IPSec: IKE

- **General**
- IKE creates SA, refreshes them and deletes them.
- IKE has the following exchanges:
  - Phase one (creation of IKE SA): There are two modes for phase one: main mode or aggressive mode
  - Phase two (creation of IPSec SA): there is only one mode: quick mode
  - Maintenance of IKE SA
  - Negotiation of private Diffie-Hellman groups
- What the last exchange means is that in the phase one there are predefined several ways to use Diffie-Hellman, but one can define own ways also using the last exchange.
- IKE protocol initial message exchanges are not encrypted.
- IKE uses (normally) the UDP port 500.

IPSec: IKE

- **The predefined Diffie-Hellman groups in IKE:**
- (group here means only an agreement of the algorithm)
- 1. MODP group with a 768-bit modulus
- 2. MODP group with a 1024-bit modulus
- 3. ECP group with a 155-bit modulus
- 4. EC2N group with a 185-bit modulus
- 5. MODP group with a 1680-bit modulus
- What this means is that you can use discrete logarithm problem (see Diffie-Hellman algorithm from a previous lecture) noted as MODP and the number p for $A=g^a \bmod p$ must have the defined length. The algorithm family EC2N is a family of elliptic curve cryptoalgorithms. They give good security level with shorter keys and less processing. ECP 155 is about as secure MODP 768, respectively EC2N 185 about as good as MODP 1024.

IPSec: IKE

- In the first part of the IKE exchange, an authentication method is agreed.
- There are five authentication methods
- 1) preshared keys
- 2) digital signature with DSA
- 3) digital signature with RSA
- 4) authentication via exchange of encrypted nonces
- 5) revised method 4)
- This method is agreed via exchange of IKE SA.
- Exchange of IKE SA contains also some secret information.
- The peers generate four secrets: SKEYID, SKEYID_d, SKEYID_a and SKEYID_e. Both sides take part in creating the secrets.

IPSec: IKE

- **Generation of the secrets:**
- Each side contributes a cookie (CKY-x) and a nonce (Nx) to SKEYID generation (x=i (initiator) or r (responder).
- A nonce is simply a pseudo-random number, a cookie is generated by taking a hash from some data (we return to this).
- For preshader key authentication
  - SKEYID=PRF(preshared key, Ni|Nr)
- For signature authentication Diffie-Hellman type g^{xy} is used:
  - SKEYID=PRF(Ni|Nr, g^{xy})
- For encrypted nonce authentication:
  - SKEYID=PRF(hash(Ni|Nr), CKY-i|CKY-r)
- Here | denotes concatenating the data, so Ni|Nr = nonce from initiator + nonce from responder. PRF is a result of a hash function, usually HMAC.

IPSec: IKE

- **All other secrets are derived from SKEYID:**
- SKEYID_d=PRF(SKEYID, g^{xy}|CKY-i|CKY-r|0)
- SKEYID_a=PRF(SKEYID, SKEYID_d|g^{xy}|CKY-i|CKY-r|1)
- SKEYID_e=PRF(SKEYID, SKEYID_a|g^{xy}|CKY-i|CKY-r|2)
- **Why all these secrets?**
- SKEYID_d is used for deriving keying data for IPSec
- SKEYID_a is used for integrity and data source authentication
- SKEYID_e is used to encrypt IKE messages.
- Different keys must be used for security purposes. Because of the hash function PRF, the original secret SKEYID cannot be calculated from the derived secrets.
- Why so many options in IKE (remember, many options were one reason OSI failed to gain popularity) ?

IPSec: IKE

- **cookie exchange**
- IKE uses the following cookie generation method:
- a cookie is the result of hashing a unique identifier of the peer (peer’s IP address, port and protocol), a secret known only to the generator of the cookie, and a time stamp.
- The initiator generates a cookie, sets the responder cookie to zero and sends to the responder.
- The responder generates a responder cookie, copies the initiator cookie to the message and sends it to the initiator.
- The initiator can easily check that the initiator cookie is to one it generated and that the peer’s addresses match.
- Only if the cookie matches, check of signatures etc. are made. I am not sure, but spoofing the addresses should pass the cookie check and force the sides to continue to computationally expensive Diffie-Hellman, so this does not protect against DoS?

IPSec: IKE

- **Phase one, normal mode**
- using preshared key authentication
- Initiator   Responder

$$\begin{tikzcd}
\text{Header, SA} \arrow[r, shift left] & \text{Header, SA} \arrow[l, shift left] \\
\text{Header, KE, Nonce} \arrow[r, shift left] & \text{Header, KE, Nonce} \arrow[l, shift left] \\
\text{Header, IDi, Hash} \arrow[r, shift left] & \text{Header, IDi, Hash} \arrow[l, shift left]
\end{tikzcd}$$

The normal mode has an exchange of six messages, several versions of the phase one normal mode exist. SA=Security Association, KE=Key Exchange, Nonce=random number, IDi= identity of the peer.

IPSec: IKE

- **Phase one of normal mode**
- using public key exchanges:
- <u>Initiator   Responder</u>

Initiator		Responder
Header, SA	$\rightleftarrows$	Header, SA
Header, KE, Ni [,Cert_Req]	$\rightleftarrows$	Header, KE, Ni [,Cert_Req]
Header, IDi, [Cert,] Signature	$\rightleftarrows$	Header, IDi, [Cert,] Signature

In this variant optional payloads are bracketed. In the optional features a certificate can be requested (Cert_Req) and then it is returned in Cert. Ni=Nonce i.

IPSec: IKE

- **Phase one of normal mode**
- the standard method using public key exchanges:
- <u>Initiator   Responder</u>

Header, SA	$\rightleftarrows$	Header, SA
Header, KE, {IDi}pub_r, {Ni}pub_r	$\rightleftarrows$	Header, KE, {IDi}pub_i, {Ni}pub_r
Header, Hash	$\rightleftarrows$	Header, Hash

In this variant {something}pub_x means something encrypted with the public key of x=i (initiator) or r (responder). Ni is nonce.

IPSec: IKE

- **Policy negotiation**
- After IKE SA is agreed, IKE will negotiate of the policy.
- Policy is something like: authenticate everything and if possible encrypt it, and if possible also compress it.
- For each operation there may be several algorithms.
- SA payload can contain several proposals for protocols and exact algorithms (transforms).
- Policy negotiation works so that the initiator proposes some algorithms and the responder removes from the list what it does not want to use.
- Negotiating compression is also included in IKE since it is not good to try to compress encrypted data (it will not compress, it is random), therefore link layer compression like in PPP will not work with IPSec. One (but not efficient) way is to compress each IP packet on IPSec layer before encryptation with PCP.

IPSec: IKE

Example proposal for SA: the offer maker proposes in the given order the following choices.

Proposal 1: AH

- Transform 1: HMAC-SHA
- Transform 2: HMAC-MD5

Proposal 2: ESP

- Transform 1: 3DES with HMAC-SHA
- Transform 2: 3DES with HMAC-MD5
- Transform 3: DES with HMAC-SHA
- Transform 4: DES with HMAC-MD5

Proposal 3: PCP (compression before encryptation)

- Transform 1: LZS (one header compression algorithm)
- Transform 2: Deflate (another header compression)

IPSec: IKE

- **Phase one: aggressive mode**
- Aggressive mode is more simple than the normal mode. In the aggressive mode there are only three messages exchanged.
- The initiator offers a list of protection suites, his Diffie-Hellman public key value, his nonce and his identity.
- The responder replies with a selected protection suite, his Diffie-Hellman public value, his nonce, his identity, and authentication payload, like a signature.
- The initiator responds with authentication payload.
- There is no chance to negotiate as much in this case as in the normal mode.
- The method suits well for connecting to own site from a remote site as then it is known in advance what kind of authentication the other side supports.

IPSec: IKE

- **Phase two: quick mode**
- Phase two of IKE creates IPSec SA. Since IKE can be used for other protocols than IPSec, like the routing protocols RIPv2 and OSPF, IKE SA is not directly IPSec SA**.**
- IKE SA protects the quick mode by encrypting messages and authenticating them. Authentication comes from use of PRF (the HMAC hash function)
- The quick mode creates keys for IPSec association.
- Many quick modes can be made using the same IKE SA, therefore a message ID (M-ID) is used to identify the IPSec SA. Nonces are added to prevent replay of the same messages by an attacker.
- The quick mode has more details, but the following figure gives the general view of the protocol.

IPSec: IKE

- **Quick mode exchange**
- Initiator Responder

Header, HASH1, SA, Ni
[, KE][, IDci, IDcr] $\longrightarrow$

$\longleftarrow$ Header, HASH2, SA, Nr
[, KE] [, IDci, IDcr]

Header, HASH3 $\longrightarrow$

HASH1=PRF(SKEYID_a, M-ID | SA | Ni [| KE] [| IDci | IDcr])

HASH2=PRF(SKEYID_a, M-ID | Ni | SA [| KE] [| IDci | IDcr])

HASH3=PRF(SKEYID_a, 0 | M-ID | Ni | Nr)

IPSec: Policy

- IPSec does not define policy, however there are guidelines.
- Policy is kept in a policy server in the policy database.
- Policy data is formatted as LDAP (Lightweight Directory Access Protocol) entries.
- There is a LDAP interface to a policy server so that policies can be read, searched, modified, added and deleted.
- There is also interface from IKE to policy database. It should be possible for IKE to search for policies, for instance get the policy for a particular <dst, src> pair.
- Policy is kept as a set of rules. Always the search gives the longest prefix matching a rule, like if you have a policy for the address 205.15.2.1 and another policy for addresses 205.15/16, the search for a policy for 205.15.2.1 will give the first policy.

IPSec: Policy

- The security policy database SPD defines the policy.
- The SA database SADB defines the IPSec protocol (AH, ESP), Security Parameter Index (SPI) authentication and encryptation keys and other parameters which are not shown (sequence number, lifetime, etc.)
- There are specific Security Associations for inbound and outbound traffic since IPSec security associations are one way.
- In the following example there are two hosts Host A (address 1.1.1.1) and Host B (address 2.2.2.2).
- The hosts have established two security associations SA1 and SA2.
- In SA1 ESP with 3DES keys is used, in SA2 ESP with DES is used for enryptation. SAs for source authentication are left out for clarity.

IPSec: Policy

- **Example policy (SPD connection to SADB)**

SDP (Security Policy Database)

From	To	Protocol	Port	Policy
1.1.1.1	2.2.2.2	TCP	1000	ESP with 3DES
1.1.1.1	2.2.2.2	*	*	ESP with DES

Inbound SADP (Security Association Database)

From	To	Protocol	SPI	SA RECORD
2.2.2.2	1.1.1.1	ESP	1	64 bit DES Key SA2
2.2.2.2	1.1.1.1	ESP	11	168 bit 3DES SA1Key

IPSec: Usage

- **End-to-end security**
- IPSec transport mode suits well for end-to-end security.
- **VPN (Virtual Private Network)**
- VPN is probably the main usage of IPSec. IPSec tunneling mode suits for this usage. VPNs are a necessity for e.g. enabling medical data to be transported in the Internet - by law the data must be protected. The poor standardization situation has prevented interworking of existing VPN solutions.
- **Road Warrior**
- The term refers to a travelling salesman trying to connect home from a hotel. He cannot be in a VPN, but he could use IPSec transport mode to connect to the VPN.

IPSec: Usage

- **Nested tunnels**
- It is possible to tunnel IPSec tunnels through other tunnels. A typical usage of such configuration is a user passing to an intranet with one tunnel and continuing with another tunnel to the home LAN. The main disadvantage is the growing overhead by IP and IPSec headers.

IP	ESP	IP	ESP	IP	TCP	Data

The other possibility is chaining IPSec tunnels. Then an IPSec channel ends to a network element (Hub, router) and another IPSec channels starts from it.

IPSec: **Problems in IPSec**

- **NAT (Network Address Translation)**
- IPSec is a point to point protocol, which makes it impossible for any network element to update fields in IP-packets.
- In the AH protocol some specific changeable fields in the outer IP header (Time to live, Type or service, flags, header checksum) can change. These fields are updated by routers and they are set to zero before calculating authentication data.
- In EPS the whole inner IP header is protected and maybe also encrypted, no field can be updated.
- Especially, IP addresses cannot be changed in either protocol, but this is what NAT does. NAT looks like address spoofing to IPSec, but actually is is a security mechanism.
- There is work done to enable both IPSec and NAT in the future.

IPSec: **Problems in IPSec**

- **Compression**
- Encrypted data does not compress. Compression will typically do nothing but it can increase data size.
- Many protocols apply compression in the link layer as the link connection is specific to the particular network (like wireless or modem connection). This link layer compression should be turned off (but you may not be able to do so on e.g. wireless link protocol).
- Using compression on upper layers (transport, session, application) would give good results.
- IPSec has an option for using compression on the IP level before encrypting data. It is PCP and has two compression algorithms Deflate and LZS.
- PCP is stateless, therefore it is compressing each packet independently without using earlier packets.

IPSec: **Problems in IPSec**

- Naturally, using data from more packets is the way effective compression works, so PCP is not effective.
- Sometimes PCP compresses, then PCP header and compressed data is put on place of the original data. If PCP does not compress, the original data is kept in the packet.
- **QoS mechanisms, Firewalls etc.**
- Any mechanism which tries to read fields from the inner IP header or payload data will have problems with ESP. Fortunately many QoS mechanisms can be still supported as they work with the outer IP header.
- Mobile IP is problematic in the way that the home agent must be the end point of encryptation as it will redirect the packet to another address. Then the home agent should decrypt and encrypt, i.e., be a trusted gateway. This means performance penalty and requires that the home agent is trusted.

IPSec: **Problems in IPSec**

- **Management of protected channels**
- Nested tunnels will create for a large network a very complicated structure. It is common to build networks instead by concatenating IPSec tunnels protected by different keys. Then the network elements decrypting and encrypting data are potential attacking sites and traffic will experience considerable delays.
- Link layer protection for each link using stream ciphers (the traditional solution) may be superior.
- Consider also the overhead of IPSec tunnels for some applications, like Voice over IP already has enough overhead and cannot necessarily tolerate much more. Still, voice is a future type of data which needs privacy.
- Increasing the IP packet size may lead to need for fragmentation.
- But, IPSec does not fragment and do not fragment bit is set. What to do?

IPSec: **Problems in IPSec**

- Let us say, the packet comes to a router connecting to another network where the message transfer unit size is too small and the packet should be fragmented.
- As the router cannot fragment because it is forbidden, it sends ICMP packet to the originator instructing to use a smaller payload. The router may have to keep the packet for some time before it learns who is the originator.
- In some cases decreasing packet size would cause excessive overhead, so fragmentation is used.
- **Multicast**
- IPSec does not suit well to multicast as it is point to point. IKE does not suit to multicast. Multicast is one of the few things IP does better than other protocols, what to do?
- Especially antireplay and source authentication are problems.

IPSec: **Problems in IPSec**

- **Multicast**
- While whole multicast problematic is not solved, there are proposals for some parts of it.
- One is source authentication in multicast. If a firm uses multicast to deliver information to selected receivers, how do we prevent one of the receivers from forging false information and claiming it is from the original sender?
- This could be done by public key cryptography (signatures), but such a solution is too slow.
- One way is to make a hash from the next packet, crypt that with a symmetric key and add it to the packet. The problem may be that finally the encrypted value will be something for which another data hashes correctly. Another way is to extend the authentication digest method of AH to contain many digests.

IPSec: **Problems in IPSec**

- **Multicast**
- Key distribution in multicast is a particular problem. There are several ways, like manual distribution of keys, hierarchical key tree, sharing a secret with many sources and so on.
- The problem of multikey management becomes difficult if it should be possible to exclude some party from a group and regeneration of new keys should be easy.
- MKMP (MultiKey Management Protocol) is yet another proposal by the authors of the IPSec book.
- Antireplay in multicast is a bit problematic as the IPSec method to prevent antireplay is by increasing sequence numbers. If there are many parties, the sequence numbers will not be synchronized easily, in fact that solution cannot be extended.

IPSec: **Problems in IPSec**

- **Summary**
- The traditional wisdom is to use security on two levels: in the link layer between point-to-point links with fast stream ciphers, and for protecting sensitive data, still on the application layer as end-to-end encryptation.
- Putting security mechanisms on the network layer has not been popular as the network functions are problematic for security mechanisms, IPSec tries to do it and is at least partially successful. However, so far IPSec has still some unsolved issues.
- **L2TP**
- Link layer tunneling protocol (L2TP) is a proposal of running link layer protocol on top of IPSec.
- Then any link layer protocol , such as PPP can be run over secured channels, at the cost of quite much overhead.

IPSec: PKI

- **Public Key Infrastructure (PKI)**
- Exchanging public keys can be done with IKE and then each party creates a mutual trust relationship.
- Such a solution scales poorly and public keys should be publicly known.
- The usual solution for this is Public Key Infrastructure where the public keys are in certificates.
- Usually the certificates have the structure of ITU-T X.509 certificates, while some variations also exist.
- Certificates are issued by Certification Authority (CA). If the parties of communication all trust the same CA there is no problem, but in a large network there will be many CAs. Then there should be cross certificates by which certificates giving the public keys of the CAs can be verified.

IPSec: PKI

- **Public Key Infrastructure (PKI)**
- As known, this finally leads to a tree structure where a public key of a party is checked by checking all CA certificates in a certification path.
- The structure of such a tree is described in the X.509 standard and has not been essentially improved by newer variants of the public key distribution idea.
- For the Internet the X.500-directory is usually replaced by an LDAP directory, but the functionality is similar.
- There are some problems with this kind of structure, like the revocation list - compromised certificates should be put to a revocation list in the certificate directory tree. The problem is that how the revocation lists are protected - it seems to require a trusted directory, which is not desirable.

IPSec: Future work

- At the moment there is no PKI defined for IPSec, but this is clearly a scalability problem, something like that is needed in the future.
- **Key recovery**
- This is a controversial issue. Police forces in many countries would like to be able to obtain keys to decrypt encrypted data transmissions, provided that some court approves it.
- Protectors of civil rights are against this. IEFT has issued an RFC with the number 1984 (Orwell’s year picked up intentionally as the RFC number) stating being against this.
- For IPSec recovery of keys is not natural for the protocol as they do not use long lasting secrets. However, key recovery can be inserted to the Quick Mode in a way which enables but does not mandate key recovery.