NetLabel: What, Why & Where

Paul Moore

paul.moore@hp.com

Agenda

- What is NetLabel?
  - Introduction to security labels
  - Applying security labels to the network
- Why NetLabel?
  - Labeled IPsec isn't for everyone
  - NetLabel capabilities
- Where is NetLabel?
  - Architectural overview
  - LSM integration

What is NetLabel?

NetLabel in a Nutshell

- Framework for labeled networking
  - Supports multiple Linux Security Modules (LSM)
    - LSMs based on security labels (SELinux, Smack)
  - Supports multiple labeled networking protocols
- Extends LSM security policy to the network
  - Network traffic is subject to security policy
    - Network packets are assigned security labels
  - Access control performed inside the LSM
    - No need for separate network security policy

What are Security Labels?

- Represent an entity's security properties
  - Security labels summarize:
    - What actions a process is allowed to perform
    - The sensitivity of the data stored in a file
- Assigned to everything on the system
  - Subjects (things that *act* on other things)
    - e.g. processes, users, etc.
  - Objects (things that *are* acted upon)
    - e.g. files, devices, etc.

What is Labeled Security?

- Access decisions based on security labels
  - Simplified access control based on:
    - Subject's security label (foo)
    - Object's security label (bar)
    - Type of access (read a file)

      "Is foo allowed to read a file labeled bar?"
  - Security labels are a prerequisite for access control
- Permissions defined by security policy
  - Security policy, not users, defines access rights
  - Single security policy applied system-wide

Labeled Security and Networking

- Security label bindings must be preserved
  - Persistent entities require persistent labels
    - e.g. filesystem and user security labels
  - Kernel maintains labels for transient entities
    - Dynamic, in-memory objects with short lifetimes, e.g. sockets
    - Subjects with short lifetimes, e.g. processes
- Propagating labels across the network
  - Protocols assign security labels to network data
    - Some provide label visibility to intermediate nodes
    - Static labeling when protocol support is not possible
  - Peers are able to determine sender's label
    - Allows security policy to be applied to network data

The Protocols Behind NetLabel

- Commercial IP Security Option (CIPSO)
  - IPv4 labeling protocol for Multi-Level Security (MLS)
    - Label carries hierarchical level and compartment bitmap, not the string formats used by existing LSMs
  - Widely supported on existing Trusted Systems
    - e.g. Trusted Solaris, HP-UX CMW, etc.
  - Defined in FIPS-188, not an approved IETF RFC
    - Created by the Trusted Systems Interoperability Group (TSIG)
- Static network labeling (the protocol that isn't)
  - Implicitly labels network traffic using sender's address
    - Administrator assigns static security labels to systems
    - Supports both IPv4 and IPv6 networks
  - Allows system to interoperate with unlabeled peers

NetLabel in a Nutshell *(Revisited)*

- Kernel subsystem for network security labels
  - Manages security labels on sockets and packets
    - Abstracts protocol specifics away from the LSM
    - Allows the LSM to focus on enforcing security policy
  - Parses and converts on-the-wire security labels
    - LSM and protocol independent security label format
- Requires integration with each LSM
  - Security label conversion and required network hooks
    - LSM implements access controls based on its security model
  - NetLabel only functions when enabled by a LSM
- Configured either by the LSM or userspace tools

Why NetLabel?

What about Labeled IPsec?

- Transfers security labels across the network
  - Labels propagated during IKE negotiation
    - Security labels protected by IKE phase-2 encryption
    - Both peers agree on labels before sending data
  - Labels assigned to Security Associations (SA)
    - Traffic is implicitly labeled based on matching SA
- Authentication, confidentiality, integrity
  - IPsec protections apply to both data and labels

The Problems with Labeled IPsec

- Interoperability with existing Trusted Systems
  - Not supported on deployed systems
    - Labeled networking provided through other protocols
  - Lack of visible security label
    - Intermediate nodes are unable to inspect the label
  - Non-standard/conflicting use of IKE protocol
- Performance and scalability
  - Additional per-packet processing overhead
    - Labeling requires IPsec on the communication endpoints
  - Potential for substantial increase in number of SAs
    - Single SA can only represent a single security label

The Arguments for NetLabel

- Interoperability with existing Trusted Systems
  - Demonstrated CIPSO interoperability
  - Allows integration of Linux into existing networks
- Flexible network encryption options
  - Works with, but does not require, IPsec
    - IPsec can protect CIPSO based security labels
    - IPsec processing can be offloaded to hardware or other nodes
  - Potential for custom network encryptors
- Designed with future expansion in mind
  - Abstractions enable new protocols with minimal impact
  - Well defined API boundary eases LSM integration

NetLabel Capabilities

- Flexible labeling configurations
  - Different configurations per subject label
    - Allows one security domain to send unlabeled traffic while another domain sends CIPSO labeled traffic
  - Additional granularity possible based on destination
    - Allows subjects to interact with differently labeled networks
- Support for both local and forwarded traffic
  - NetLabel supports labeling forwarded traffic
    - Not available in Smack due to security model philosophy
  - Enables Linux as a labeled network gateway
    - Route and label traffic among differently labeled networks

Current Drawbacks to NetLabel

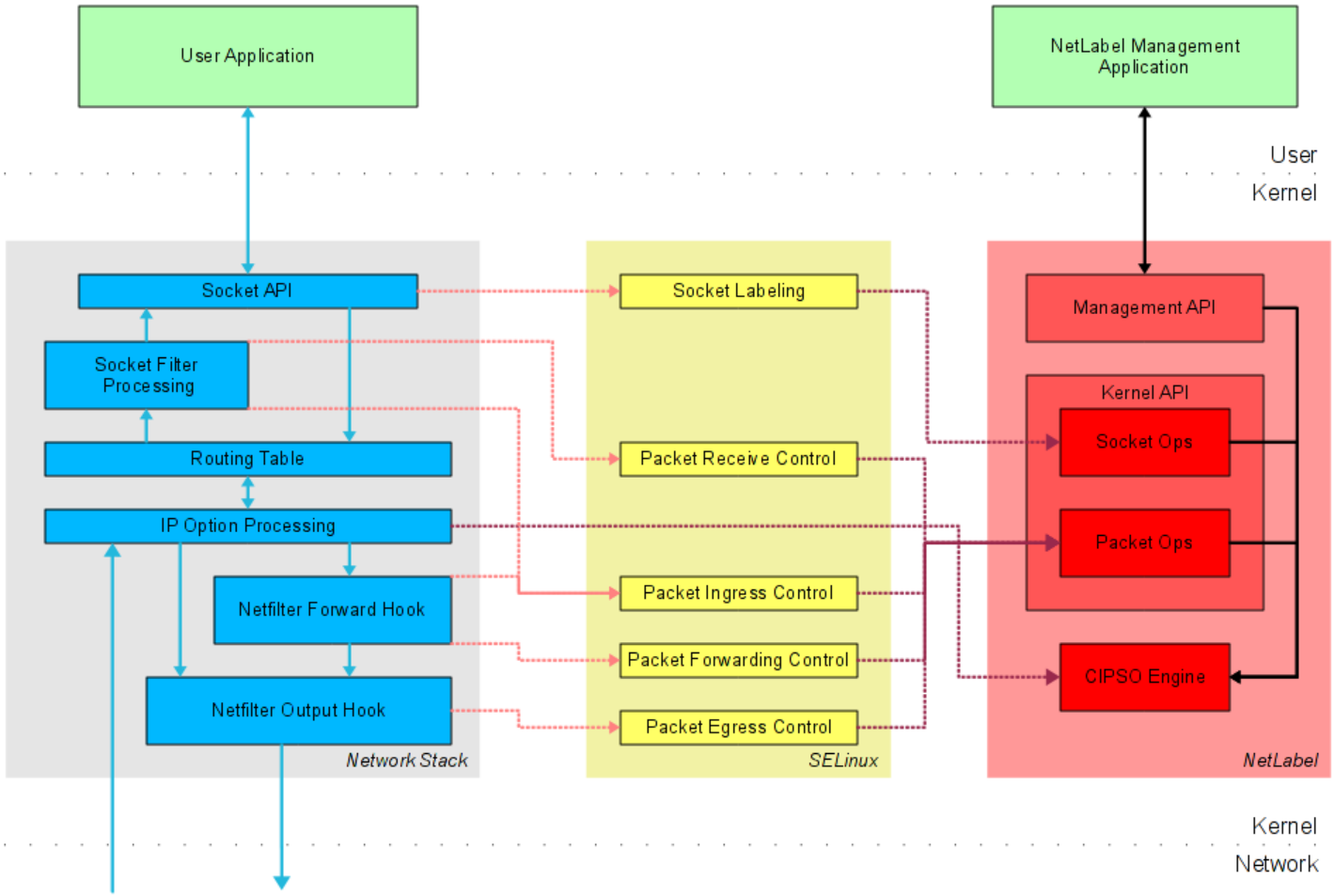
- Labeling protocol support limited to IPv4
  - Not a fundamental design problem
    - Static labeling supports IPv6 network traffic
  - IPv6 labeling RFC only recently published
    - CALIPSO (RFC5570) support is currently “in progress”
- Limited to MLS portion of the SELinux label
  - Not a fundamental design problem
    - Full SELinux label support *does* exist for loopback traffic
  - Lack of an accepted spec for string based labels
    - Requires a new spec or extensions to existing specs

Where is NetLabel?

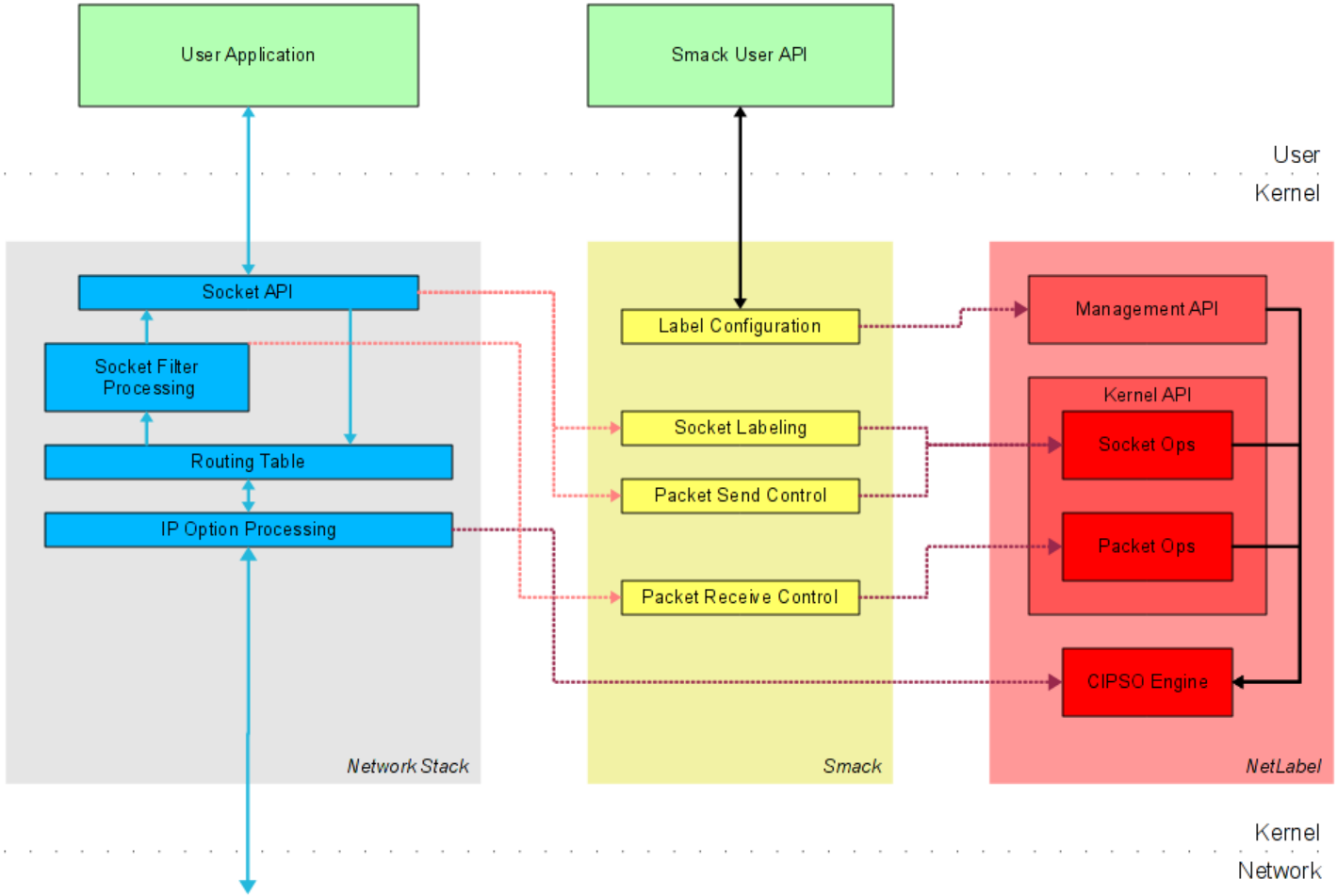
NetLabel Architecture

- Designed to tread lightly on the Kernel
  - No additional hooks in the main code paths
    - One CIPSO specific validation call in the IPv4 option path
    - Leverages existing LSM networking hooks
  - Code is well contained and easily disabled
    - CIPSO protocol lives in a single file in the IPv4 stack
    - NetLabel code lives in a separate, dedicated directory
- LSM independent labeled networking framework
  - LSM specific code lives within the LSM itself
    - Handles importing/exporting of NetLabel security labels
  - Abstracts protocol specifics away from the LSM
    - Minimizes the necessary LSM specific “glue” code

NetLabel and SELinux



NetLabel and Smack



NetLabel Userspace

- Single command line management tool
  - Talks directly to NetLabel management API
    - Independent of any LSM configuration methods
    - Utilizes Generic Netlink for Kernel communication
  - Built on top of a separate NetLabel library
    - Allows for third-party NetLabel management tools
    - Fully documented API via doxygen
- Project hosted on SourceForge
  - Information and code at *http://netlabel.sf.net*
  - Available in distributions as “netlabel_tools”

Additional Information

NetLabel Resources

- NetLabel project page on SourceForge
  - http://netlabel.sf.net
- Presenter's blog on LiveJournal
  - http://paulmoore.livejournal.com
- Presenter's contact information
  - paul.moore@hp.com


invent