SELinux Network Access Controls

Paul Moore
Red Hat

December 2012

What We Cover In These Slides

- Different types of SELinux network labels
- SELinux network access control points
- Configuration basics

The Fine Print

Disclaimer: RHEL5 is Really, Really Old

The information presented here applies only to RHEL6. While some of the high level concepts may apply to RHEL5, most only apply to RHEL6 or newer (e.g. Fedora) systems.

Disclaimer: Socket Level Controls are Boring

- SELinux has both network packet and socket controls
  - Socket controls
    - Ability to restrict operations between application and socket
      - e.g. bind(...), connect(...), read(...), write(...)
  - Packet controls
    - Ability to restrict data flow between socket and the network
      - Sockets are considered the communication endpoints
      - Network interface and node labels represent “the network”
- These slides will focus mostly on the packet controls
  - It is where all the interesting stuff happens anyway

SELinux Network Labels

Network Traffic Labels: Twice the Fun, Confusion

- Network traffic can have two different types of labels
  - Peer labels
    - Represent the security attributes of the traffic's sender
      - e.g. label is the security label of the sender, “firefox_t”
  - Secmark labels
    - Represent the network attributes of the traffic
      - e.g. label is a value set by an iptables rule, “web_packet_t”
- Both labels can exist on the traffic simultaneously
- Both labels have separate access controls

Network Peer Labels: Proper Labeled Networking

- Peer labels are sent across the network via a protocol
  - Labeled IPsec
    - Leverages existing IPsec protections: auth, encryption, etc.
    - Transmits full SELinux label (user,role,type,MLS/MCS)
    - Performance and scalability impacts
  - NetLabel / CIPSO
    - Based on FIPS standard, interoperable with other OSs
    - Transmits MLS/MCS attribute only, user/role/type are fixed
    - Labeling protocol only, minimal to no performance impact
  - NetLabel / Fallback
    - Not a labeling protocol, a fallback for unlabeled networks
    - Assign a single peer label to traffic from a network interface

Secmark Labels: The Other Guy

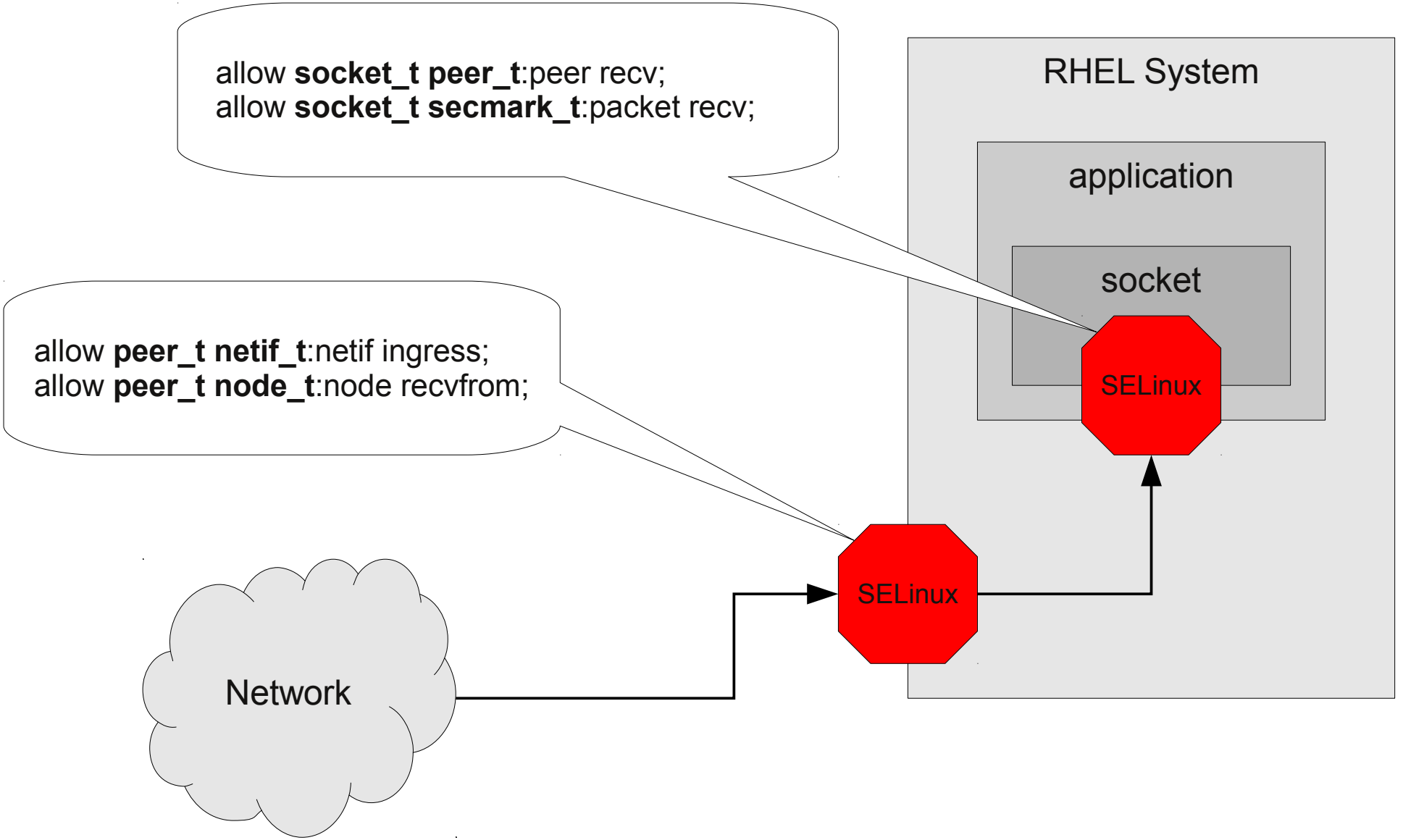
- Represents only the packet's network attributes
  - Does not take into account any security attributes
- Leverages the iptables/netfilter mechanism
  - Assigns labels to traffic based on iptables rules
    - Uses the SECMARK and CONNSECMARK targets
  - Powerful and efficient traffic matching mechanism
    - Same code as the Linux Kernel's firewall
  - Labels exist only on the local machine

Interface and Node Labels: Labeling the “Cloud”

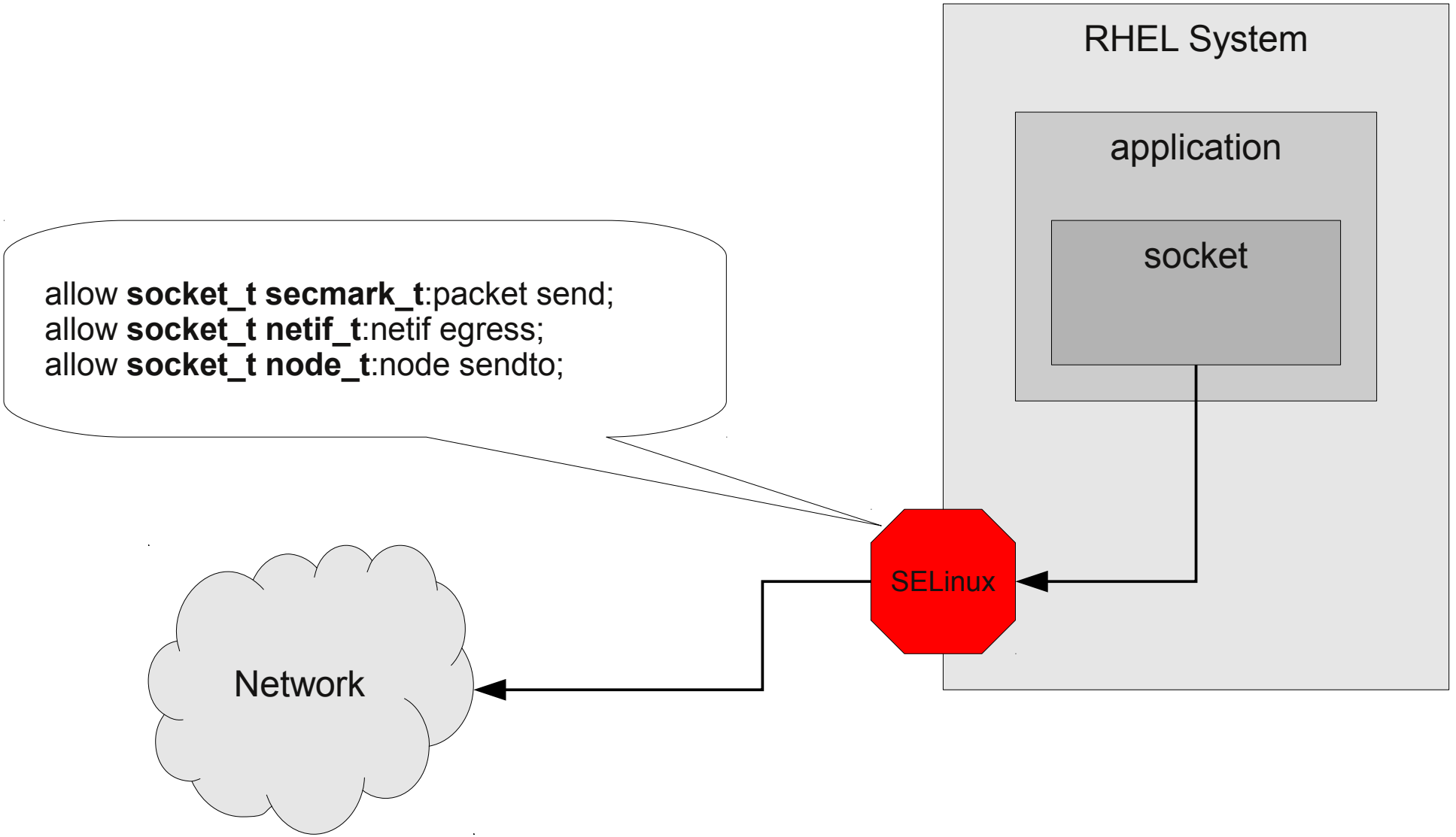
- Network interface labels
  - Both physical and logical interfaces can be labeled
  - Allow us to limit traffic flowing through a network
    - e.g. restrict “Top Secret” traffic to a “Top Secret” network
- Node labels
  - Allow us to limit traffic to and from a particular node
  - Assumes IP addresses are trustworthy (they aren't)
    - Useful for private, trusted networks and development
- Both interface and node labels are set via semanage

SELinux Network Access Control Points

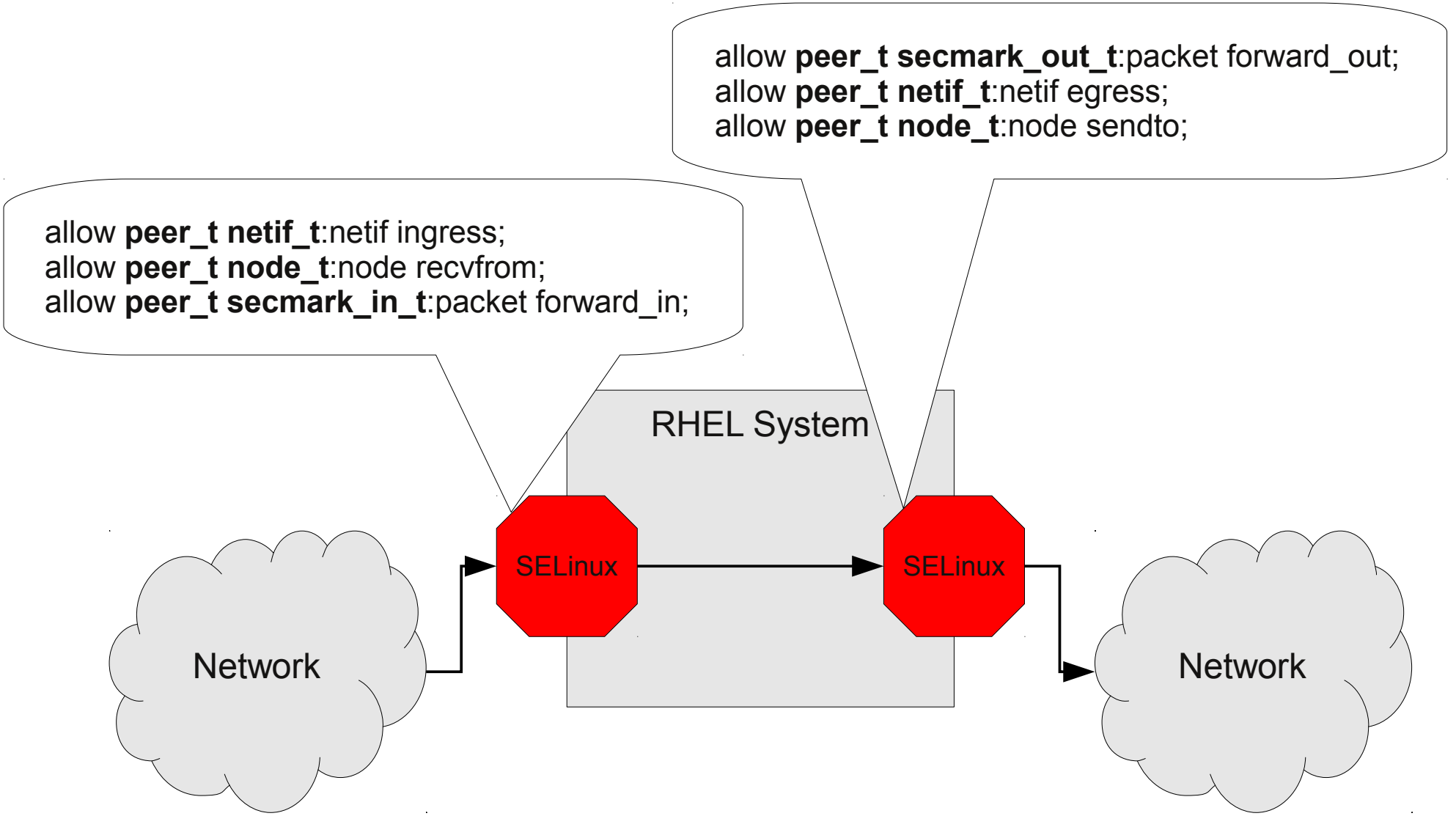
A Day in the Life: Inbound Packets



A Day in the Life: Outbound Packets



A Day in the Life: Forwarded Packets



Configuration Basics

Dynamic Controls: The First Step is a Doozy

- Modern SELinux network access controls are dynamic
  - Kernel access control checks are disabled until needed
    - No point in access control when everything is unlabeled
  - Improves performance and simplifies policy
- Network access controls triggered by configuration
  - Peer controls: labeled IPsec or NetLabel configuration
  - Secmark controls: iptables rules
- Network access controls are triggered system wide
  - Be careful when “flipping the switch” for the first time

Interface and Node Labels: The semanage Tool

- Example: adding an interface label to eth0
  - **semanage interface -{a|m|d} -t netif_t -r s1:c3 eth0**

- Example: adding a node label to 10.10.3.1
  - **semanage node -{a|m|d} -t node_t -r s1:c3 10.10.3.1**

- More information
  - Man page for “semanage”

Secmark Labels: The iptables Tool

- Example: label all TCP/22 (SSH) packets
  - **iptables -t mangle -A INPUT -p tcp --dport 22**
    -j SECMARK
    --selctx system_u:object_r:my_ssh_packet_t:s0

- More information
  - Man page for “iptables”

NetLabel: Configuration Overview

- NetLabel configuration divided into two stages
  1. Define the labeling protocol configuration
     a) CIPSO: define the CIPSO Domain of Interpretation (DOI)
     b) Fallback: define the static network labels
  2. Define the mapping of protocols to outbound traffic

- More information
  - Man page for “netlabelctl”

NetLabel: Protocol Configuration

- Example: add a CIPSO DOI
  - **netlabelctl cipsov4 add pass doi:1 tags:1**

- Example: add a fallback label to eth0
  - **netlabelctl unlbl add**
    interface:eth0 network:0.0.0.0/0
    label:system_u:object_r:my_peerlbl_t:s0
  - **netlabelctl unlbl add**
    interface:eth0 network:::0/0
    label:system_u:object_r:my_peerlbl_t:s0

NetLabel: Outbound Traffic Mapping

- Example: map CIPSO DOI #1 to all traffic from “foo_t”
  - **netlabelctl map add domain:”foo_t” protocol:cipsov4,1**
- Example: send CIPSO traffic from “bar_t” to 1.2.3.0/24, unlabeled otherwise
  - **netlabelctl map add domain:”bar_t” address:1.2.3.0/24 protocol:cipsov4,1**
  - **netlabelctl map add domain:”bar_t” address:0.0.0.0/0 protocol:unlbl**
  - **netlabelctl map add domain:”bar_t” address::::0/0 protocol:unlbl**

Labeled IPsec: Configuration Overview

- Labeled IPsec support included in Openswan
- Labeled IPsec configuration divided into two stages
  1. Configure IPsec as you would normally
     - **Way** beyond the scope of this presentation
  2. Add a security label and labeled flag to the connection

- More information
  - Man page for “ipsec.conf”

Labeled IPsec: Connection Block Configuration

- Example: add the ipsec_spd_t label to a connection

```
conn labeled_example
        auto=route
        rekey=no
        authby=secret
        type=transport
        left=10.10.2.4
        right=10.10.2.5
        ike=3des-sha1
        phase2=ah
        phase2alg=sha1
        labeled_ipsec=yes
        policy_label=system_u:object_r:ipsec_spd_t:s0-s15:c0.c1023
```


More Information

More Information: According to Google

- My contact information
  - Paul Moore, pmoore@redhat.com
- My blog on SELinux network access controls
  - http://paulmoore.livejournal.com
  - Not updated very regularly, but is a good source of info
- NetLabel project page on SourceForge
  - http://netlabel.sf.net
  - NetLabel tool sources, fairly technical