Capabilities and DIFT

Tony Espinoza

am.espinoza@utexas.edu

Security mechanisms

- Talk about two security concepts.
  1. Capabilities.
     - Preempt with introduction of principle of least privilege.
  2. Information Flow Tracking.
- Project questions.

Least privilege

The principle of least privilege states that a subject should be given only those privileges that it needs in order to complete its task.[1]

- Real world(ish) examples:
  - Valet key.
  - Amazon trunk delivery.

[1] Introduction to Computer Security, Matt Bishop

Least privilege

- Guest speaker mentioned least privilege, what did he have to say?
- How can we follow least privilege in containers?
  - Don't run as root.
  - Limit system calls allowed.
  - Secrets a container has access to.

Capabilities

- A way to enforce least privilege.
- An access control mechanism.
- What other access control mechanisms have we discussed?
  - Mandatory access control.
  - Discretionary access control.
- Capabilities are similar to Access Control Lists (ACL).
  - Given an object, give the list of subjects and rights for each subject.
  - acl(file a) = { (proce, executess 1, { read, write}), (process 2, { append, execute }) }
  - `getfacl` Linux command line tool for ACL.

Capabilities

Each subject (**e.g.** process) has associated with it a set of pairs, with each pair containing an object (**e.g.** file) and a set of rights (**e.g.** read, write).

Capabilities

Capability list c is a set of pairs of objects (o) and rights (r):
$c = \{o, r : o \in O, r \subseteq R\}$

cap() is a function that returns a capability list associated with subject s.

Capabilities example

cap(process 1)

Capabilities example

cap(process 1) = { (file 1, { read, write, own }), (file 2, { read }),
(process 1, {read, write, execute, own}), (process 2, { write }) }

Capabilities example

- File descriptor in Linux is a capability.
  - The capability is tightly bound to the file object.
  - If the file is deleted and a new file with the same name is created, the file descriptor still refers to the previous file.

Security of Capabilities

- Security is assured by three properties:
  - Capabilities are unforgeable and tamper proof.
  - Processes are able to obtain capabilities only by using the authorized interfaces.
  - Capabilities are only given to processes that are authorized to hold them.

Capabilities vs Access Control Lists

- Answer two questions:
  1. Given a subject, what objects can it access, and how?
  2. Given an object, what subjects can access it, and how?
- Which does each answer easily?
  - First question capabilities answers easily
  - Second question ACL answers easily

EROS: a fast capability system

- EROS is a system built to run on the Pentium processor.
- Capabilities are the only mechanism for naming and using resources.
- Used a microkernel architecture.
  - Microkernel is the opposite of a monolithic kernel like Linux.
  - Every OS in use is pretty much a monolithic kernel.
  - Kernel is in charge of as little as possible.
    - **e.g.** low-level address space management, thread management, and inter-process communication (IPC)
- Development stopped in 2005.

EROS

- Leverages microkernel architecture.
  - Microkernel architecture has many subsystems that are in charge of different areas:
    - **e.g.** device drivers, protocol stacks and file systems.
  - EROS uses protected domains:
    - A set of capabilities accessible to a subsystem.

EROS evaluation

- No applications ported to it.
  - No apples to apples comparison is possible.
- What did they do instead?
  - micro benchmarks that are motivated by real performance bottlenecks from real applications.
  - Is this a good practice?

EROS evaluation

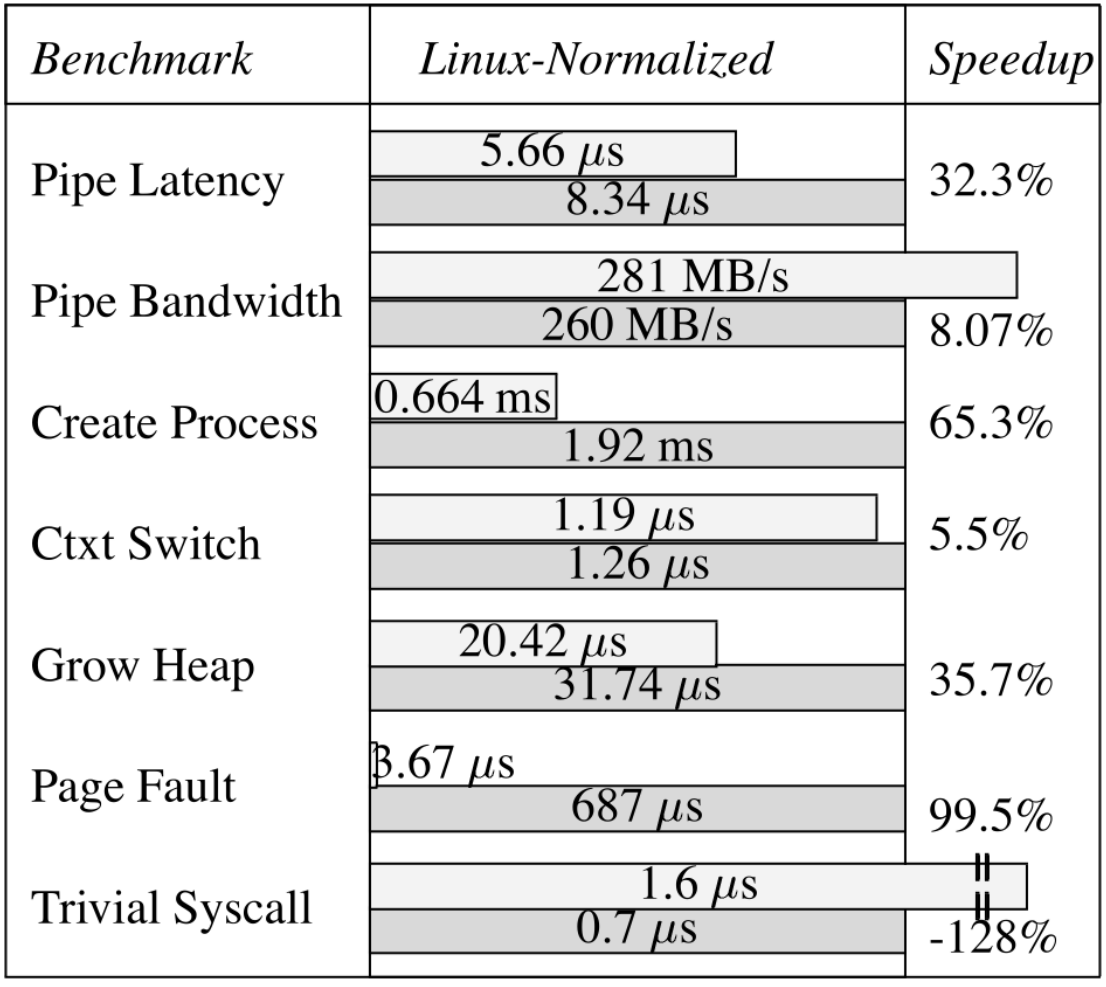

Figure 11. Summary of benchmark results. For pipe bandwidth, larger is better. Linux *lmbench* results appear in dark gray. EROS results are normalized to the Linux numbers, and appear in lighter gray.

Linux capabilities

- man capabilities
- Not the exact same as the capabilities discussed.
- Linux (2.2+) divides the privileges traditionally associated with superuser into distinct units, known as capabilities, which can be independently enabled and disabled. Capabilities are a per-thread attribute.

Linux capabilities

- These were the capabilities discusses last lecture.
- Can be directly applied to containers.
- Enforce the idea of “lest privilege”.
- Capsicum: practical capabilities for UNIX
  - A more traditional capabilities method.

Docker seccomp

- Secure computing mode (seccomp) is a Linux kernel feature. You can use it to restrict the actions available within the container.

```
$ docker run --rm \
         -it \
         --security-opt seccomp=/path/to/seccomp/profile.json \
         hello-world
```

https://docs.docker.com/engine/security/seccomp/

Dynamic Information Flow Tracking (DIFT)

- Sometimes called Dynamic Taint Analysis (DTA).
- Mark data(memory/registers) with a tag that propagates through program execution.
- What do I mean by tag?
  - First versions were 0,1 (trusted, untrusted).
  - Metadata that is attached to data.
  - Can be a color, label, vector **etc.**.

Reading and writing tags

- Source
  - Tags are introduced into a system through a source.
    - File, network data, **etc.**
- Sink
  - Sinks are locations in the execution where you check a policy.
    - Send, write, conditional jumps, **etc.**

DIFT

- Five types of dependencies[2]
  - Load Address
    - a = b[2]
  - Store Address
    - b[2] = a
  - Computation
    - a = b + c
  - Copy
    - a = b
  - Control
    - if, switch, test, jump

[2] SecureProgramExecutionviaDynamicInformationFlow Tracking, Suh et al.

Control dependency

Probably the most difficult, due to it's indirect effect.

```
z = 5;
if 4 > y:
    z = 10;
```

DIFT

- pros
  - Can be done on binaries.
    - Don't need source code.
  - Can set policies to how information is allowed to flow.
    - "If privledged data tries to leave the system error"
- cons
  - overhead.
  - implementation difficulties.

Over-tainting

- An issue that happens with control dependencies

```
test eax, ebx    ; set ZF to 1 if eax == ebx
je 0x804f430   ; jump if ZF == 1
```

At this point we test 2 registers for equality and then jump if they are equal. After the `je` instruction how do we propagate tags/taint?

DIFT use cases

- Catch:
  - Buffer overflows.
  - SQL injection.
  - Directory traversal.
  - Command injection.
  - SQL injection.
  - Cross-site scripting.
- Reverse engineering:
  - Find keys in memory.
- Raksha: A Flexible Information Flow Architecturefor Software Security.

LaTeX links

- https://en.wikibooks.org/wiki/LaTeX
- http://detexify.kirelabs.org/classify.html