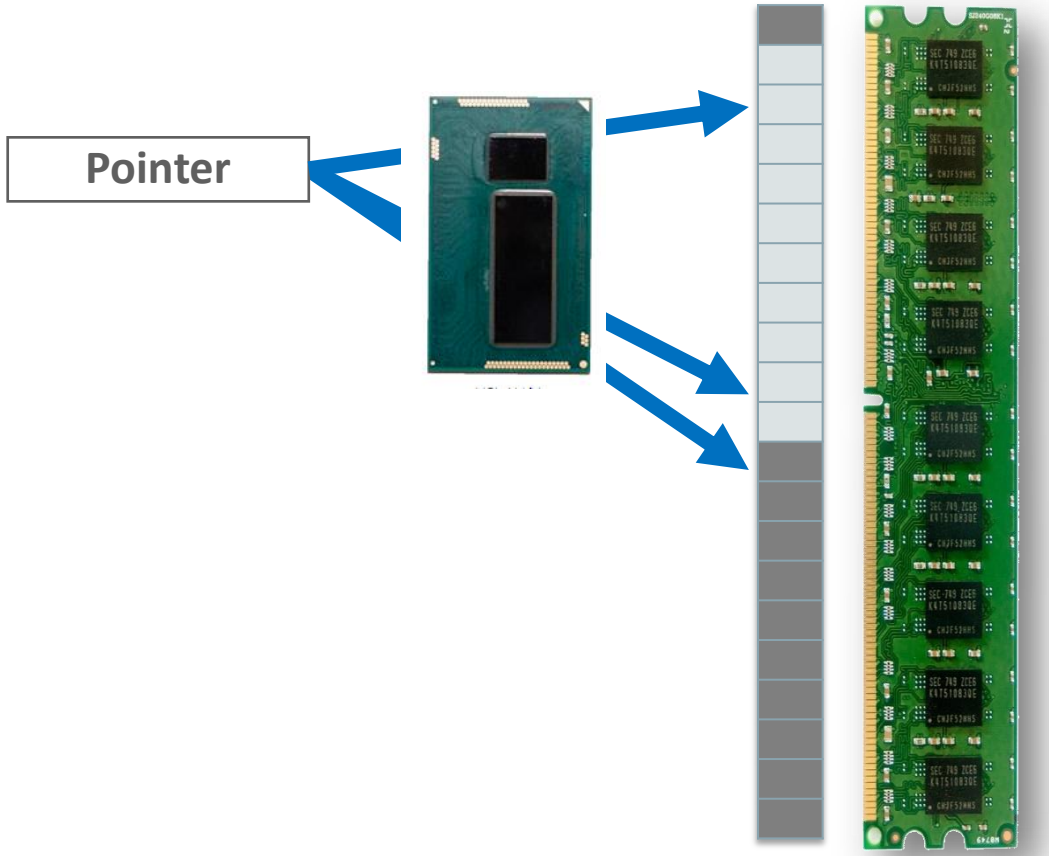
Hacker exploits VENOM to escape VM

VENOM executes instructions in hypervisor and gains control of host hardware

- Memory access vulnerability discovered in the open source Quick Emulator hypervisor platform (QEMU)
- Allows malicious code inside a VM guest to execute code in the host machine's hypervisor security context. The code then escape the guest VM to gain control over the entire host
- Caused by a buffer over-write condition that allows data to be stored beyond allocated buffer limits

Silicon Secured Memory: Buffer Overflows

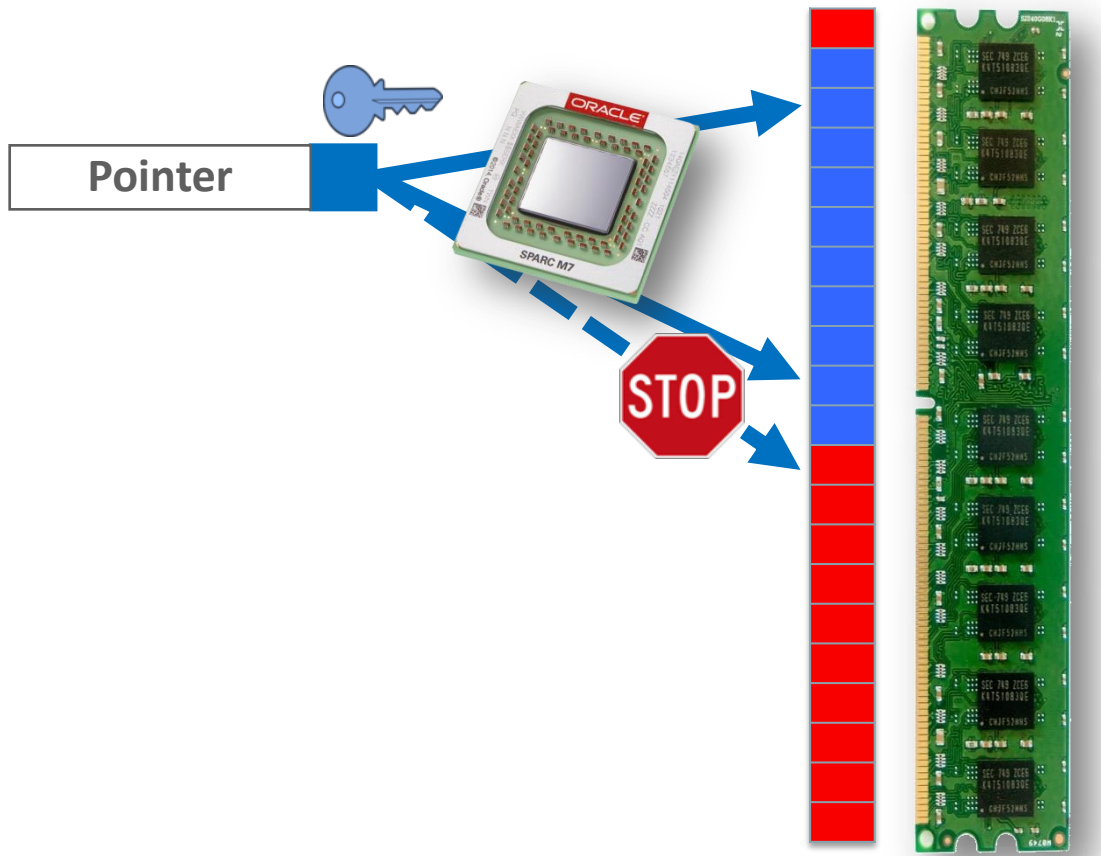
Any Processor



Applications

Memory

SPARC M7Processor

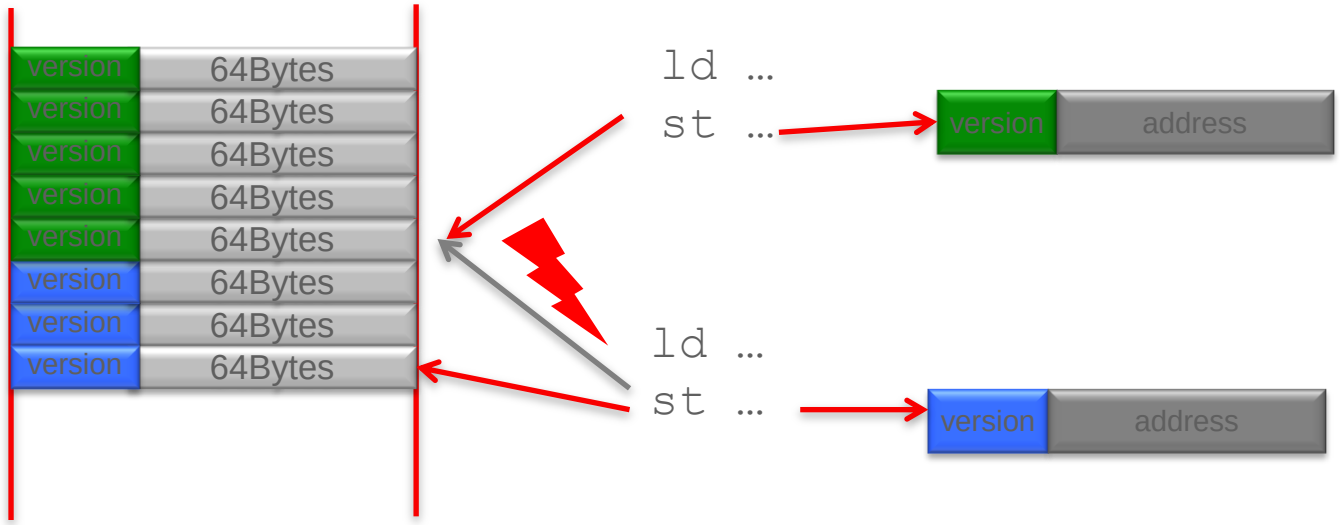


Applications

Memory

ORACLE

SSM Implementation: Application Data Integrity



(dbx) run
signal SEGV (ADI version 13 mismatch for VA 0x4a900) in main at 0x10988
(dbx) where
…stack trace…

- H/W compares pointer "key" with memory "lock"
  - are 4bit numbers
  - called "versions"
- Traps if they don't match
  - Sends SEGV or utrap to process
- H/W masks "key" before it hits the MMU

ADI version numbers and coloring

- *version* numbers use 4 bits
  - Valid range : 1 – 13
    - 0, 14 and 15 are reserved for system usage
    - By default all the memory is tagged with 0
    - 0 is not a valid *version* value for ADI checking
- Adjacent area paradigm
  - Adjacent areas are tagged with different version numbers
  - 4 bits are sufficient to tag uniquely adjacent buffers (for alloc and free)
  - Example

```
int *ptr = malloc(128);
free(ptr);
```

will set *version* as follow:

ptr[offset] (int)	version # (malloc)	version # (free)	notes
0 - 31	1	8	malloc'ed area
32 - 47	8	8	uphill adjacent cache line (the downhill adjacent cache line is not tagged)

Silicon Secured Memory

Support for Both Development and Deployment

DEVELOPMENT: Studio provides detailed diagnostics for developers to find and fix memory corruptions

DEPLOYMENT: Solaris enables applications to take appropriate recovery actions in real-time *

Solaris Studio 12.4/12.5 Beta *discover* tool

libdiscoveradi

libadimalloc

Solaris Kernel
(Provides syscalls for user-level applications)

SPARC M7 hardware
(Enables software stack for Silicon Secured Memory checking)

* App must be coded to use ADI APIs

ORACLE

Example Use of `libadimalloc.so`

Demo Code

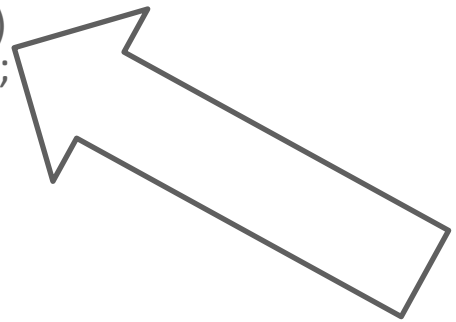
```
#include <stdio.h>
#include <stdlib.h>

int
main(void){
  char* public = (char*)malloc(sizeof(char)*100);
  char* secret = (char*)malloc(sizeof(char)*100);

  printf("public text -> ");  scanf("%s", public);
  printf("secret text -> ");  scanf("%s", secret);

  for(int ii = 0; ii < 150; ii++)
    printf("%c\n", public[ii]);

  printf("\n");
  return 0;
}
```



- Obvious Buffer Overflow (read beyond end)
- “public” buffer is 100bytes wide
- Code reads 150bytes
  - 50bytes are read from adjacent buffer

Output of Demo

On any system

```
franzh@SPARC-M7,ADI>./malloc
public text -> hello
secret text -> secret
h
e
l
l
o
---snip---

s
e
c
r
e
t
---snip---
franzh@SPARC-M7,ADI>
```

On a SPARC M7 using `libadimalloc.so`

```
franzh@SPARC-M7,ADI>
  LD_PRELOAD=libadimalloc.so ./malloc
public text -> hello
secret text -> secret
h
e
l
l
o
---snip---

Segmentation Fault (core dumped)
franzh@SPARC-M7,ADI>
```

Silicon Secured Memory

Support for Both Development and Deployment

DEVELOPMENT: Studio provides detailed diagnostics for developers to find and fix memory corruptions

DEPLOYMENT: Solaris enables applications to take appropriate recovery actions in real-time *

Solaris Studio 12.4/12.5 Beta *discover* tool

libdiscoveradi

libadimalloc

Solaris Kernel
(Provides syscalls for user-level applications)

SPARC M7 hardware
(Enables software stack for Silicon Secured Memory checking)

* App must be coded to use ADI APIs

ORACLE

Oracle Solaris Studio

Solaris & Linux, SPARC & x86, Remote Development

Multi-language Development

C, C++, Fortran Compilers

Debugger

Performance Library

Java

Application Analytics

Code Analyzer

Performance Analyzer

Thread Analyzer

Extensible IDE

ORACLE

Oracle Solaris Studio

Using discover and ADI to Find Memory Access Errors

- libdiscoverADI.so
  - enables discover to detect and understand runtime-related memory errors identified by ADI
    - `% LD_PRELOAD_64=<compiler>/lib/compilers/sparcv9/libdiscoverADI.so a.out`
    - `% discover -i adi a.out`
      `% a.out`
  - prints a comprehensive error analysis report for memory errors (text or html)

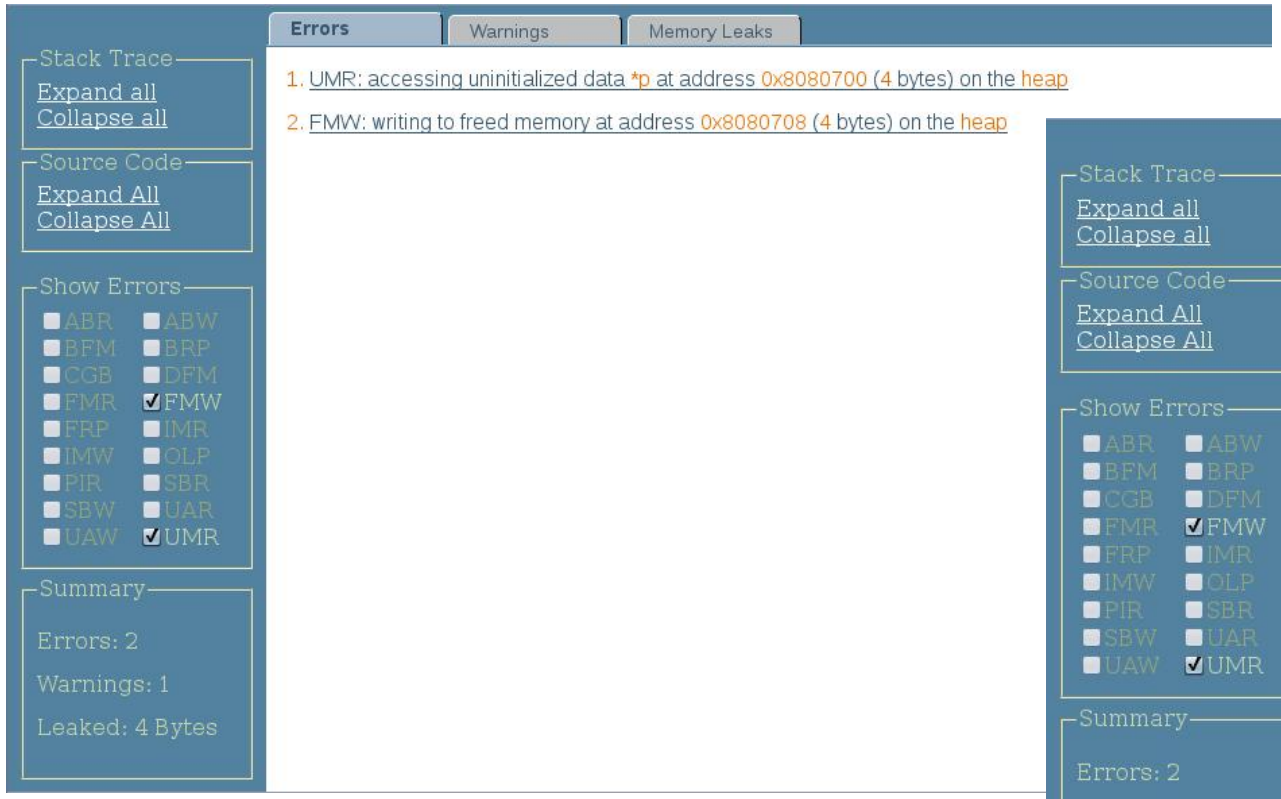
```
$ a.out
ERROR 1 (UAW): writing to unallocated memory at address 0x50088 (4 bytes) at: main() + 0x2a0 <ui.c:20>
17: t = malloc(32);
18: printf("hello\n");
19: for (int i=0; i<100;i++)
20:=> t[32] = 234; // UAW
21: printf("%d\n", t[2]); //UMR
22: foo();
23: bar();
_start() + 0x108 ERROR 2 (UMR): accessing uninitialized data from address 0x50010 (4 bytes) at: main() + 0x16c <ui.c:21>$
...
```

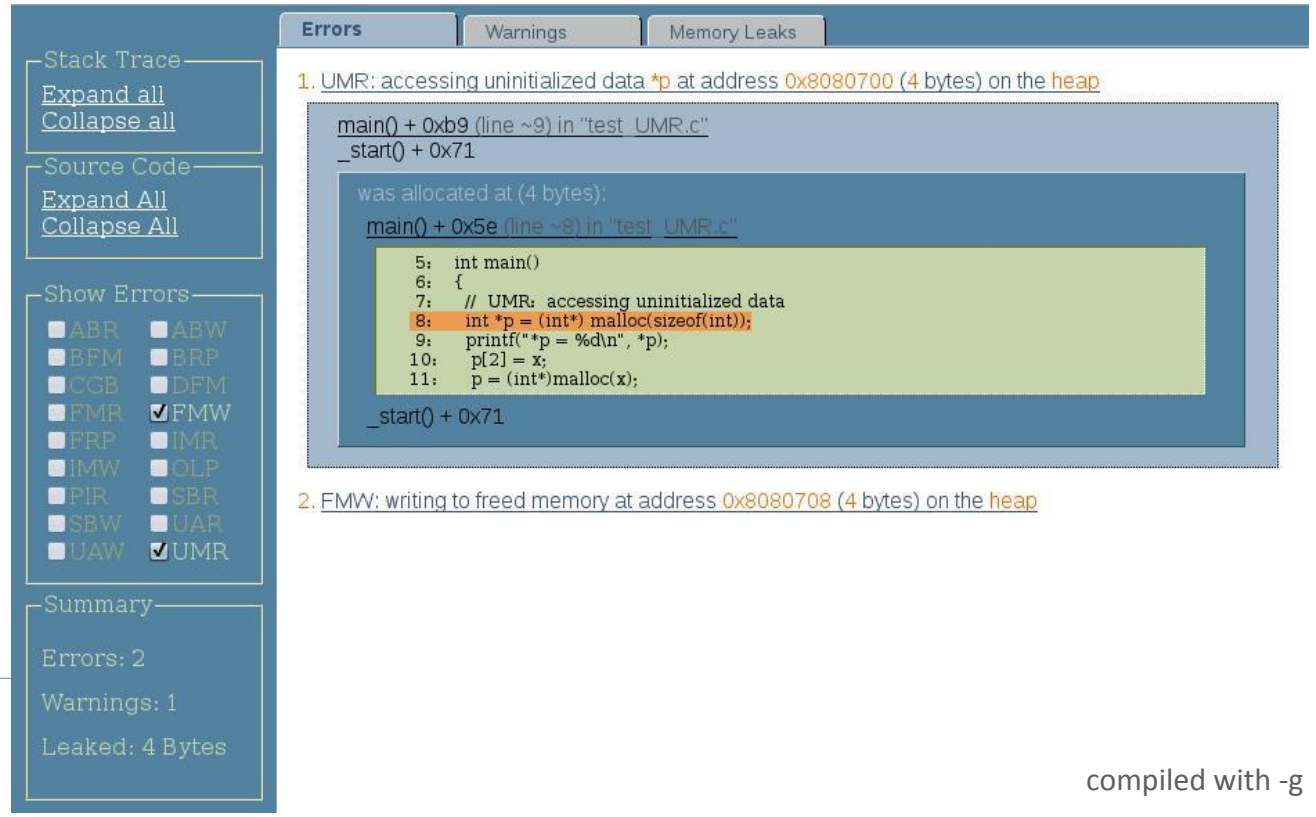
ORACLE

Oracle Solaris Studio 12.5 Beta: Discover and Uncover User's Guide - **Memory Error Discovery Tool (discover)**
https://docs.oracle.com/cd/E60778_01/html/E60755/gmzsf.html

Oracle Solaris Studio

Interactively Analyzing discover HTML-Report





compiled with -g

Oracle Solaris Studio 12.5 Beta: Discover and Uncover User's Guide – **Analyzing discover Reports**
https://docs.oracle.com/cd/E60778_01/html/E60755/gjzce.html

Oracle Solaris Studio

Code Analyzer: GUI to Navigate Tool Results

Oracle Solaris Studio Code Analyzer

File View Tools Window Help

Issues Sources

Issues: Core All

Analysis
Dynamic (1)
Severity
ERROR (1)
Bug type: ABR
ABR (Beyond Arra
ABW (Beyond Arr
OLP (Overlapping
UMR (Uninitialized
Binaries
libnsl.so.1
openssl (1)
Files
t1_lib.c (1)

Open Analysis | Results | Details Window | t1_lib.c

Showing 1 Issues Show: Snippets Reviewed Ignored New Fixed

ABR Beyond Array Bounds Read: at address 8ccc9eb (65535 bytes) on the heap

/export/sheldon/openssl/openssl-1.0.1f/ssl/t1_lib.c:2586

```
2583:           /* Enter response type, length and copy payload */
2584:           *bp++ = TLS1_HB_RESPONSE;
2585:           s2n(payload, bp);
2586:           memcpy(bp, pl, payload);
2587:           bp += payload;
```

Stacktrace

Call Stack
- **tls1_process_heartbeat** at t1_lib.c:2586
- **ssl3_read_bytes** at s3_pkt.c:1092
- **ssl3_get_message** at s3_both.c:458
- **ssl3_check_client_hello** at s3_srvr.c:882
- **ssl3_accept** at s3_srvr.c:581
- **SSL_accept** at ssl_lib.c:940
- **ssl23_get_client_hello** at s23_srvr.c:634
- **ssl23_accept** at s23_srvr.c:210

Allocated At Stack (size 17,736)
- **default_malloc_ex** at mem.c:79
- **CRYPTO_malloc** at mem.c:308
- **freelist_extract** at s3_both.c:708
- **ssl3_setup_read_buffer** at s3_both.c:770
- **ssl3_setup_buffers** at s3_both.c:827
- **ssl23_get_client_hello** at s23_srvr.c:266
- **ssl23_accept** at s23_srvr.c:210
- **ssl23_read** at s23_lib.c:127

Error Type

Memory Freed

Memory Allocated

Oracle Solaris Studio 12.5 Beta: **Code Analyzer User's Guide**
https://docs.oracle.com/cd/E60778_01/html/E60757

Errors Caught by discover and ADI

- Buffer overflow errors

```
int *area1 = malloc(sizeof(int)*16);
for (int i = 0; i <= 16; i++)
      area1[i] = 0;      // Array Out of Bounds
```

- Freed memory access errors

```
free(area1);
area1[0] = 0; // Freed memory access error
```

- Stale pointer memory access errors

```
int *area1 = malloc(sizeof(int)*16);
free(area1);
char *area3 = malloc(sizeof(char)*64); // area3 gets the memory area just
freed by area1 area1[0] = 0; // Stale Pointer Access
```

- Double free memory access errors

```
free(area3);
free(area3); // double free
```

Silicon Secured Memory Developer Tool: *discover*

- *Discover* detects runtime memory access violations and memory leaks
- *Discover* provides detailed diagnostics to find and fix these errors
- Studio 12.5 *discover* uses M7 Silicon Secured Memory, making violation detection significantly faster than a software-only approach

- **[ABR | ABW] – Beyond Array Bounds Read/Write**
- **[FMR | FMW] – Freed Memory Read/Write**
- [IMR | IMW] – Invalid Memory Read/Write
- **[UAR | UAW] – UnAllocated memory Read/Write**
- [NAR | NAW]– Non-Annotated Read/Write
- [SBR | SBW]- beyond Stack Bounds Read/Write
- BFM – Bad Free Memory
- BRP – Bad Realloc address Parameter
- CGB – Corrupted Guard Block
- **DFM – Double Freeing Memory**
- PIR – Partially Initialized Read
- UMR – Uninitialized Memory Read
- OLP – Overlapping source and dest
- AZS – Allocating Zero Size
- SMR– Speculative Memory Read
- [UFR | UFW] – Unknown stack Frame Read/Write
- [USR | USW] – Unknown Status while Read/Write

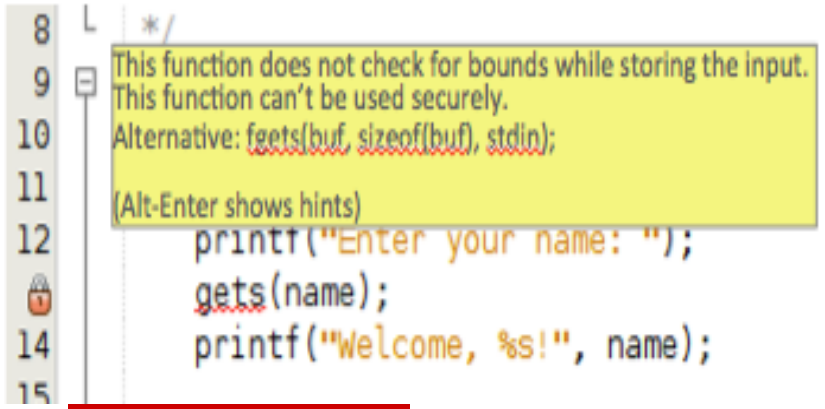
Oracle Solaris Studio 12.5 Beta: Code Analyzer User's Guide – **Dynamic Memory Access Errors**
https://docs.oracle.com/cd/E60778_01/html/E60757/glmrb.htm
Code Analyzer previse may detect additional error types through static code analysis
https://docs.oracle.com/cd/E60778_01/html/E60757/glmsy.html

ORACLE

Studio 12.5: Security Features beyond SSM

1. Write Secure Code

- IDE identifies unsafe code
  - Uses Solaris C guidelines and some CERT C/C++ rules
- Explains issue and offers a more secure alternative



2. Build Secure Code

- Source code analysis done with every compile by default
  - previse
- Checks include:
  - Beyond array bounds access
  - Freed memory
  - Memory leaks

```
$ cc -O –c  test.c
”test.c", line 5: Warning: Likely out-of-bounds
read: a[i] in function main
```

3. Run Secure Code

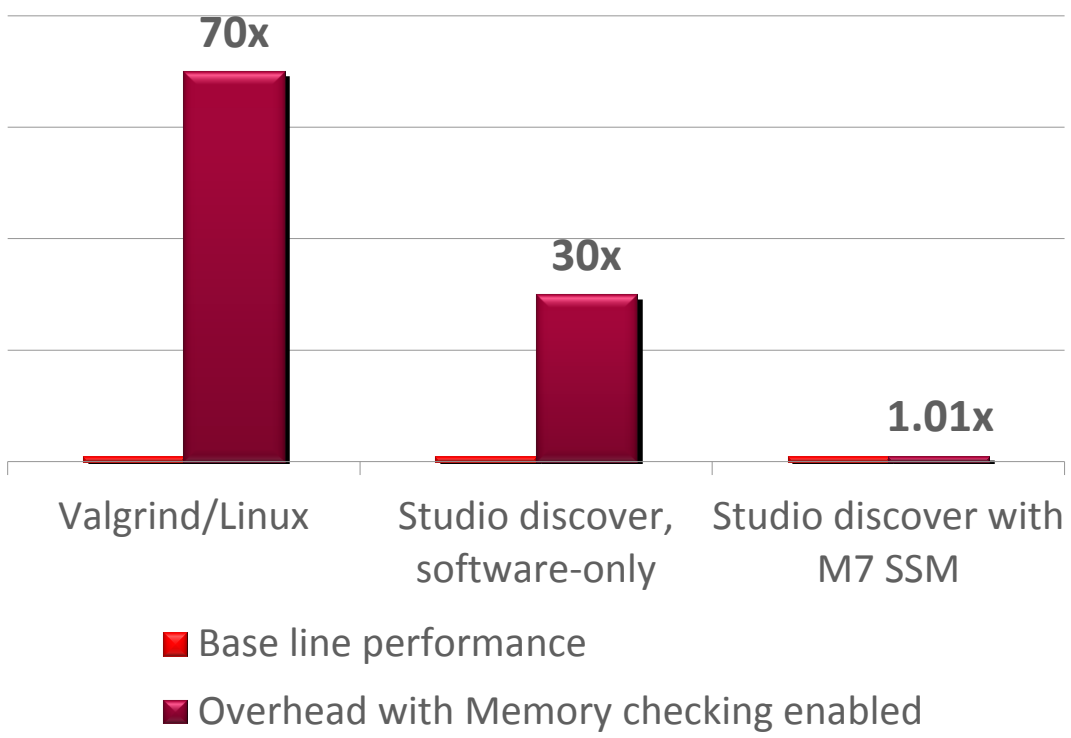
- Compiler includes checks in every app to catch:
  - Stack overflow [*-xcheck=stkovf*]
  - Falling off the end of a routine [*-xcheck=noreturn*]

ORACLE

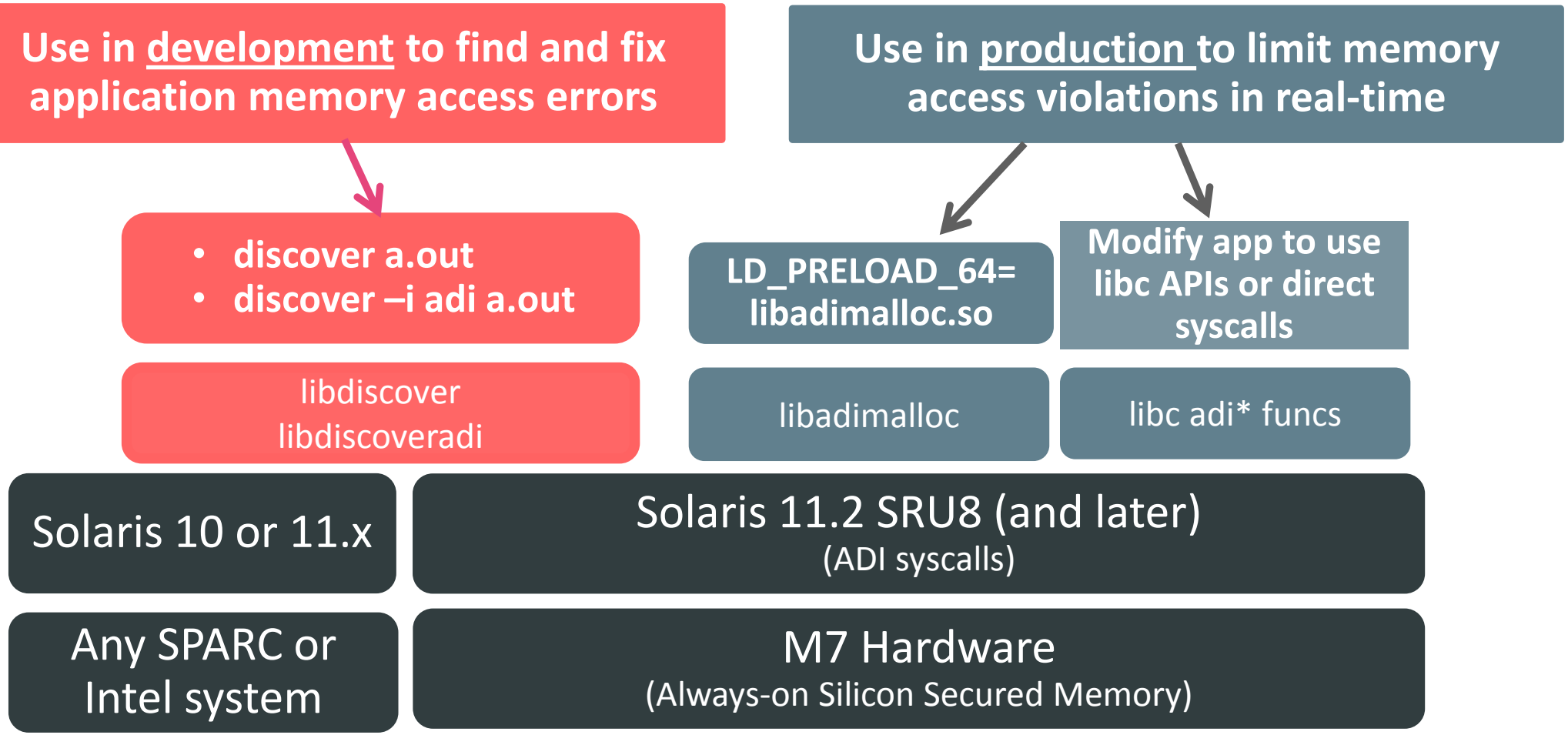
Developer's Guide to Oracle Solaris 11 Security - **Secure Coding Guidelines for Developers** http://docs.oracle.com/cd/E53394_01/html/E54753/scode-1.html
Appendix G: Security Considerations When Using C Functions http://docs.oracle.com/cd/E53394_01/html/E54753/gnclc.html

Developing Secure Software using Oracle Solaris Studio

- How Discover SSM works
  - Interposes on mem allocation routines
  - Assigns versions to pointers; ensures version doesn't match a neighbor's version
  - Catches the SEGV traps when illegal access occurs (i.e. version mismatch)
  - Collects error source line/stack trace and allocation/free source line/stack trace, then allows app to continue
  - Generates report of all recorded errors at end of run



Silicon Secured Memory for Both Development & Production



Low Level SSM – Solaris ADI API

Custom Memory Allocator

- An application needs to meet the following requirements for ADI code checking
  - The application binary must be built in 64-bit mode
  - The application needs to enable ADI on the target memory area
  - The allocated memory needs to be 64-byte aligned and its size must be multiple of 64
  - The allocated area should be set a version number
    with the pointer value being adjusted with the corresponding version number.
  - Complex pointer manipulation should be avoided, but simple pointer operations works

Custom Memory Allocator

Example

- Memory allocator needs to maintain version data
  - Writes version into memory during allocation
  - Returns pointer with version embedded
  - Allocator writes different version to cache line when freed
- 2 ranges of version numbers: one for memory **allocation** and one memory **free**

1 : used for the area of block object including block buffer
2 – 7 : used for the allocated name locations inside the block buffer
8 – 13 : used for the freed name locations inside the block buffer
mapping : 2 - 8, 3 - 9, 4 - 10, 5 - 11, 6 - 12, 7 - 13

ADI'fying Custom Memory Allocators

64-bit mode	cc –m64

Memory ADI enabled	`large_block_ptr = (large_block*) memalign(8192, 64 * 1024);`
	`memcntl(large_block_ptr, 64 * 1024, MC_ENABLE_ADI,NULL,0,0)`
	- Both address and size must be PAGESIZE (8k) aligned

64-bit alignment	`object_ptr = (my_object*) my_malloc(sizeof(my_object));`
	needs to be changed to:
	`adjusted_size = (sizeof(my_object) + 63) & ~63; // adjust to multiple of 64`
	`object_ptr = (my_object*) my_malloc(64,adjusted_size); // 64-byte aligned`

Version numbers	`adjusted_object_ptr = (my_object*)` `  adi_set_version(object_ptr, adjusted_size, new_version_number);`

Pointer manipulation	Pointer operations such as array element access by adding pointer and index value still work
	`adjusted_array_ptr = (my_array*)`
	`  adi_get_version(array_ptr, adjusted_array_size, my version_number);`
	`(adjusted_array_ptr + 2)->value = 100; // set the third array element`
	`// structure value field to 100`

Custom Memory Allocator

More Information

- Fully documented example in SSM cookbook
  - http://swisdev.oracle.com -> Resources
- Using Application Data Integrity and Oracle Solaris Studio to Find and Fix Memory Access Errors
  - https://community.oracle.com/docs/DOC-912448
- Custom Memory Allocators and the discover SSM Library
  - https://docs.oracle.com/cd/E60778_01/html/E60755/gphwb.html

ADI Caveats

- 64-bit processes only
- Performance impact
  - Negligible for the default disrupting traps
  - Optional precise traps for store mismatches have a noticeable impact, should only be used for debug
  - Updating versions is negligible
- Normalize pointers before
  - compare
  - arithmetical operations
- ADI has a high probability of catching bugs, but a bad pointer may accidentally have a matching version
- DMA read (write to memory) resets ADI version to 0
  - Impacts userland only if Direct I/O is used

ADI Observability

pmap without/with libadimalloc.so

- ADI not used

```
> pmap -xs `pmap malloc`
2899:   ./malloc
         Address     Kbytes        RSS       Anon     Locked Pgsz Mode   Mapped File
---snip---
FFFFFFFF7F7D0000         16         16         16          -   8K rwx----    [ anon ]
FFFFFFFF7F7D4000          8          -          -          -    - rwx----    [ anon ]
---snip---
```

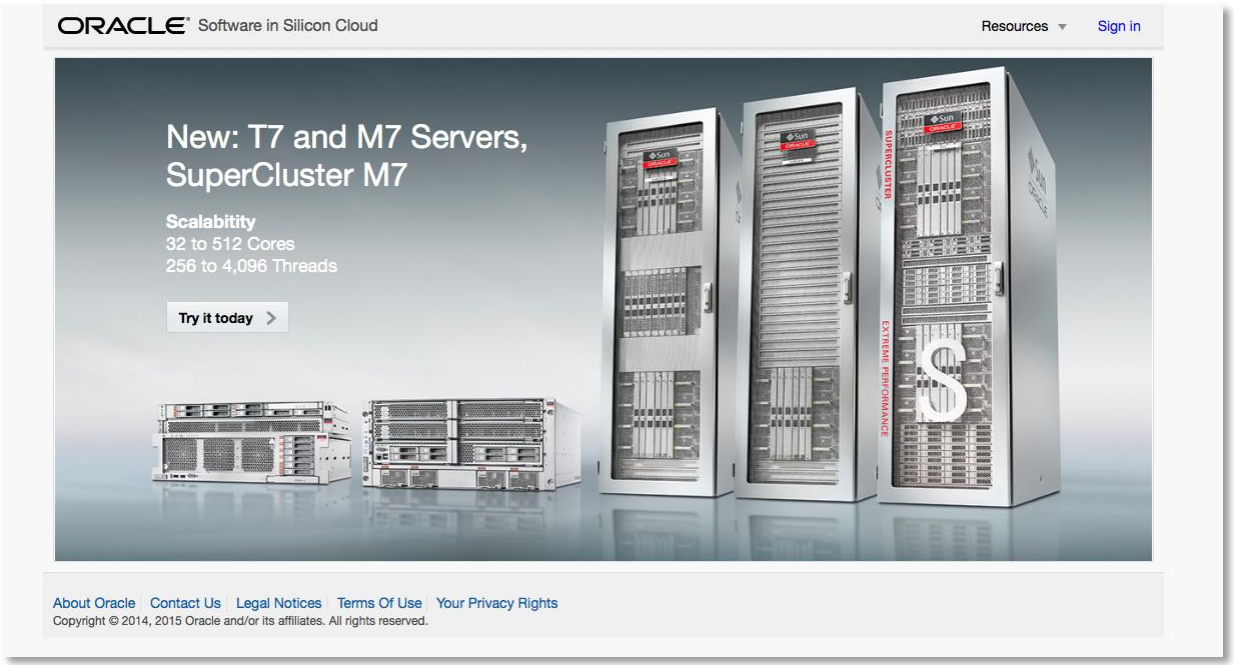
- ADI “active”

```
> pmap -xs `pmap malloc`
2903:   ./malloc
         Address     Kbytes        RSS       Anon     Locked Pgsz Mode   Mapped File
---snip---
FFFFFFFF7DCF0000        256        256        256          -  64K rwx--i-    [ anon ]
FFFFFFFF7DD40000        320        320        320          -  64K rwx--i-    [ anon ]
---snip---
```

Oracle Software In Silicon Cloud

Opening up to broader set of developers

- Online Click through agreement
- Free for OPN partners
- SPARC Enterprise Developers
- University Researchers



http://SWiSdev.Oracle.com

ORACLE

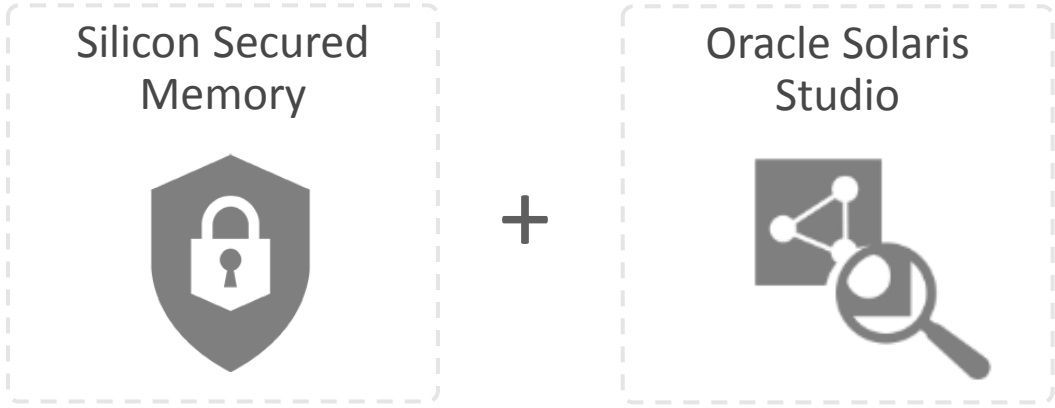
How You Can Use Silicon Secured Memory

- Enable your existing software – No need to recompile!
  - Check application binaries with Solaris Studio 12.4 / 12.5 Beta
  - Link with Solaris libraries – e.g., *malloc()* enhanced with ADI: *libadimalloc*
  - Certify on your test environment
- Develop your applications with Silicon Secured Memory
  - C/C++ 64-bit code, Solaris ADI API
  - Comprehensive tools available with Solaris Studio 12.4 / 12.5 Beta
- Run applications that are enabled with Silicon Secured Memory (examples):
  - Oracle Database 12c (12.1.0.2) uses Silicon Secured Memory in SGA
    - 12.1.0.2 Readme: 2.4 Data Analytics Accelerators on SPARC for Oracle Database Overview
  - ISV software that has been developed with Silicon Secured Memory

Real World Experience

A Case Study

- Large enterprise app with heavy use of memory intensive processing
- Time to value for SPARC M7
  - **4 cross platform** bugs tagged in 2 days
  - **180x faster** bug identification
    - Other memory validation tool: 3 hours
    - Silicon Secured Memory and *Discover*: 1 minute



Integrated. Simple. Fast.

The M7 Microprocessor Can Protect the Entire Cloud

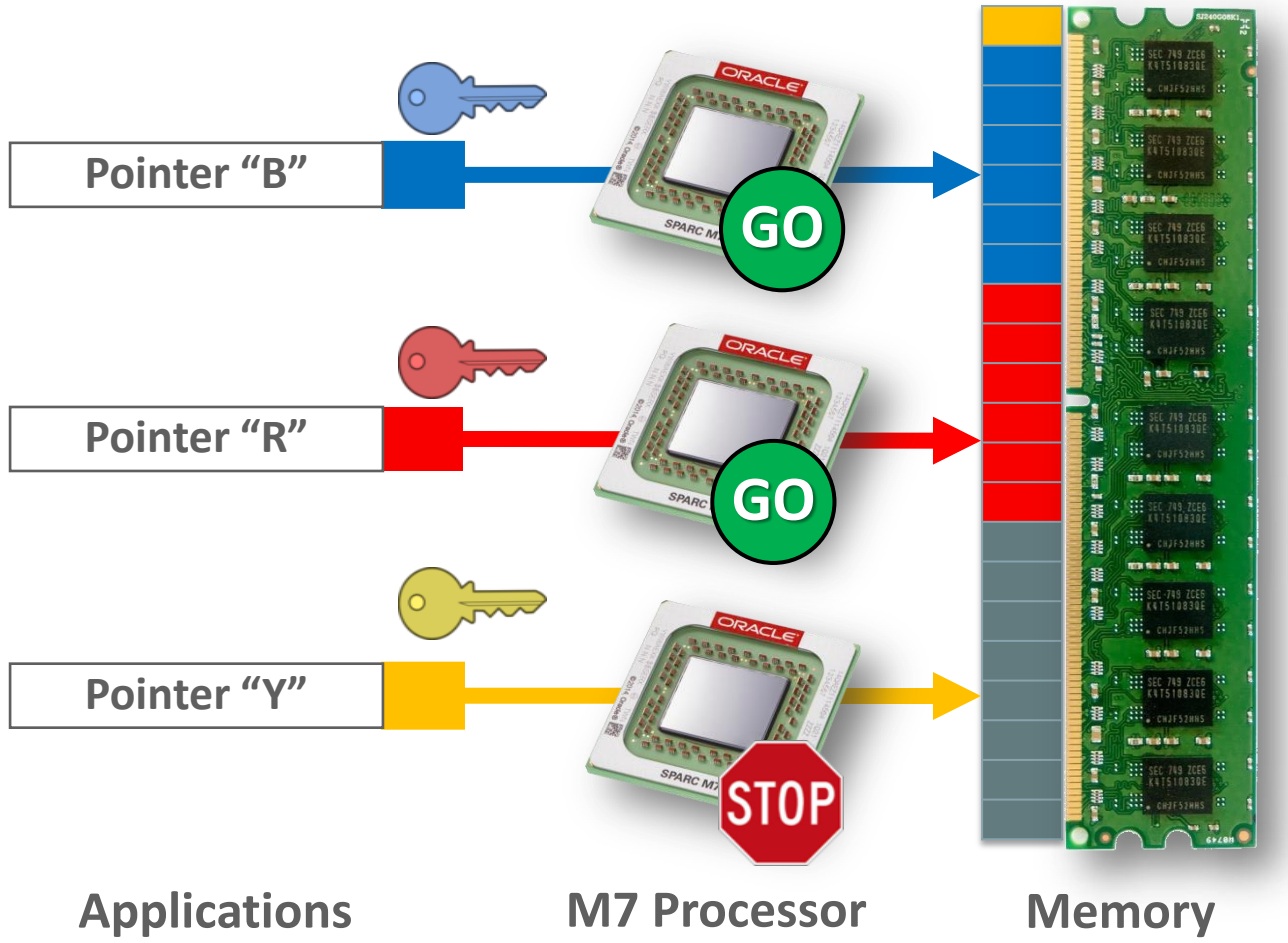
Even if 90% of the Microprocessors are not M7s

- Even a few deployed M7 systems can detect an attack on the entire compute cloud
- Once an attack is discovered, the other unprotected systems then can be patched

Oracle M7

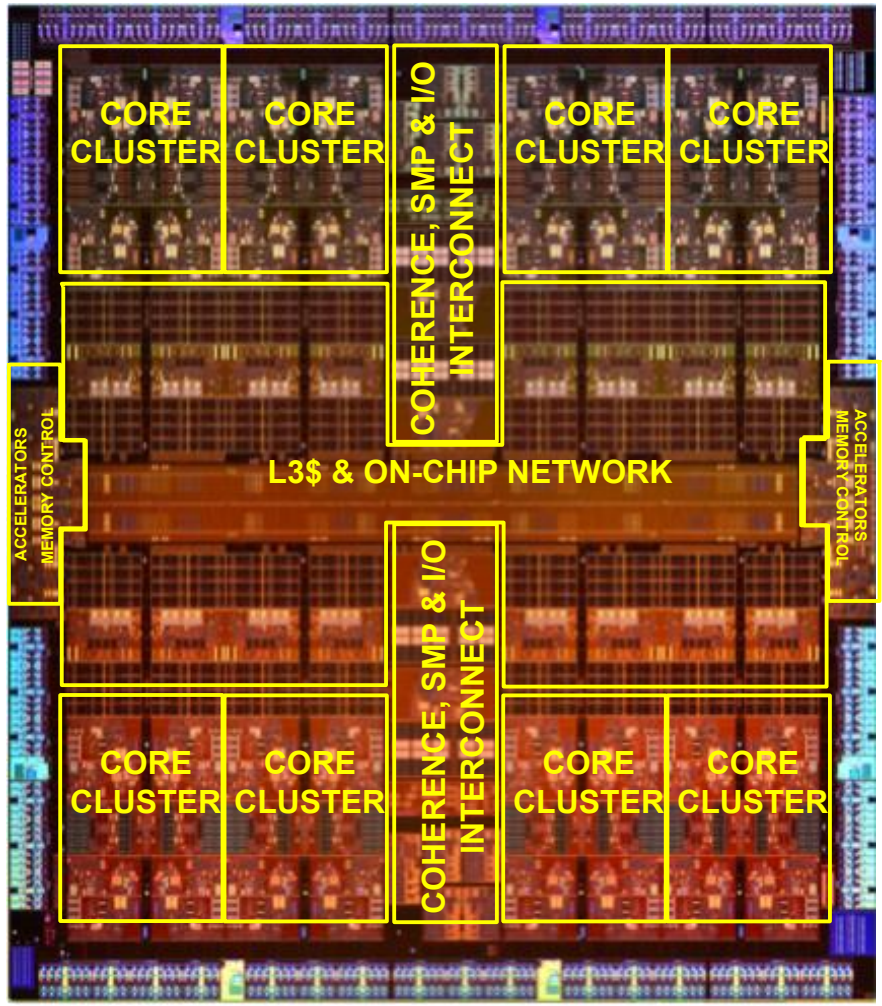
Silicon Secured Memory

- **Protects data in memory**
- Hidden “color” bits added to *pointers* (key), and content (lock)
- Pointer color (key) must match content color or program is aborted
  - Set on *memory allocation*, changed on *memory free’*
  - Protects against *access off end of structure*, *stale pointer* access and malicious attacks
- **Extremely efficient for software development**



Advancing the State-of-the-Art

M7 Microprocessor – World's First Implementation of Software Features in Silicon



- **Always-On Security in Silicon**
  - Near zero performance impact
  - Use in production
  - Silicon Secured Memory (SSM)
    - Application Data Integrity (ADI)
- **High-Speed Encryption**
  - Near zero performance impact
  - 32 Crypto Accelerators
- **SQL in Silicon**
  - High-Speed Memory Decompression
    - "Capacity in Silicon"
  - Primitives to accelerate In-Memory Database Operations
  - 8 Data Analytics Accelerators (DAX) w/ 32 Pipelines
  - Apache SPARK demo at OOW2015

Silicon Secured Memory

More Information

- Silicon Secured Memory Cookbook
  - https://swisdev.oracle.com/_files/ssm-cookbook-page1.html
- Using Application Data Integrity and Oracle Solaris Studio to Find and Fix Memory Access Errors
  - https://community.oracle.com/docs/DOC-912448
- See Raj Prakash's blog @ https://blogs.oracle.com/raj/
  - Oh, no! What Have I Done Now? - Common Types of Memory Access Errors
  - Let's Get The Low Hanging Fruits - Static detection of memory access errors using Previse
  - Solving Trickier Problems - Detecting Dynamic Memory Access Errors Using Discover
  - Surprise! Unexpected Benefits of Hardware Support for Detection of Memory Access Errors

PDF: https://blogs.oracle.com/raj/resource/Silicon-Secured-Memory-Application.pdf

Oracle Solaris Studio

More Information

- History from SPARCWorks to Sun Workshop to Forte Developer to Sun Studio to Oracle Solaris Studio
  - https://blogs.oracle.com/tatkar/entry/studio_release_names_from_the
  - http://www.oracle.com/technetwork/server-storage/solarisstudio/training/index-jsp-141991.html
- on OTN
  - http://www.oracle.com/technetwork/server-storage/solarisstudio/overview/index.html
- Oracle Studio YouTube Channel
  - https://www.youtube.com/watch?v=9gOtXtHfvI4&list=PLKCk3OyNwIzuRh2YsM2MtFAwB_qEWC5Rn&index=3
- Remote Development
  - https://www.youtube.com/watch?v=R8ELRznEoSQ&list=PLKCk3OyNwIzuRh2YsM2MtFAwB_qEWC5Rn&index=24
- Oracle Solaris Studio Learning Library (Screencasts)
  - https://apexapps.oracle.com/pls/apex/f?p=44785:141:10078869691805::NO:141:P141_PAGE_ID%2CP141_SECTION_ID:147%2C1059


ORACLE®