Securing Inter-process Communications in SELinux

2007 SELinux Symposium

Spencer Shimko, Joshua Brindle
Tresys Technology, LLC

Introduction

- Inter-process communication (IPC)
  - method of information exchange between processes
    - local machine, or network
  - used to share data, state, and synchronization
- Numerous forms for various use cases
  - files, message queues, shared memory, etc.
- Each has documented benefits
  - throughput, latency, ease of use, portability
- Linux, SELinux supports BSD, POSIX, and SysV IPC
  - don’t worry, no Mach Messages

The Problem

- Little to no IPC security documentation
  - current docs (man pages, books) don't address information-flow security
- How do I know if a denial is my problem or yours?
  - control options, write implies read, etc.
- Which type of IPC should I use?
  - RTFM
  - rely on intuition and experience
- Isn't there an easier way to answer these questions?

Objectives

- Security Objectives
  - provide access control
  - ensure proper directionality
  - facilitate revocation of access
  - protect integrity of both data and processes
- System objectives must also be considered
  - functional requirements
    - performance, usage
  - security goals
    - risks and threat model

Security Objectives (cont)

- Fine-grained access controls
  - fine-grained mediation of access on most IPC
    - controls read, write, control options, etc
    - overloaded, userspace supported the exception
- Revocation
  - must be able to rescind access previously granted
  - SELinux supports access time revocation on most IPC
    - “memory based” IPC is the exception
      - attributed to kernel architecture, unlikely to change

Security Objectives (cont)

- Unidirectional information flow
  - not just for guard/CDS use
    - pipes, FIFO, datagrams
  - hard to achieve
    - guarantees of delivery almost always necessary
    - numerous backchannels exist, such as blocking
- Integrity
  - SELinux provides integrity
    - fine-grained access control

Pipes & Unix Domain Sockets

- Pipes & FIFOs (named/unnamed Pipes)
  - SELinux *can* enforce directionality but…
    - non-blocking storage channel via partial writes
    - FIFOs also have flow throw the file system
  - access time revocation supported
- Unix Domain Sockets
  - creation bitmap
  - channels through files
  - access time revocation supported

Shared Memory & Message Queues

- Shared memory
  - access controls fine-grained
  - possible to enforce directionality
    - synchronization in the unidirectional case?
  - permissions migrated
- Message queues
  - access controls eliminate the large back channels
    - blocking back channel exists
    - helper process eliminates “bitmap” attack vector

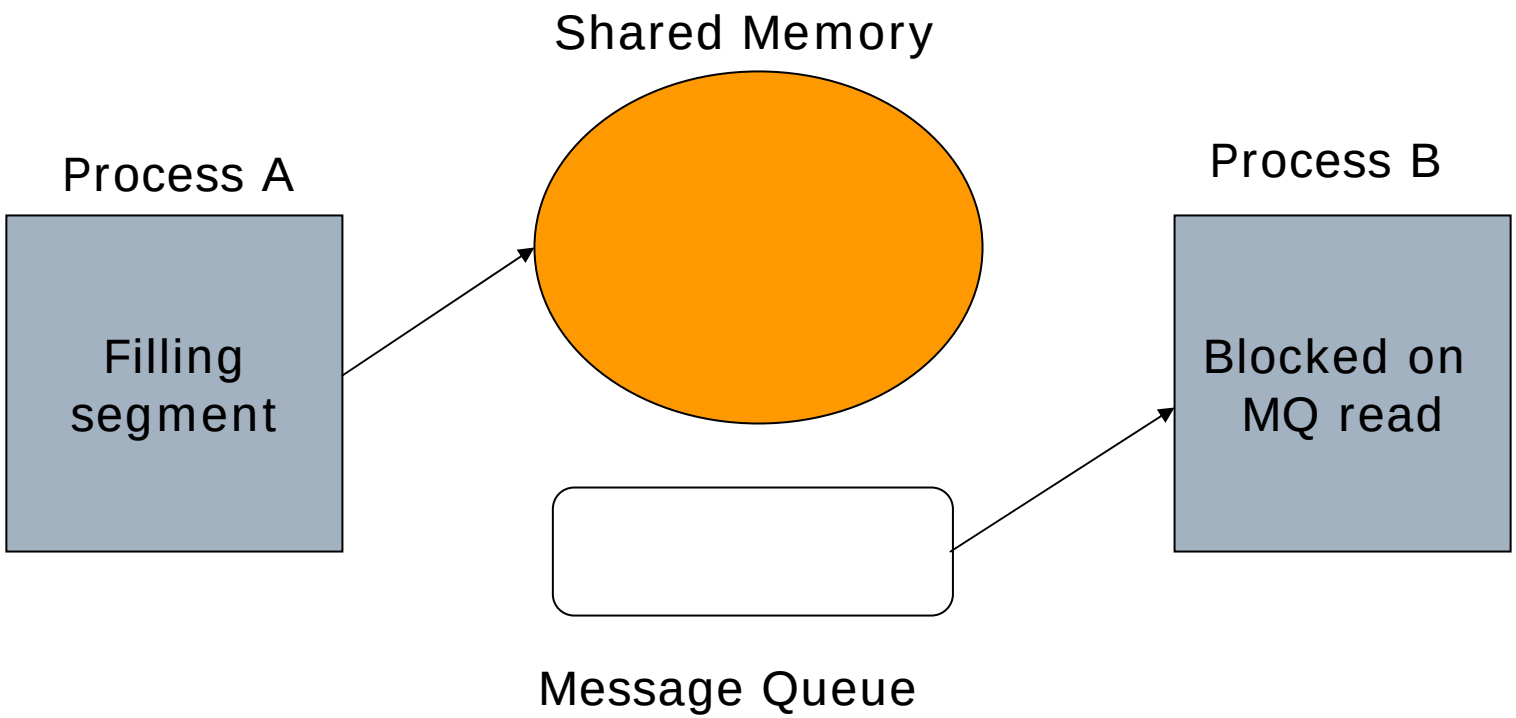
Building a Secure IPC Mechanism

- Unidirectional shared memory needs synchronization or control channel
- Message queues suit that need
- Shared memory acts as large bandwidth data channel
  - forward only
- Message queues act as control channel
  - small, noisy back channel via blocking

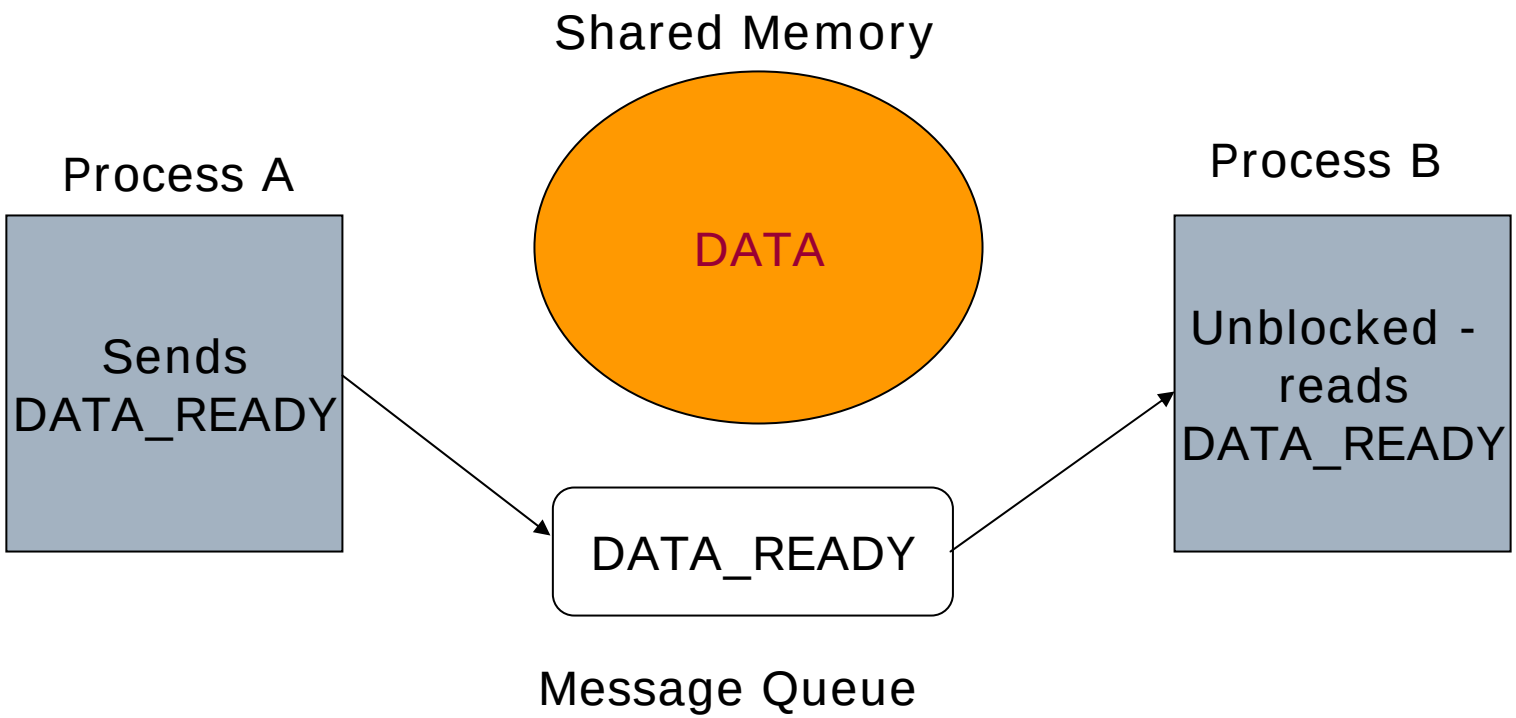
Using IPC Securely

- Helper process
  - starts, creates queue, terminates
  - eliminates bitmapping attack vector
    - neither process can create queues
  - queue retains the type of this process
- Participating processes
  - associate with the queue created above
  - process A creates a shared memory segment and attaches
  - process B attaches

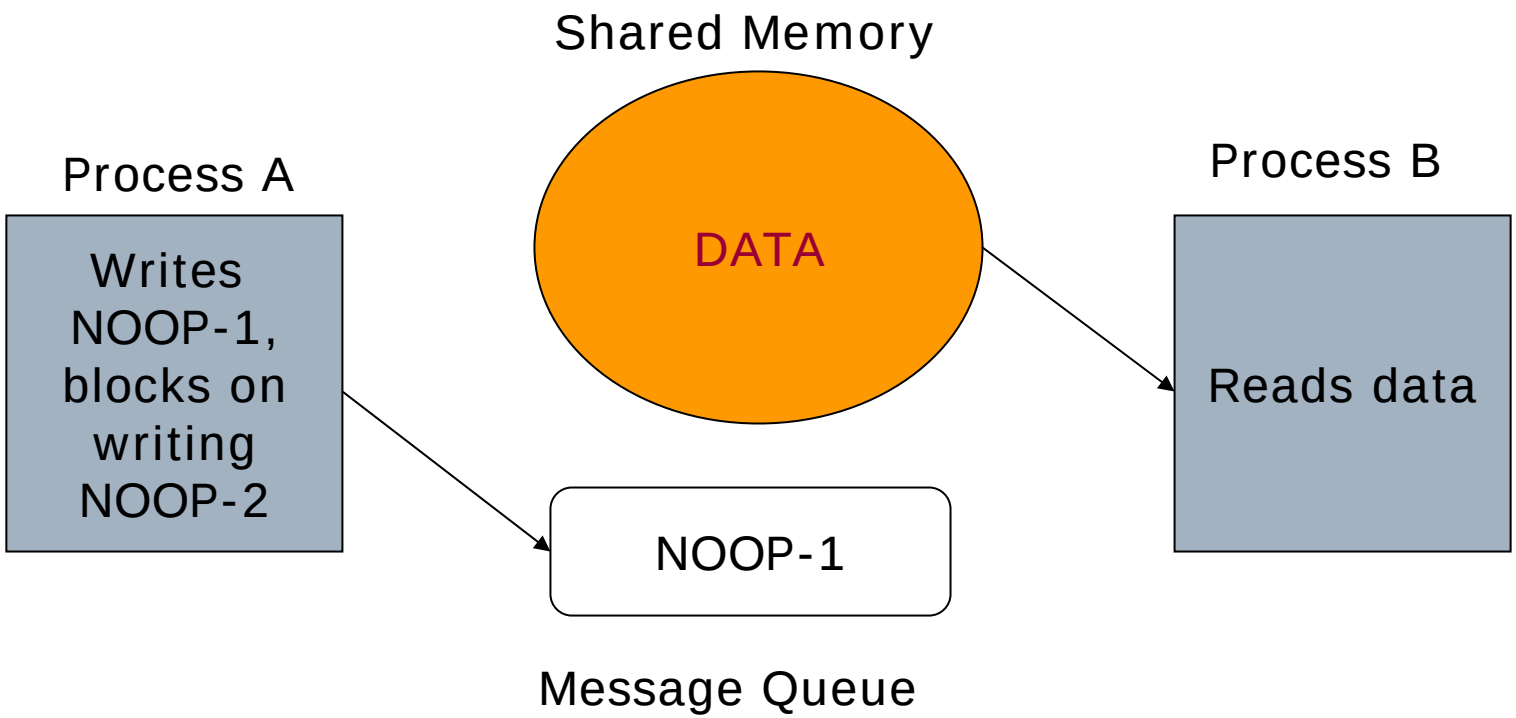
Using IPC Securely- not drawn to scale



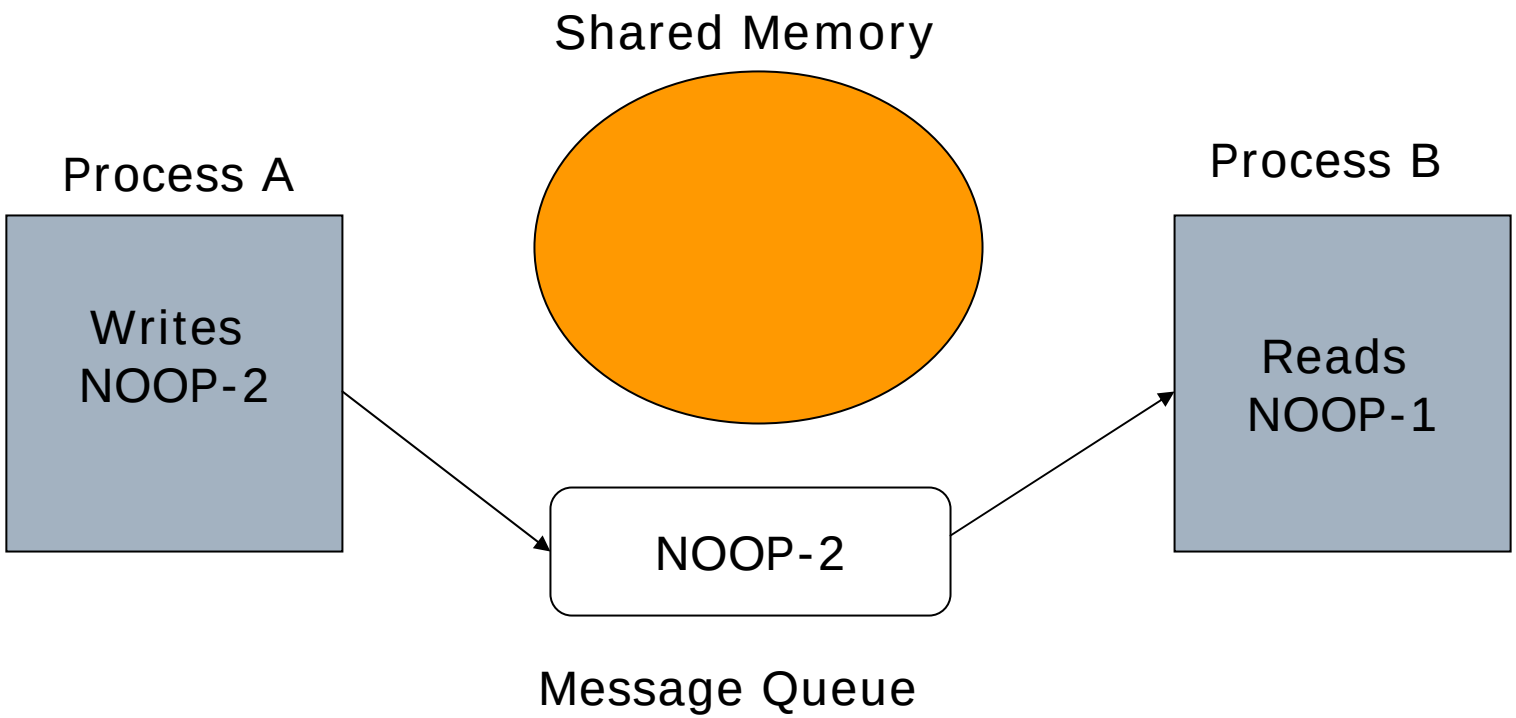
Using IPC Securely- not drawn to scale



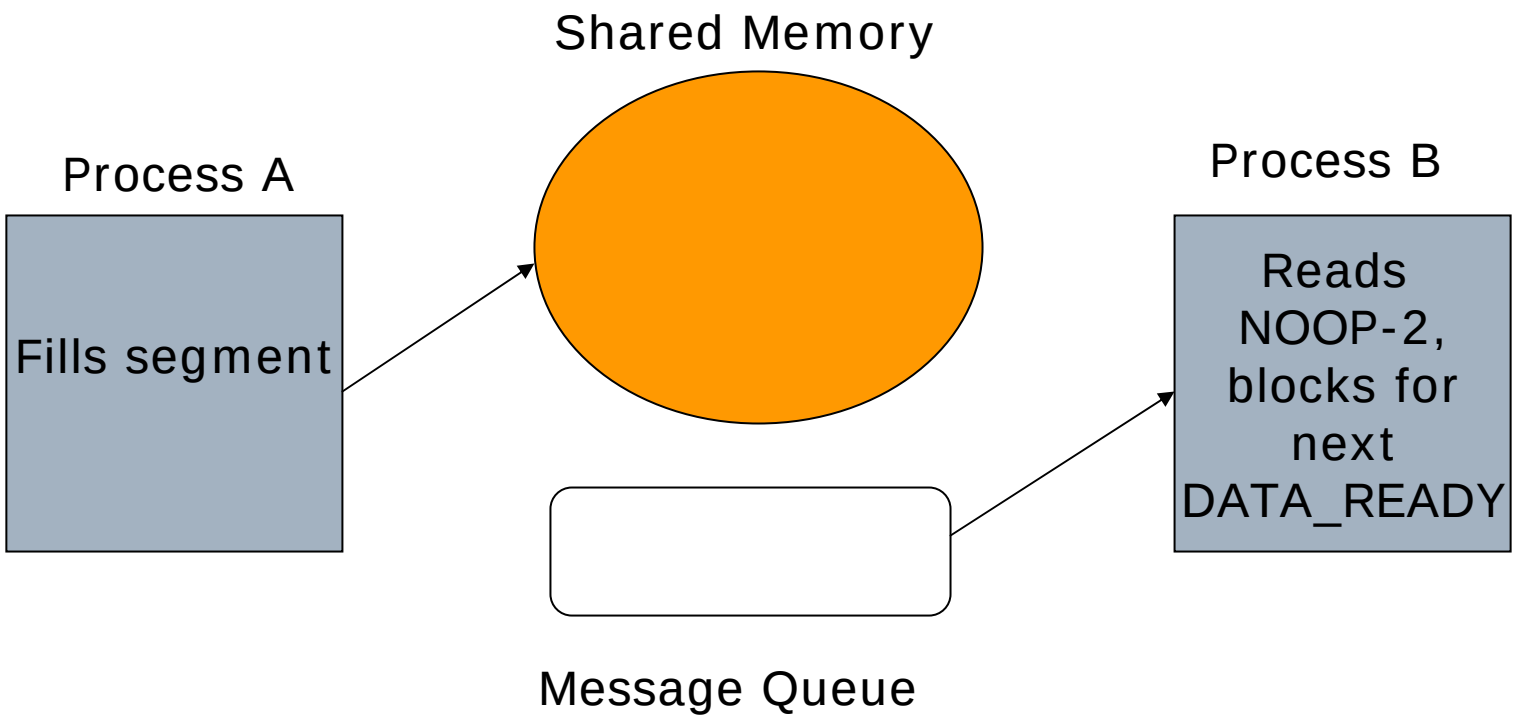
Using IPC Securely- not drawn to scale



Using IPC Securely- not drawn to scale



Using IPC Securely- not drawn to scale



Using IPC Securely

1. Wash
2. Rinse
3. Repeat

Conclusions

- IPC is not unidirectional by definition
  - SysV, POSIX, BSD APIs compliance requires backchannels
- In SELinux, message queues and shared memory
  - a large bandwidth forward channel
  - a limited bandwidth back channel
  - enforced with a MAC policy
- C library, Java and Mono bindings, and policy available
  - http://oss.tresys.com/projects/sipc

Questions?

http://oss.tresys.com/projects/sipc