Aalto University

Network Security: IPsec architecture

Tuomas Aura

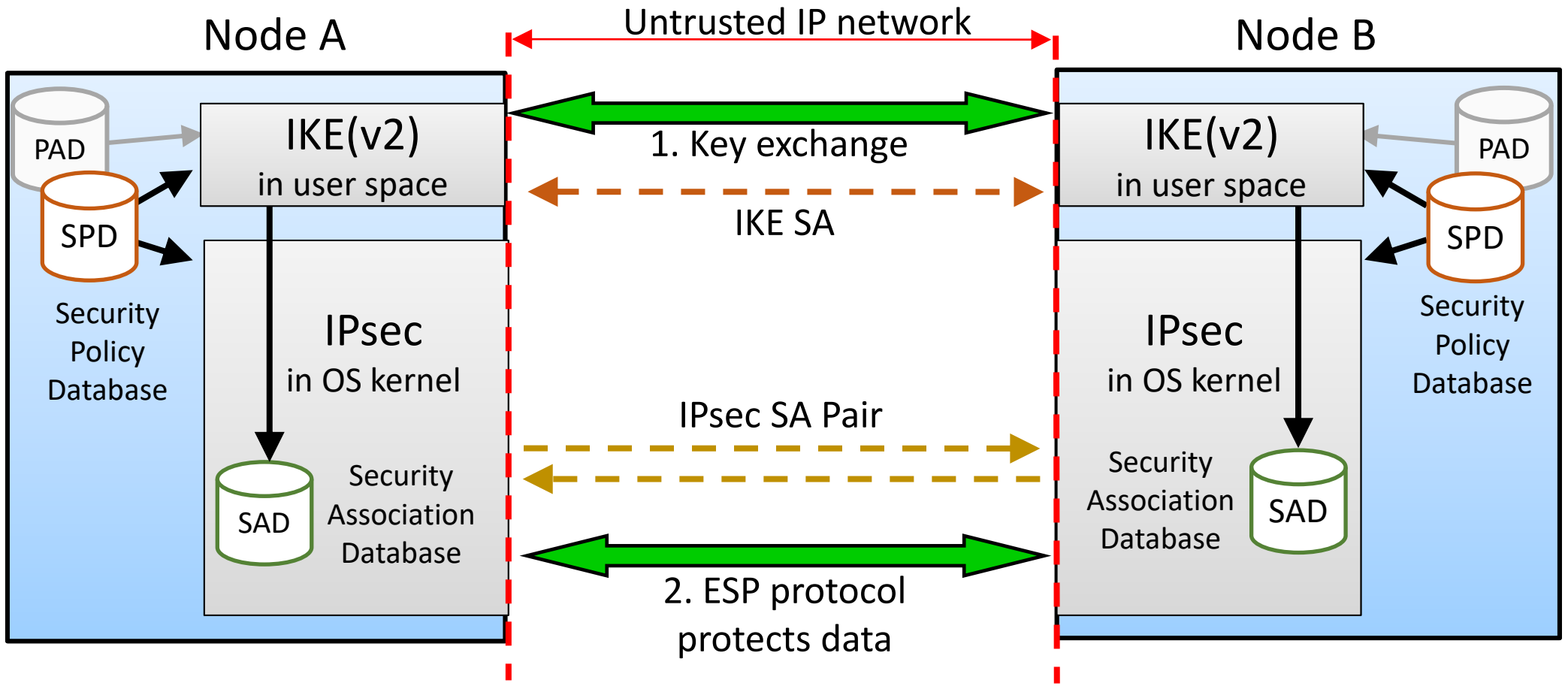
CS-E4300 Network security

Aalto University

Internet protocol security (IPsec)

- Network-layer security protocol
  - Protects IP packets between two hosts or gateways
  - Transparent to transport layer and applications: security policy is defined and enforced on network level
  - IP addresses are used as host identifiers
- Two steps:
  1. IKE authenticated key exchange creates security associations
  2. ESP session protocol protects data
- Specified by Internet Engineering Task Force (IETF)
  - Initially designed as part of for IPv6

IPsec architecture [RFC 4301]



- Security associations (SA) in SAD created by IKE, used by IPsec ESP
- Security policy in SPD guides SA creation and use

Security Associations (SA)

- One IKE SA for each pair of nodes
  - Stores a master session key for creating IPsec SAs
- At least one IPsec SA pair for each pair of nodes
  - Stores negotiated algorithms, keys, and algorithm state
  - IPsec SAs always come in pairs, one in each direction
- IPsec SAs identified by a 32-bit security parameter index (SPI)
  - The destination node selects the SPI value
- Node stores IPsec SAs in its security association database (SAD)

IPsec databases

- Security association database (SAD)
  - Contains the IPsec SAs i.e.t the dynamic protection state
- Security policy database (SPD)
  - Contains the static security policy
  - Set by system admin (e.g. Windows group policy) or VPN application
- Peer authorization database (PAD)
  - Mapping between authenticated names and IP addresses
  - Conceptual; not implemented as an actual database
- The IKE service/daemon stores IKE SAs
  - Master secret for creating IPsec SAs; hash of DH secret and nonces

Note: our description of SPD differs from RFC 4301 but is closer to most implementations.

Gateway SPD/SAD example

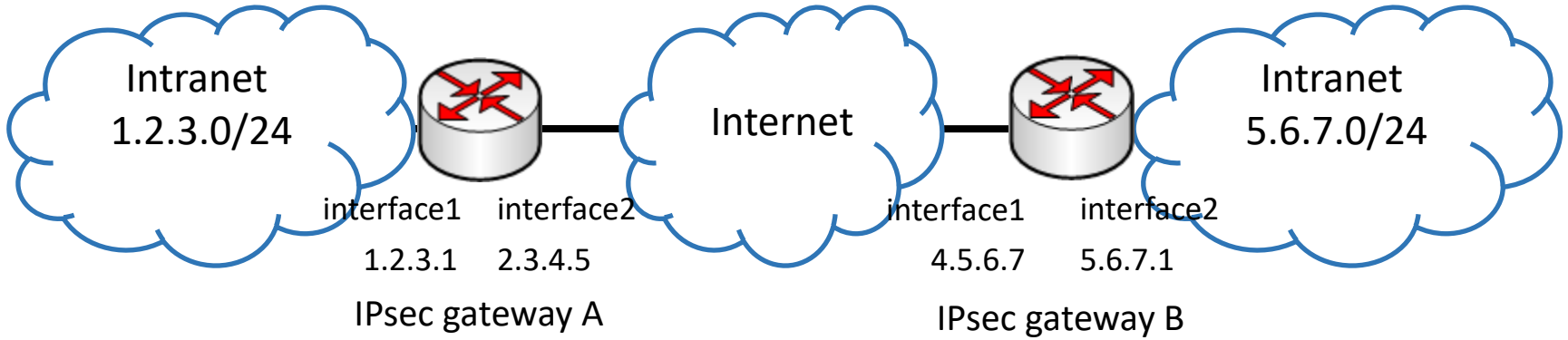
SPD of gateway A, interface 2

Protocol	Local IP	Port	Remote IP	Port	Action	Comment
UDP	2.3.4.5	500	4.5.6.7	500	BYPASS	IKE
*	1.2.3.0/24	*	5.6.7.0/24	*	ESP tunnel to 4.5.6.7	Protect VPN traffic
*	*	*	*	*	BYPASS	All other peers

Pointers to created associations

SAD of gateway A

SPI	SPD selector values	Protocol	Algorithms, keys, algorithm state
spi1	TCP, 1.2.3.0/24, 5.6.7.0/24	ESP tunnel from 4.5.6.7	…
spi2	—	ESP tunnel to 4.5.6.7	…



Security policy database (SPD)

- Specifies the static security policy
- Policy maps inbound and outbound packets to actions
  - SPD = linearly ordered list of policies
  - Policy = selectors + action
  - The first policy with matching selectors applies to each packet
- Policy selector is a 5-tuple:
  - Local and remote IP addresses and ports, transport protocol (TCP, UDP, ICMP)
- Actions: BYPASS (allow), DISCARD (block), or PROTECT
  - PROTECT specifies also the session protocol and algorithms
  - Packet is mapped to a suitable IPsec security association (SA)
  - If the SA does not exist, IKE is triggered to create them
- Multi-homed nodes have a separate SPD for each network interface (can be implemented as one database)
- Policies at peer nodes must match if they are to communicate

Security association database (SAD)

- Contains the dynamic encryption and authentication state
- IPsec SAs always come in pairs: inbound and outbound
- SAD is indexed by SPI (for unicast packets)
- SAs are typically created by IKE but they may also be configured manually or by other software,
  e.g. to create a VPN tunnel
- Each SAD entry remembers also the policy selector values that were used when creating it

Host SPD example

Host-to-host IPsec is problematic, as explained in a later lecture

SPD for host 1.2.3.101 in intranet 1.2.3.0/24,
connecting to server 1.2.4.10 in network 1.2.4.0/24 (DMZ) and to the Internet

Protocol	Local IP	Port	Remote IP	Port	Action	Comment
UDP	1.2.3.101	500	*	500	BYPASS	IKE
ICMP	1.2.3.101	*	*	*	BYPASS	Error messages
*	1.2.3.101	*	1.2.3.0/24	*	PROTECT: ESP in transport-mode	Encrypt intranet traffic
TCP	1.2.3.101	*	1.2.4.10	80	PROTECT: ESP in transport-mode	Encrypt to server in DMZ
TCP	1.2.3.101	≥1024	1.2.4.10	443	BYPASS	Allow TLS to server in DMZ
*	1.2.3.101	*	1.2.4.0/24	*	DISCARD	Others in DMZ
*	1.2.3.101	*	*	*	BYPASS	Internet

- Note that the other endpoint (other intranet hosts and 1.2.4.10) must have an IPsec policy that specifies the same protection for the same packets
- What is the danger in bypassing TLS traffic (line 5) and ICMP (line 2)?
- What if the attacker can poison DNS?

IPsec policy implementation differences

- Firewall and IPsec policies can be unified into one policy:
  - Which incoming/outgoing packets to drop or log, and which require authentication and encryption?
- IPsec policy may be specified in terms of
  - local and remote addresses
  - left and right, so that the same policy file works at both ends, or
  - source and destination addressed, with mirror flag

Mirror	Protocol	Source IP	Port	Destination IP	Port	Action	Comment
yes	UDP	2.3.4.5	500	4.5.6.7	500	BYPASS	IKE
yes	*	1.2.3.0/24	*	5.6.7.0/24	*	ESP tunnel to 4.5.6.7	Protect VPN traffic
yes	*	*	*	*	*	BYPASS	All other peers

Outbound packet processing

Additional reading

- Processing outbound packets in IPsec:
  1. For each outbound packet, IPsec finds the first matching policy in the security policy database (SDP)
  2. If the policy requires protection, IPsec maps the packet to the right security association (SA) in the SA database (SAD)
  3. If no such SA exists, IPsec invokes the IKE service (running in user space) to create a new IPsec SA pair
  4. While waiting for the IPsec SA, at most one outbound packet (often TCP SYN) is buffered in the kernel
  5. When the SA has been created, the buffered packet is encrypted and a MAC added and sent to the network

Inbound packet processing

Additional reading

- Processing inbound IPsec packets:
  1. IPsec looks up the inbound SA in SAD based on the SPI in the packet
  2. IPsec processes the packet with the SA, i.e. verifies the MAC and decrypts it
  3. IPsec compares the packet with the selector values that were used when creating the SA entry. For tunnel-mode packets, the comparison is done with the inner IP header. This is done to check that the protection of inbound packets matches the policy.
- Processing of inbound non-IPsec packets:
  - IPsec finds the first matching policy in SPD and checks that the action is BYPASS
  - If the action is not BYPASS, the packet is dropped

Exercises

Additional reading

- Why is IPsec used mainly for VPN implementations? Does IPsec VPN suffer from any of the problems mentioned in the lectures, or can they be avoided?
- For the IPsec policy examples of this lecture, define the IPsec policy for the peer nodes i.e. the other ends of the connections.
- Try to configure the IPsec policy between two computers. What difficulties did you meet? Use ping and telnet to test connectivity. How can you be sure that the packets on the wire are encrypted?
- What administrative problems arise from the fact that IPsec security policies in two communicating nodes must match? How is this solved in Windows?
- RFC 4301 requires that the SPD is *decorrelated*, i.e. that the selectors of policy entries not to overlap, i.e. that any IP packet will match at most one rule (excluding the default rule which matches all packet). Yet, the policies created by system administrators almost always have overlapping entries. Device an algorithm for transforming any IPsec policy to an equivalent decorrelated policy. (Real protocol stacks do not implement decorrelation. Why?)
- Each SAD entry stores (caches) policy selector values from the policy that was used when creating it. Inbound packets are compared against these selectors to check that the packet arrives on the correct SA.
  - What security problem would arise without this check?
  - What security weakness does the caching have compared to a lookup through the full SPD?
  - Would policy decorrelation simplify the situation?