Labeled IPsec Phase 1: Label-aware SADB Design

Bill Sommerfeld

IPsec I-Team
ipsec-core@sun.com
Solaris Security
Sun Microsystems, Inc.

Comments on this document should be sent to:
security-discuss@opensolaris.org

Revision 0.6
January 20, 2009

Contents

1 Introduction **2**

2 Use Cases **2**

3 Rationale **2**

4 Labels, Explicit and Implicit **3**

5 Specific Changes **3**

5.1 PF_KEY . . . 3
5.2 SADB . . . 4
5.3 Trusted Networking Packet Processing . . . 4
5.4 IPsec Packet Processing . . . 4
5.4.1 Outbound SA Selection . . . 4
5.4.2 Outbound Traffic Labeling . . . 4
5.4.3 Inbound Decapsulation . . . 5
5.5 in.iked . . . 5
5.5.1 Single vs. Multi-label . . . 5
5.5.2 Wire encoding . . . 6
5.6 ipseckey . . . 6
5.7 Packaging . . . 6

Glossary **9**

1 Introduction

This document describes a proposed set of changes to OpenSolaris to add a minimal Labeled IPsec capability to the kernel IPsec code, and a related set of changes to the (closed-source) Solaris IKE daemon to provide minimal support for managing and negotiating sensitivity labels between systems under common management. This project is intended to lay the groundwork for follow-on work involving greater flexibility both in label policy and in the types of labels handled; however, support for other types of labels is out of scope for this phase.

2 Use Cases

The current TX database template structure ties policy to ip addresses and conflates policy (the clearances of each node) with mechanism (the use of CIPSO or null representation of labels on the wire). This project will start to pare this apart within the implementation but, to avoid entaglement with other projects, will leave the databases unchanged.

Accordingly, the primary target for Phase 1 Labeled IPsec is a network consisting of a single-DOI core of multi-level systems at fixed ip addresses under common administration with some number of single-level, single-label systems on the periphery. It will be possible to use standard IPsec without any exotic extensions (such as `SIT_SECRECY`) to protect communications with these single-label systems. In some cases, this should work even in the presence of middleboxes which add and remove CIPSO labels.

Note that future phases may weaken these constraints. Note also that two multi-label systemsadministration may be configured to treat each other as single-label.

The existing Trusted Extensions networking database structure will be used as-is, but with IPsec configuration affecting the on-the-wire protocol.

Two distinct operating modes will be provided:

1. Implicit-only

   Key management and IPsec-protected traffic will not carry CIPSO or other explicit labels, permitting it to transit CIPSO-unaware or CIPSO-hostile networks.

2. CIPSO+IPsec

   IPsec-protected traffic and key management traffic may also carry CIPSO labels. CIPSO labels in received packets will be validated against the SADB. The on-the-wire CIPSO label used for protected traffic is under the control of key management, and may either match the internal label of the traffic, or be assigned by key management.

Mode selection will be controlled by a directive in the IKE configuration file that, like other IKE rules, will be configurable on a per-peer-address basis.

Multi-level transport-mode and tunnel-mode traffic can be protected using IPsec implicit labeling.

3 Rationale

The constraints on this project are intended to simplify the phase 1 effort while producing a feature set which should be useful to at least some customers. Avoiding changes to the tn*db schema and

other significant changes to trusted networking greatly reduces the scope of the project; while in the long run we will need to revisit this, we will be able to get this phase out more quickly as a result.

Similarly, there will be no changes to the IPsec SPD schema as part of this project, while there will be some extensions to the `PF_KEY` interface as well as to the in.iked configuration file syntax.

4 Labels, Explicit and Implicit

It is important to distinguish between the sensitivity label of a packet and the way in which that label is represented (both internally and on the wire).

Internally to the kernel, each `dblk_t` contains a `cred_t *db_credp` field which may point to the credentials (including the label) associated with the message.

On the wire, the label of a packet may be explicit; for instance, it may be contained in a CIPSO option in the IP header. It may also be implicit; for instance, it may be implied by the peer address (if the peer is unlabeled).

This project adds an additional form of implicit labeling in which the sensitivity label is a property of the IPsec security association used to protect the traffic.

5 Specific Changes

5.1 PF_KEY

One new `PF_KEY` extension type is necessary: the `SADB_X_EXT_OUTER_SENS` extension. The standard `SADB_EXT_SENSITIVITY` extension describes the sensitivity of the traffic being carried. Our new SADB_X_EXT_OUTER_SENS extension defines the sensitivity of the packet as modified by ESP and/or AH, which may differ from the sensitivity of the original plaintext packet. If ESP is used with encryption to provide data confidentiality, the label contained in the original `SADB_EXT_SENSITIVITY` extension reflect the sensitivity of the plaintext while the label in the new `SADB_EXT_OUTER_SENS` extension reflects the sensitivity of the ciphertext.

In addition, we have redefined the `sadb_sens_reserved` pad field in the `sadb_sens_t` as a flags word (with a backwards-compatible `#define`); one flag bit, `SADB_X_SENS_IMPLICIT`, is defined in that flags word. It is valid only in `SADB_X_EXT_OUTER_SENS` extensions. When set, it tells the kernel to omit explicit packet labels from the outer packet header of the ciphertext packets; the outer label is still used to constrain the choice of next-hop router.

Care must be taken with outer sensitivity labels in the event that no confidentiality protection is requested (either by the use of AH alone, or the use of ESP without encryption).

A future phase may provide for kernel-level constraints on the relationship between the inner label, the outer label, and requested security services (perhaps allowing the kernel to require that the outer label dominate the inner label if confidentiality is not used). Note that key management runs only in the global zone and is part of the TCB already; kernel-level constraints would thus serve only as an additional check on an already trusted process. Such constraints are out of scope for this phase.

5.2 SADB

The SADB will now associate and store the sensitivity label component of the standard (but rarely implemented) `SADB_EXT_SENSITIVITY` extension and the new `SADB_X_EXT_OUTER_SENS` outer sensitivity extension as attributes of each security association.

If the inner sensitivity extension is not present, the SA will be considered unlabeled and, for compatibility with existing behavior, may carry traffic under multiple labels. Precisely how the label is carried in this case is TBD, but will most likely involve a CIPSO option protected by either AH or ESP integrity protection and perhaps ESP confidentiality protection.

5.3 Trusted Networking Packet Processing

In phase 1, the only change required to trusted networking policy is the addition of a bypass socket option, `SO_MAC_IMPLICIT`. This option is used by in.iked when operating in the implicit-only labeling mode to prevent key management traffic packets from carrying explicit labels.

The new `NET_MAC_IMPLICIT` privilege is required to permit use of this option. In phase 1, `NET_MAC_IMPLICIT` will only be available in the global zone, and therefore on labeled systems, key management daemons must run in the global zone. (Note that exclusive IP instances – the only way that in.iked can run in a non-global zone today – do not currently work with TX, and this project does not propose to change this).

Note also that we contemplated overloading the existing NET_MAC_EXEMPT privilege to also enable SO_MAC_IMPLICIT but reviewers were uncomfortable with this additional use of NET_MAC_EXEMPT.

Outbound traffic from sockets with SO_MAC_IMPLICIT set will be subjected to all usual outbound MAC checks, but the packet will be transmitted without an explicit label will not be inserted into the packet.

Inbound traffic from multi-label hosts not bearing an explicit sensitivity label will be assigned the highest sensitivity label allowed for the host in the tn*db and processed accordingly.

5.4 IPsec Packet Processing

Currently, outbound AH and ESP processing happens after trusted networking policy is enforced.

IPsec packet processing involving the SADB will take into account the sensitivity label of traffic.

5.4.1 Outbound SA Selection

Security Association selection for outbound traffic will include the inner sensitivity label as part of the match; an unset inner label will be treated as a wildcard label.

5.4.2 Outbound Traffic Labeling

If an outbound SA has an outer sensitivity label, that label is substituted for any existing packet label in either or both of the `dblk_t` and in the IP header during outbound IPsec processing. If the `SADB_X_SENS_IMPLICIT` flag is set in the `PF_KEY` label extension, no explicit label (other than the SPI) is used in the packet.

5.4.3 Inbound Decapsulation

If an inbound SA has an outer sensitivity label, inbound traffic must be labeled consistently with that label.

If an inbound SA has an inner sensitivity label, traffic exiting it is marked with that label, superceding any previous label derived from a CIPSO option in the outer header or from the tn*db.

The key management daemon is responsible for assigning an appropriate label to the security associations it creates.

The outer label describes the sensitivity of the ciphertext. An attacker needs to break the encryption or learn the decryption key to learn the plaintext, so it would seem appropriate for the plaintext label to be different from the ciphertext label in some environments.

The key management daemon (running in the global zone and part of the TCB) is responsible for ensuring that the inner label is allowed to be used by the peer. We could add an additional "belt and suspenders" check to the kernel SADB to ensure the label is allowed to the peer but it doesn't seem necessary in phase 1 given the global-zone-only limitation on in.iked and would require extending the tn*db to carry additional policy.

5.5 in.iked

in.iked is one of several possible key management daemons. It is based on a closed-source toolkit.

In project phase 1, in.iked merely transports sensitivity labels between kernels without inspecting them in any way. It does not do any sanity checking or access controls or any correlation between the label and the peer's authenticated identity.

Clearly in.iked could be doing these sorts of access controls but some environments may not need them.

Several new keywords in `/etc/inet/ike/config` will be defined by this project. Because this project is inherently experimental, the IKE config extensions will initially be given a Volatile stability level, with the intent that they would be upgraded to Committed given successful deployment experience.

5.5.1 Single vs. Multi-label

The ike configuration file will provide ways to select between two modes of operation on the basis of the remote IP address:

1) Peers identified as `multi-label` in the ike config file will be assumed to use the SIT_SECRECY IKE situation flag and will specify and accept sensitivity labels.

2) Peers identified as `single-label` (the default) will not use SIT_SECRECY; instead, the label to use for the peer address will be extracted from the trusted extensions host database.

At most one of `multi-label` or `single-label` may appear in a rule. In addition, the directive may appear at top level to define a global default; if not present, IKE will treat all peers as single-label.

5.5.2 Wire encoding

In phase 1, the on-the-wire encoding to be used for labels will be set in a global IKE configuration parameter.

One of the following may appear:

1. `wire-label inner`

   Use the inner label of traffic in the on-the-wire CIPSO label of ESP and AH packets created by the key management daemon. IKE Key management traffic is sent as Admin_Low

2. `wire-label` *`label`*

   Use the specified label in the on-the-wire CIPSO label of ESP and AH packets created by the key management daemon. IKE Key management traffic is sent as that label.

3. `wire-label none` *`label`*

   Omit CIPSO labels from the IP header entirely. Route ciphertext and key management traffic as if it had the specified label.

In the foregoing, *label* may be either a binary label in hexadecimal form (such as `0x0002-08-08`) or a string-form label in quotes (such as `"PUBLIC"` or `"SANDBOX PLAYGROUND"`)

5.6 ipseckey

The ipseckey(1m) command will be enhanced to display inner and outer label attributes.

When sensitivity label attributes are present, the `ipseckey dump` command will include human-readable output similar to:

```
SNS: Plaintext Sensitivity DPD 1, sens level=2, integ level=0, flags=0
SNS:  Plaintext Sensitivity Label: PUBLIC (0x0002-08-08)
OSN: Ciphertext Sensitivity DPD 1, sens level=0, integ level=0, flags=2
OSN:  Ciphertext Sensitivity Label: ADMIN_LOW (ADMIN_LOW)
```

5.7 Packaging

This project introduces no new packaged deliverable files to solaris – all of the new functionality is contained in existing files.

However, one packaging change seems potentially helpful in the context of this project. The closed source in.iked and its toolkit library libike are currently delivered as part of core packages. Splitting IKE-related closed-source-derived binaries into a separate package could simplify delivery and use of the prototype.

Historically, IKE was kept with the kernel IPsec bits because if we extended PF_KEY, we wanted any corresponding IKE modifications to come along for the ride. But in practice, PF_KEY has been extended in a compatible way, allowing older key management daemons to continue to work; actual flag days have been very rare.

Glossary

This glossary covers a mixture of general networking, general security, and opensolaris-specific terms. Definitions found here may not be precisely correct outside the specific intersection of contexts involved in this project.

`cred_t`

Credentials structure. Kernel data structure containing a set of security-related process attributes identifying a subject responsible for some process or action

`dblk_t`

Message data block. STREAMS data structure storing a chunk of a message

Admin_High

Highest possible sensitivity label, more sensitive than any user data. Generally reserved for system use only

Admin_Low

Lowest possible sensitivity label, less sensitive than any user data. Generally reserved for system use only

AH

Authentication Header. An IP protocol which provides integrity protection and data origin authentication of packet contents and some packet headers. Part of IPsec

CIPSO

Commercial IP Security Option. A specification for the explicit labeling of IP packets through the use of IP header options.

Committed

An interface stability level indicating that a Public interface will (generally) change incompatibly only in a Major release. See `attributes(5)` for more information

DOI

Domain of Interpretation; an identifier used to distinguish between different sets of labels

DPD

Data protection domain. PF_KEY term for a DOI

ESP

Encapsulating Security Payload. An IP protocol which provides confidentiality, integrity protection, and data origin authentication of packet contents. Part of IPsec

Flag Day

An incompatible change to an interface requiring the simultaneous update of both the implementation and the users of an interface

global zone

TBD

IKE

Internet Key Exchange, a key management protocol for IPsec

IP

The Internet Protocol

IPsec

the standard network-layer security protocol for IP

Label

In a system providing Mandatory Access Control, security-related metadata attached to some or all data

MAC

Mandatory Access Control. *Anyone got a nice concise definition?*

multi-label

A peer with which we may communicate data with multiple labels. A multi-label peer must be aware of labels

non-global zone

TBD

PF_KEY

A sockets-based API for manipulating an IPsec SADB, defined by RFC2367

privilege

A solaris process attribute which, if granted to a process, allows that process to perform certain operations. See `privileges(5)` for more information.

SA

IPsec Security Association. Associates IPsec keying material with a set of selectors and other attributes. Created and managed manually or by a key management protocol such as IKE.

SADB

IPsec Security Association Database. A node's SADB contains the set of currently active IPsec security associations terminating at that node

Sensitivity Label

A label identifying or describing the sensitivity of some piece of data, typically including a classification level and compartment information

single-label

A peer with which we may only communicate data with a single label. A single-label peer may be completely unaware of labels, or may be label-aware but only trusted to communicate data of a single label

SIT_SECRECY

An IKE *situation* which allows for the negotiation of sensitivity labels

situation

An attribute of an IKE exchange which provides information that can be used by the responder to make a policy determination about how to process an incoming Security Association request. Most IKE implementations only support the SIT_IDENTITY_ONLY situation.

SPD

IPsec Security Policy Database. A set of rules describing what sort of IPsec protection is required on traffic entering or leaving a node.

SPI

Security Parameter Index. In IPsec and IKE, an identifier used on the wire to indicate a specific Security Association

tn*db

The trusted networking databases as a whole, including the tnrhdb and the tnrhtp

tnrhdb

Trusted networking remote host database; associates a template with peer ip addresses

tnrhtp

Trusted networking remote host template database; defines various Trusted networking properties of a network entity, including clearances, default labels, etc.

TX

Trusted Extensions; a set of features and extensions to Solaris which (among other things) provide for mandatory access control with sensitivity labels

Volatile

An interface stability level indicating that a Public interface may change incompatibly at nearly any time. See `attributes(5)` for more information

zone

TBD