

Fine-grained Sharing

Friday, March 27, 2009
12:53 PM

Goal - Fine-grained sharing

Obstacle - Cost of cross-domain calls

Remote Procedure Call - Machine 1 passes data to Machine 2 across network
Process 1 to Process2 using the kernel - unavoidable overhead, context switch. 2000 times slower than simple function call.

SFI - allow mutually distrusting modules to exist in the same address space.

Security Goals

- Integrity of each domains memory and OS resources
- Confidentiality of the same
- How to support cross domain procedure calls

Assumption

- Load/store architecture
 - o Load
 - o Store
 - o Reg-reg instr
- Wealth of Registers
- Untrusted party inserts checks
- Trusted party verifies checks before executing code.

Burn register for checking
\$sid=0x020000.0

```
Mov $p,$spr
Xor $spr, $sid, $t
Shiftr $t,24,$t
Jnz $t, ERROR
store $p, $r
```

To Verify - before execution

- Untrusted code never writes to \$sid
- All stores must use \$spr
- Each module given own stack and heap
- Os resources must be accessed through trusted arbitrator module
- \$spr is checked after each modification
- Untrusted code performs no sys calls.

\$sid=0x020000.0
\$spr

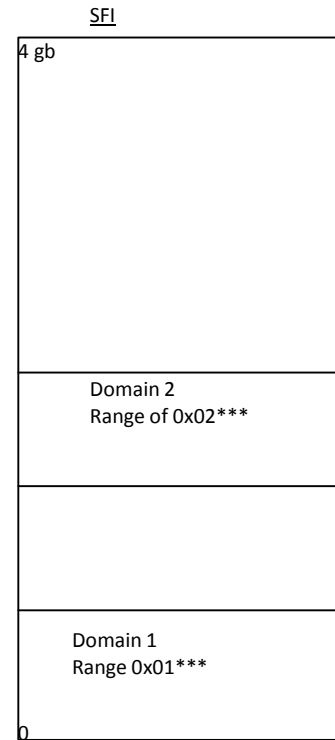
```
Xer $p,$sid,$t
Shiftr $t,24,$t
Jn2 $t ERROR
Store $r, $spr
```

Cross Domain Call

- Each domain has entry point
- Other domains should only be able to enter at the points
- T bookkeeping

Overhead of SFI

Function call	SFI	CS
.1ms	1ms	200ms



unwritable

