Labels and Event Processes in the Asbestos Operating System

Petros Efstathopoulos, Maxwell Krohn, et al.

KARTHIK ANANTAPUR BACHERAO

10/28/2005

1

MOTIVATION

- Computer Systems do not provide adequate security
  - Exploitable software flaws (Buffer Overflows,etc)
- Source of Problem:
  - Bugs in Software.
  - Users willing to run untrusted code.
- No isolation of services

2

Motivation (Contd)

- Principle of Least Privilege (POLP) not enforced.
  - Each bit of code that executes in a machine should run with least amount of privilege.
- Developers should follow five requirements:
  - Split application into protection domains or compartments
  - Assign exact privileges to the compartments.
  - Engineer communication between compartments.
  - Compartments should be isolated from one another.
  - Should be easy to perform a security audit

3

OUTLINE

- SECURITY MODELS
- ASBESTOS OS
- ASBESTOS LABELS
- ASBESTOS EVENT PROCESSES
- PERFORMANCE

4

Security Models

- Mandatory Access Control:
  - Power with the owner of the system.
  - Uses labels.
  - Generally employs a variant of the *-Property
    - Whenever a process P can observe Object O1 and modify Object O2, O2's security level should dominate O1's
- Discretionary Access Control
  - Security by Ownership.
- POLP with MAC

5

Asbestos: A New Operating System

"Asbestos should support efficient, unprivileged and large-scale server applications whose application-defined users are isolated from one another by the operating system, according to application policy."

- A message passing micro-kernel based architecture.
- New Labeling and isolation mechanism
  - Asbestos labels provide both mandatory and discretionary access control
  - Decentralized MAC.
  - A process can bypass the *-property by declassifying information

6

Asbestos: A New Operating System (Contd)

- Event Processes
  - Helps to support and isolate multiple concurrent users.
  - Provides light-weight isolated contexts.

7

Asbestos Labels (Contd)

LABEL BASICS

- Handles:
  - Are 61-bit unique identifiers to name compartments.
  - Handle privileges are represented by Levels which are members of the ordered set {*, 0 , 1, 2, 3 }
- Labels:
  - A function from handles to levels.
  - Eg. {a 0, b 1, 2}
- Label Comparison:
  - A ≤ B iff A(h) ≤ B(h) for all h.
- Least Upper Bound
  - (A U B)(h) = max(A(h),B(h))
- Greatest Lower Bound
  - (A ∩ B)(h) = min(A(h),B(h))

8

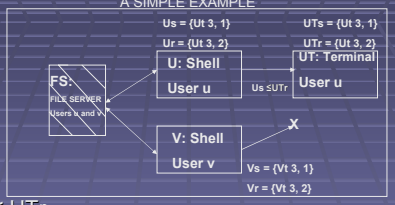
Asbestos Labels (Contd)

- Label Basics (Contd)
  - Each process in Asbestos has two labels:
    - A send label Ps
    - A receive label Pr
  - A process P may send to process Q if
    - Ps ≤ Qr
  - When the message is delivered, Qs send label is contaminated by Ps send label
    - Qs = Qs U Ps
  - In Send label: lower levels are more permissive
  - In Receive label: lower levels are more restrictive

9

Asbestos Labels (Contd)

A SIMPLE EXAMPLE



- Us ≤ UTr
  - Us(ut) = UTr(ut), U can send to UT.
- Vs is not ≤ UTr
  - Vs(vt) = 3, UTr(vt) = 2
  - V cannot send to UT

10

Asbestos Labels (Contd)

- Four Levels:
  - Default send level is 1, Default receive level is 2
  - Default labels are in the middle of the labeling order.
  - Flexible isolation schemes possible

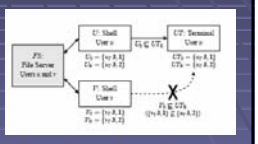
	A	B	C
Ps	{h 3,1}	{1}	{h 2,1}
Qr	{2}	{h 0,2}	{h 1,2}

11

Asbestos Labels (Contd)

Effective Labels

- Ability to taint different user processes in different ways
- Uses Contamination and Verification Labels Cs and V
  - Label Es:
    - Es = Ps U Cs
  - Label Er:
    - Er = Qr ∩ V



12

Asbestos Labels (Contd)

- Declassification Privileges
  - Uses *-level to decentralize declassification.
  - A process P with Ps(h) = *, is said to have declassification with respect to h.
  - Modified equation:
    - Qs = Qs U (Es ∩ Qs*) is same as:
      - Qs(h) = Qs(h), if Qs(h) = *
        (Qs U Es)(h), otherwise

13

Asbestos Labels (Contd)

- Decontamination
  - A process with declassification privilege can decontaminate other processes
  - Done by lowering their send labels and raising their receive labels
  - Uses two optional arguments Ds and Dr to the send system call
    - Modified Equations:
      - Es ≤ Qr U Dr
      - Qs = (Qs ∩ Ds) U (Es ∩ Qs*), Qr = Qr U Dr

14

Asbestos Labels (Contd)

- Preventing Contamination
  - To prevent processes from getting contaminated unwillingly.
    - Every port p is associated with a port receive label pr
    - This acts like a verification label imposed by the receiver rather than the sender.
    - Modified Equation:
      - Er = Qr ∩ V ∩ pr

15

Event Processes

- Handling multiple users data:
  - User level threads
  - Separate Process per user
- Simple event-driven dispatch loop:

```
while(1){
        event = get_next_event();
        user = lookup_user(event);
        if(user not yet seen)
                user.state = create_state();
        process_event(event, user);
    }
```

  - No isolation of user states.

16

Asbestos Event Process

- Isolates different event process's state.
- Each event process associated with one base process
- Event process's kernel state consists of:
  - Send label, Receive label, Receive rights for a port and a set of memory pages and book keeping information.

17

Asbestos Event Process (contd)

- A typical event process dispatch loop

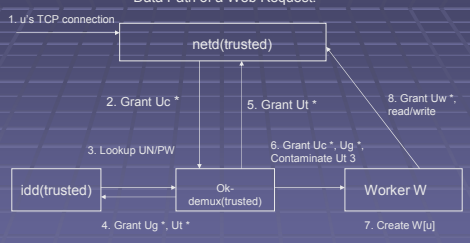
```
ep_checkpoint(&msg);
If(!state.initialized){
        initialize_state(state);
        state.reply = new_port();
}
process_msg(msg,state);
ep_yield();
```

- Uses the following system calls:
  - ep_checkpoint, ep_yield, ep_clean, ep_exit.

18

Web Server Design using Asbestos

Data Path of a Web Request:



19

Web Server Design using Asbestos

Data Path of a Web Request:

1. netd accepts incoming connection . Sets Ucr to {Uc 0, 2}
2. netd grants ok-demux Uc at level *
3. Authenticates user.
4. If authenticated, idd grants ok-demux Ut, Ug at level *
5. ok-demux grants Ut * to netd. Netd raises Ucr to {Uc 0, Ut 3, 2}
6. If the requested service exists in W, ok-demux forwards Uc, grants Ug * and contaminates it with Ut 3
7. W returns from ep_checkpoint into W(u).
8. W(u) creates new port Uw, grants it to netd at *.
9. W(u) calls ep_exit.

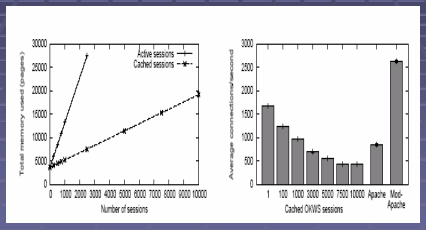
20

Performance

- Memory Use
  - Cached session: Requires additionally ~1.5 4KB pages
  - Active sessions: Requires additionally ~9.5 4KB pages
- Web Server Performance
  - Throughput
    - With one cached session, the avg no. of connections is greater than that of apache's
  - Latency
    - With 1000 cached sessions, almost same as that of apache's
- Label Costs
  - Linear degradation in performance.

21

Performance



22

Thank You!
Questions?

23