IKE

Internet Key Exchange:

Before IPSec sends authenticated or encrypted IP data, both the sender and receiver must agree on the protocols, encryption algorithms and keys to use for message integrity, authentication and encryption. IKE is used to negotiate these and provides primary authentication.

Key lifetimes can be set and re-keying can be done automatically

IKE

Internet Key Exchange:

Protocol for doing mutual authentication and establishing a shared secret key to create an IPSec SA.

Uses:

long term keys (public signature-only keys, pre-shared secret keys, public encryption keys)

Pieces:

ISAKMP (Internet Security Association and Key Management Protocol) framework (OAKLEY implementation)

IKE (Internet Key Exchange) defines fields, chooses options of ISAKMP

DOI (Domain of Interpretation) specifies particular use of ISAKMP

IKE

ISAKMP:

Framework developed by the NSA – mainly concerned with the details of Security Association management

Consists of procedures and fields for

- Authentication of peers
- Negotiation, modification, deletion of Security Associations
- key generation techniques
- threat mitigation (e.g. DoS, replay attacks)

Is distinct from key exchange protocols so details of managing security associations are separated from details of managing key exchange

IKE

ISAKMP:

Framework developed by the NSA – mainly concerned with the details of Security Association management

Consists of procedures and fields for

- Authentication of peers
- Negotiation, modification, deletion of Security Associations
- key generation techniques
- threat mitigation (e.g. DoS, replay attacks)

Is distinct from key exchange protocols so details of managing security associations are separated from details of managing key exchange

An implementation requires a key exchange protocol like IKE

IKE

ISAKMP:

Framework developed by the NSA – mainly concerned with the details of Security Association management

Consists of procedures and fields for
- Authentication of peers
- Negotiation, modification, deletion of Security Associations
- key generation techniques
- threat mitigation (e.g. DoS, replay attacks)

Is distinct from key exchange protocols so details of managing security associations are separated from details of managing key exchange

An implementation requires a key exchange protocol like IKE

Common implementation is OAKLEY, a key-agreement protocol using DH. Basis of IKE.

IKE

IKE: Internet Key Exchange (IKE or IKEv2)

The protocol used to set up a security association (SA) in IPsec.

Uses Diffie-Hellman to get a shared session secret
That secret is used to derive up to 6 cryptographic keys.

IKE

IKE: Internet Key Exchange (IKE or IKEv2)

The protocol used to set up a security association (SA) in IPsec.

Uses Diffie-Hellman to get a shared session secret
That secret is used to derive up to 6 cryptographic keys.

Public key algorithms or a pre-shared key are used to mutually authenticate communicating parties.

IKE

IKE: Internet Key Exchange (IKE or IKEv2)

The protocol used to set up a security association (SA) in IPsec.

Uses Diffie-Hellman to get a shared session secret
That secret is used to derive up to 6 cryptographic keys.

Public key algorithms or a pre-shared key are used to mutually authenticate communicating parties.

IKE builds upon the Oakley protocol.

IKE

IKE: Internet Key Exchange (IKE or IKEv2)

The protocol used to set up a security association (SA) in IPsec.

Uses Diffie-Hellman to get a shared session secret
That secret is used to derive up to 6 cryptographic keys.

Public key algorithms or a pre-shared key are used to mutually authenticate communicating parties.

IKE builds upon the Oakley protocol.

Implementation: a daemon in user space (access to databases)
packets parsed by kernel modules (for speed)

IKE

IKE: Internet Key Exchange (IKE or IKEv2)

The protocol used to set up a security association (SA) in IPsec.

Uses Diffie-Hellman to get a shared session secret
That secret is used to derive up to 6 cryptographic keys.

Public key algorithms or a pre-shared key are used to mutually authenticate communicating parties.

IKE builds upon the Oakley protocol.

Implementation: a daemon in user space (access to databases)
packets parsed by kernel modules (for speed)

IKEv2 solved many IKE problems: DoS, poor SA negotiation, not completely specified.

IKE

DOI: Domain of Interpretation

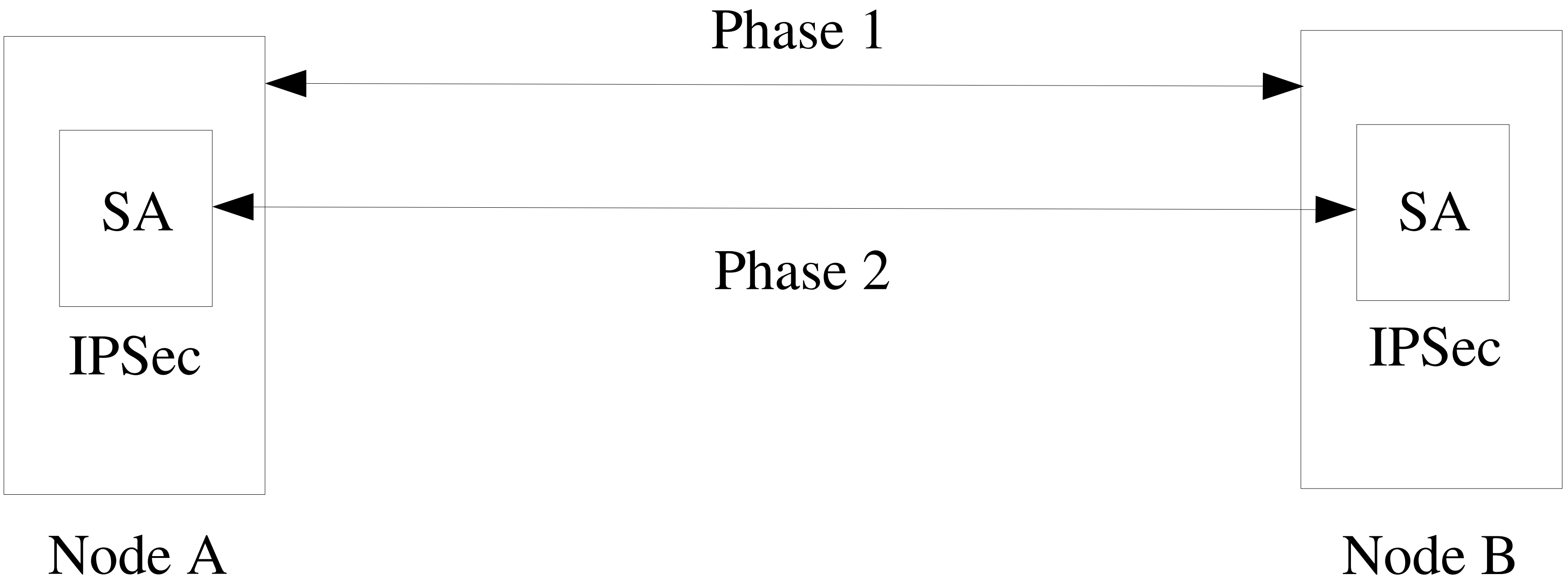
A 32-bit value which identifies the context in which the Security Association payload is to be evaluated.

A DOI identifier is used to interpret ISAKMP payloads

A DOI specification is publicly available, e.g. as a RFC and includes the following, among others:

- naming scheme for DOI-specific protocol identifiers
- interpretation for the Situation field
  SA from assoc ID packet, needs secrecy, needs integrity check
- set of applicable security policies
- syntax for DOI-specific SA Attributes
- syntax for DOI-specific payload contents
- additional Key Exchange types, if needed
- additional Notification Message types, if needed

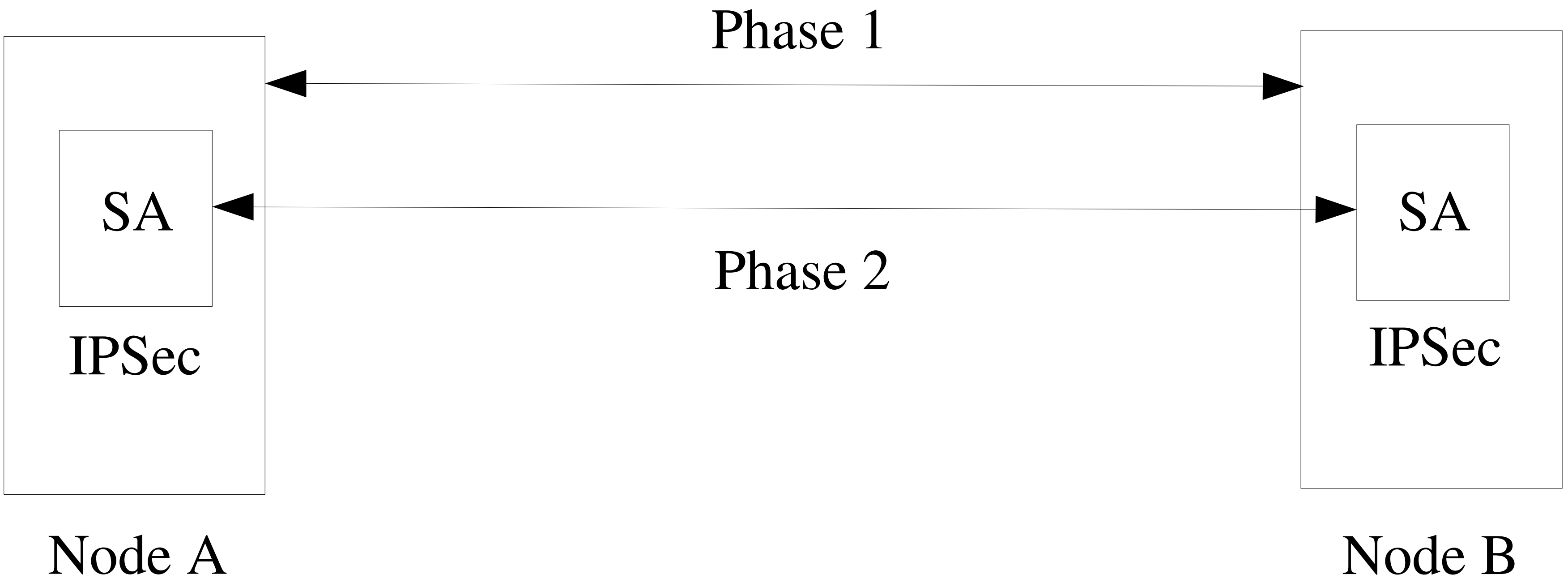
IKE



Phase 1: Does mutual authentication and establishes session keys based on identities such as names, and secrets

Phase 2: SAs are established between two entities

IKE



Phase 1: Does mutual authentication and establishes session keys based on identities such as names, and secrets

Phase 2: SAs are established between two entities

Reason: different SAs may be established for different traffic flows; phase 1 need be done once, phase 2 uses the same phase 1 session key to generate multiple SAs.

IKE

Cookies: used to prevent DoS. Both sides have a cookie which is a hash over the IP source and destination addresses, the source and destination ports, and a locally generated secret value. Cookies are sent in the opening transaction. If cookie is not received in the second round of messages, connection is cancelled.

IKE

Cookies: used to prevent DoS. Both sides have a cookie which is a hash over the IP source and destination addresses, the source and destination ports, and a locally generated secret value. Cookies are sent in the opening transaction. If cookie is not received in the second round of messages, connection is cancelled.

But ISAKMP requires that the cookie is unique for every SA so SA information needs to be maintained during handshake So the cookies are not actually stateless

IKE

Cookies: used to prevent DoS. Both sides have a cookie which is a hash over the IP source and destination addresses, the source and destination ports, and a locally generated secret value. Cookies are sent in the opening transaction. If cookie is not received in the second round of messages, connection is cancelled.

But ISAKMP requires that the cookie is unique for every SA so SA information needs to be maintained during handshake So the cookies are not actually stateless

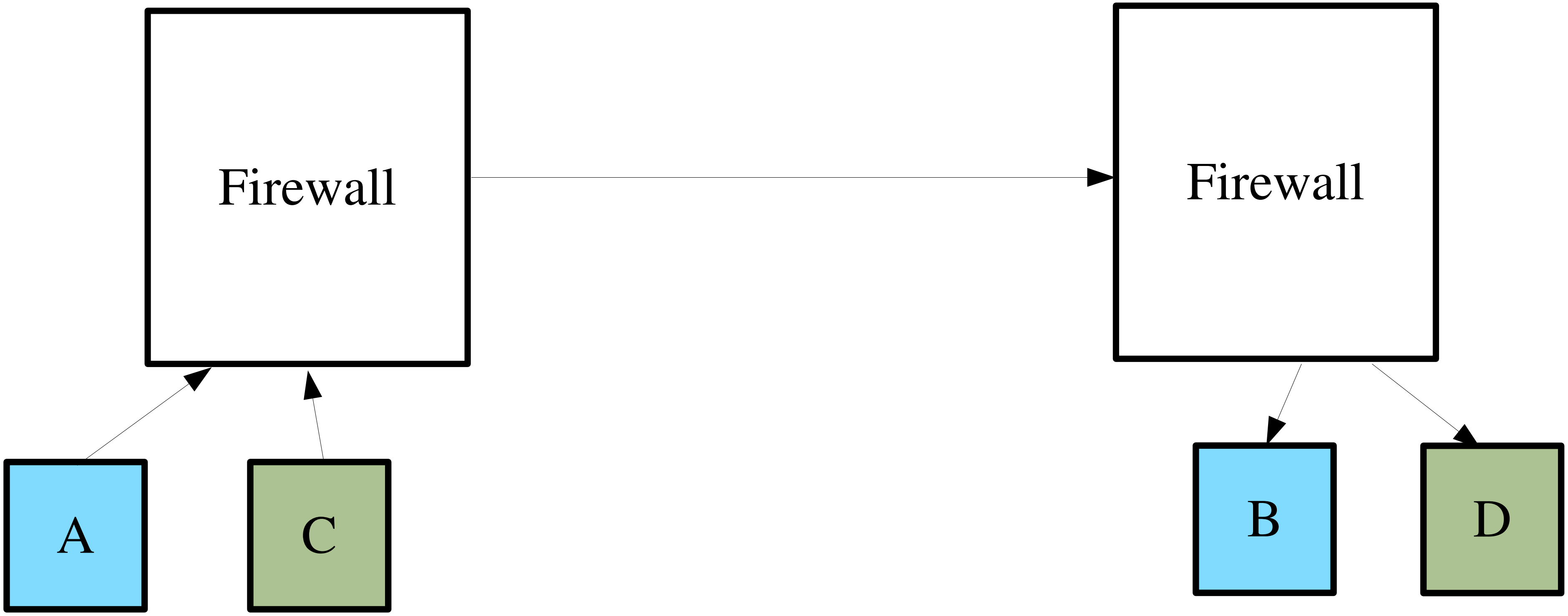
Attacker can only force an acknowledgment, not a Diffie-Hellman calculation.

IKE

Possible Security Problem: (encryption w/o integrity)

C can decrypt packet sent by A to B

- Record packet from A to B and packet from C to D
- Splice the encrypted part contain src-dst from C to D onto A to B
- Forward packet to Firewall, Firewall decrypts, sends result to D



IKE

Internet Key Exchange Phase 1:

Aggressive Mode: Accomplishes mutual authentication in three messages

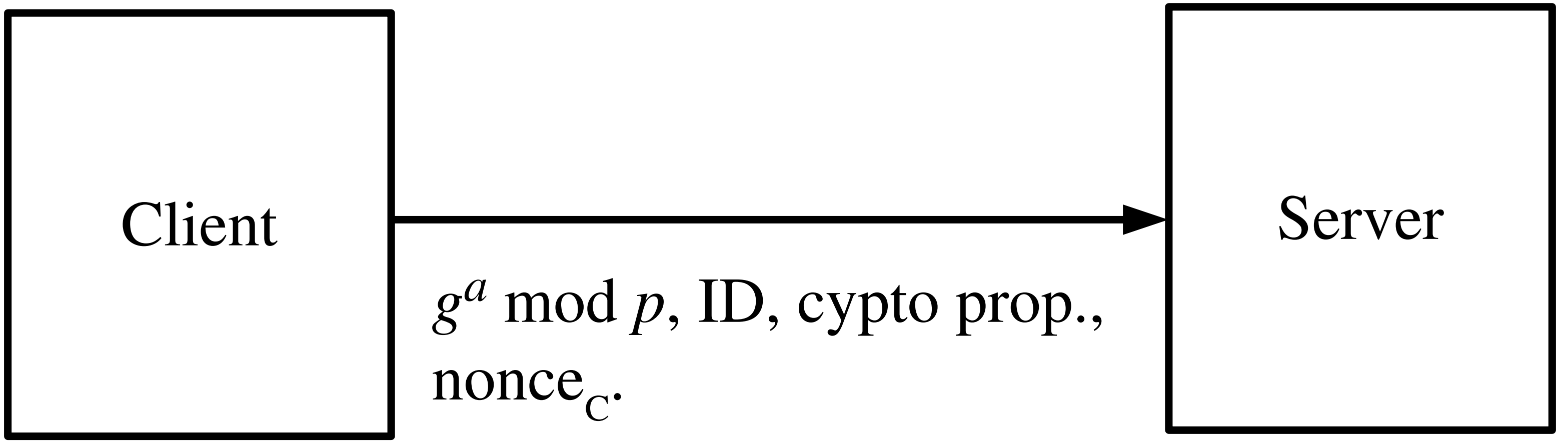
Client

Server

IKE

Internet Key Exchange Phase 1:

Aggressive Mode: Accomplishes mutual authentication in three messages



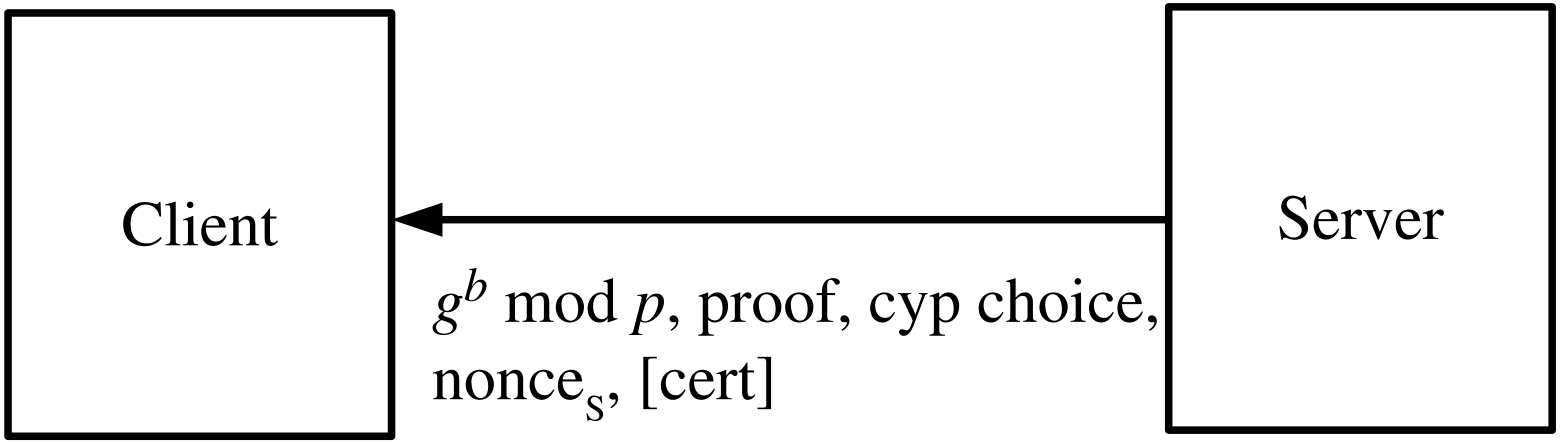
g^a mod p, ID, cypto prop., nonce$_C$.

Diffie-Hellman Exchange

IKE

Internet Key Exchange Phase 1:

Aggressive Mode: Accomplishes mutual authentication in three messages



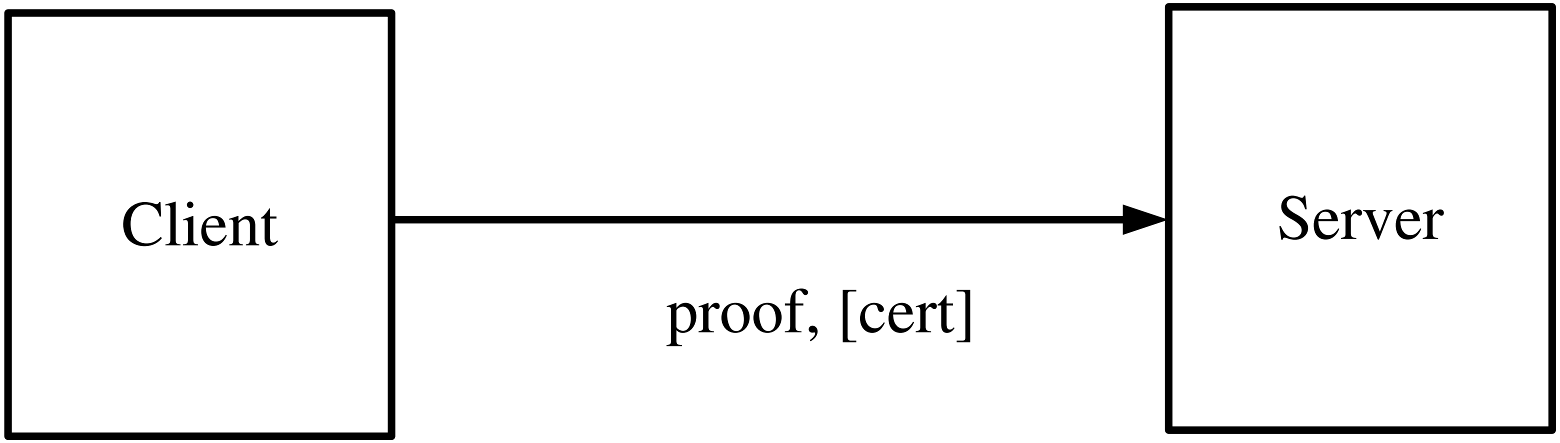
g^b mod p, proof, cyp choice, nonce_S, [cert]

Diffie-Hellman Exchange
proof (of ID) might be a signature

IKE

Internet Key Exchange Phase 1:

Aggressive Mode: Accomplishes mutual authentication in three messages



IKE

Internet Key Exchange Phase 1:

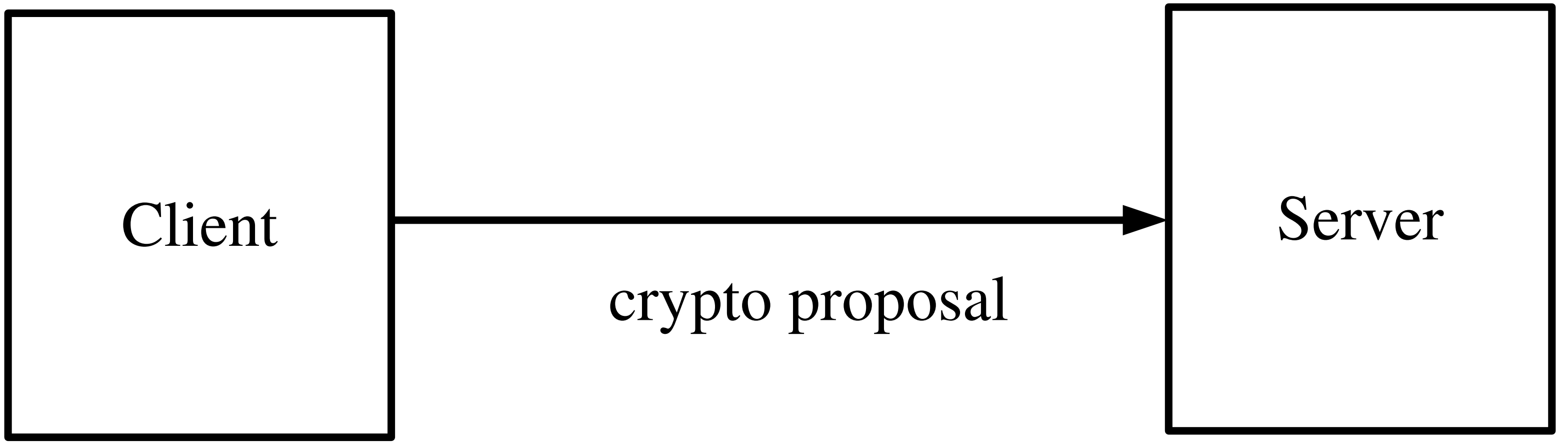
Aggressive Mode Problems:

1. Someone other than Server can send a refusal back to Client and Client cannot tell if it is fake (would want such a message to be sent encrypted).

IKE

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms

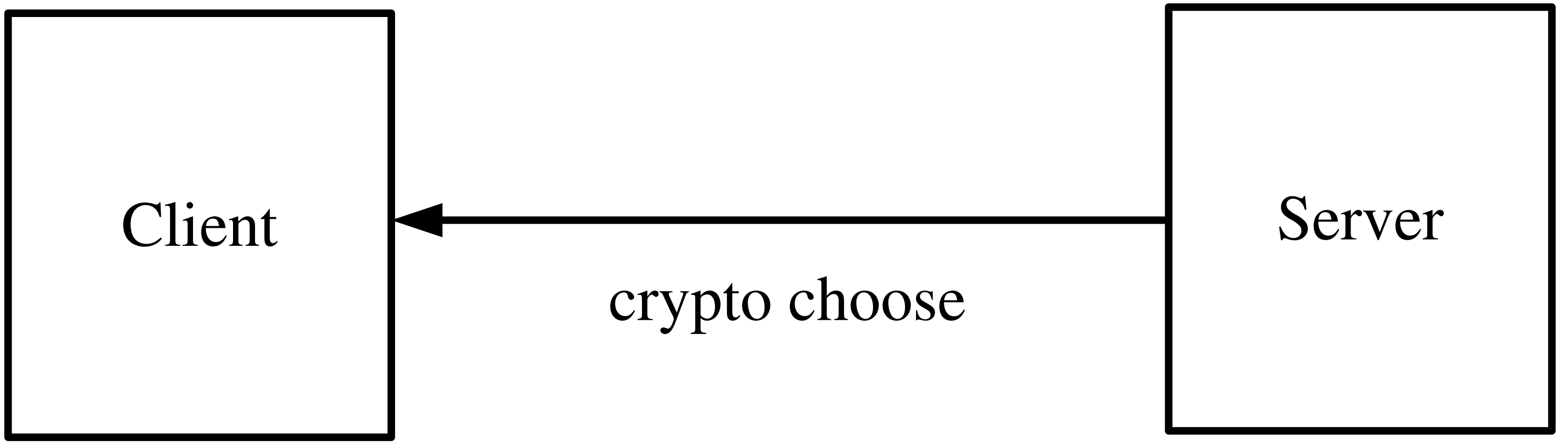


Parameter negotiation

IKE

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms

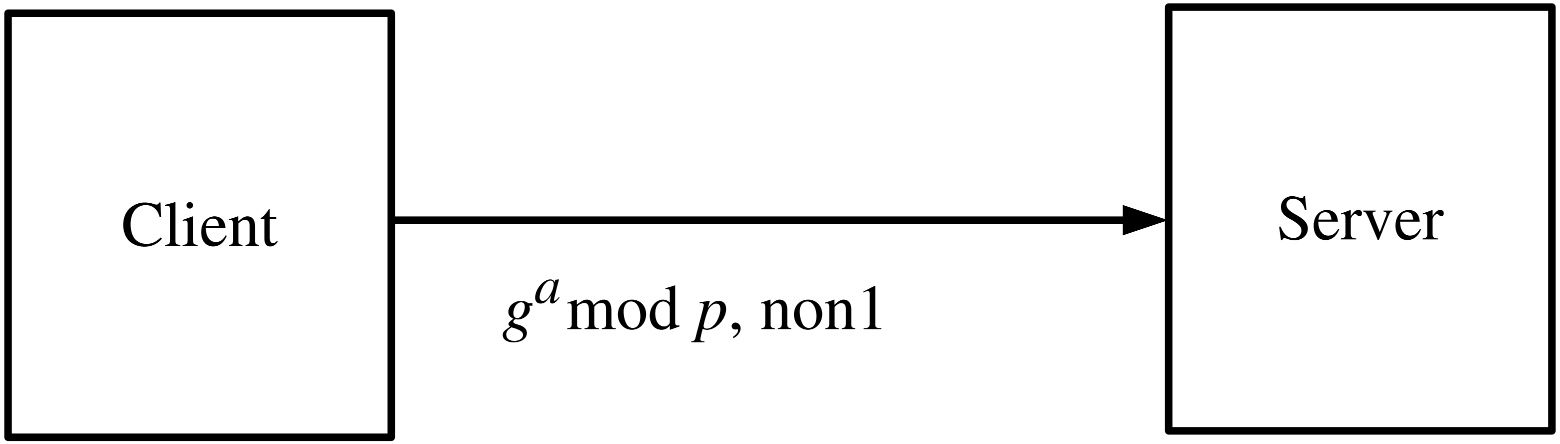


Parameter negotiation

IKE

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms

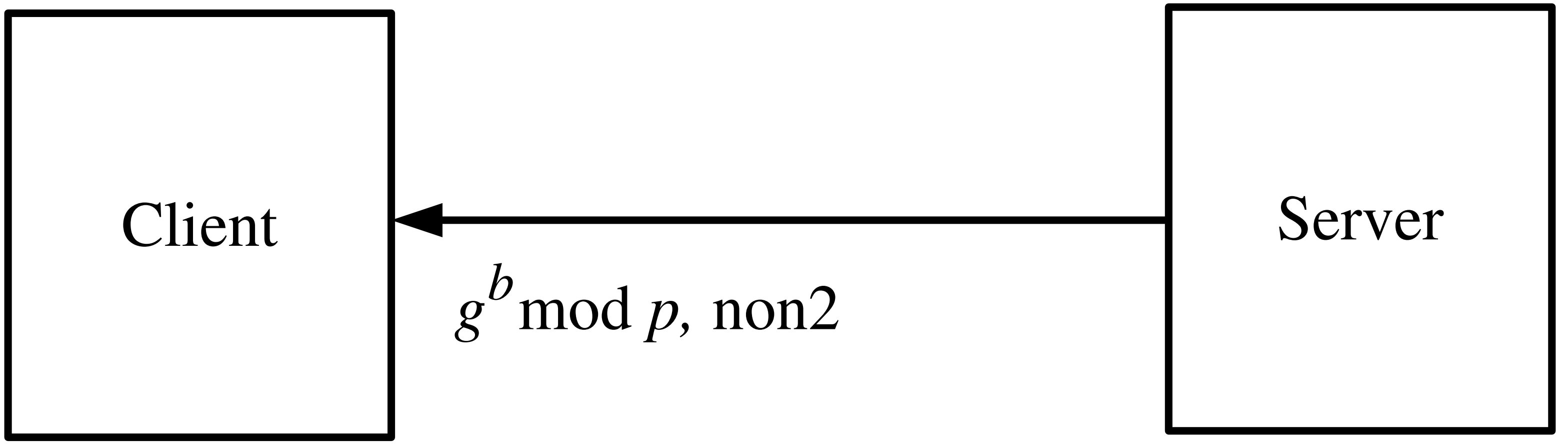


Diffie-Hellman exchange

IKE

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms

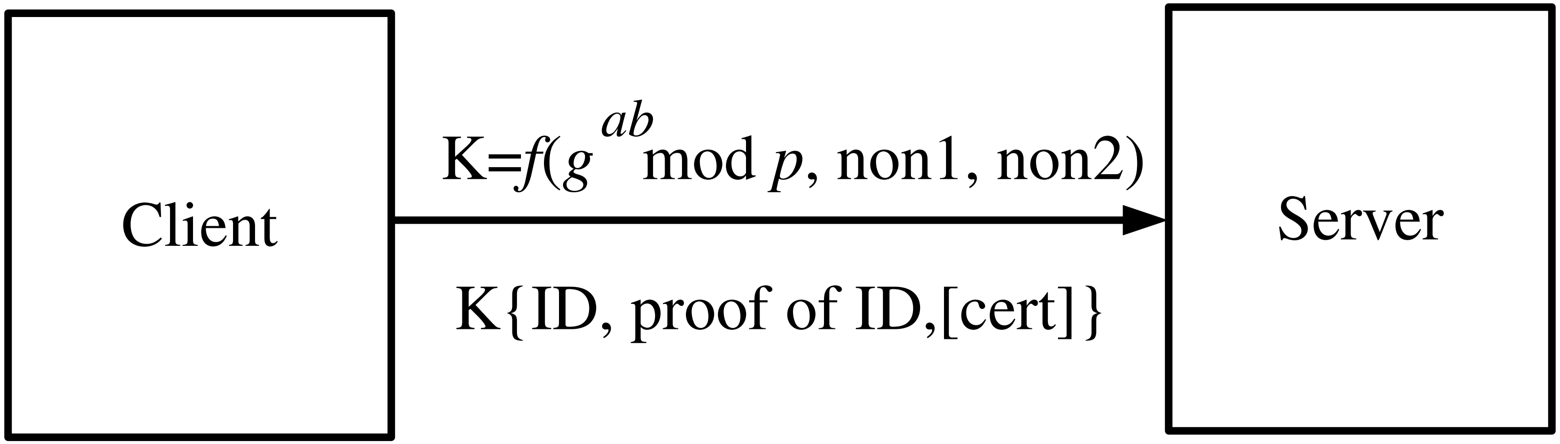


Diffie-Hellman exchange

IKE

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms


Client
$K=f(g^{ab} \bmod p, \text{non1}, \text{non2})$
Server
K{ID, proof of ID,[cert]}

authenticate, encrypted

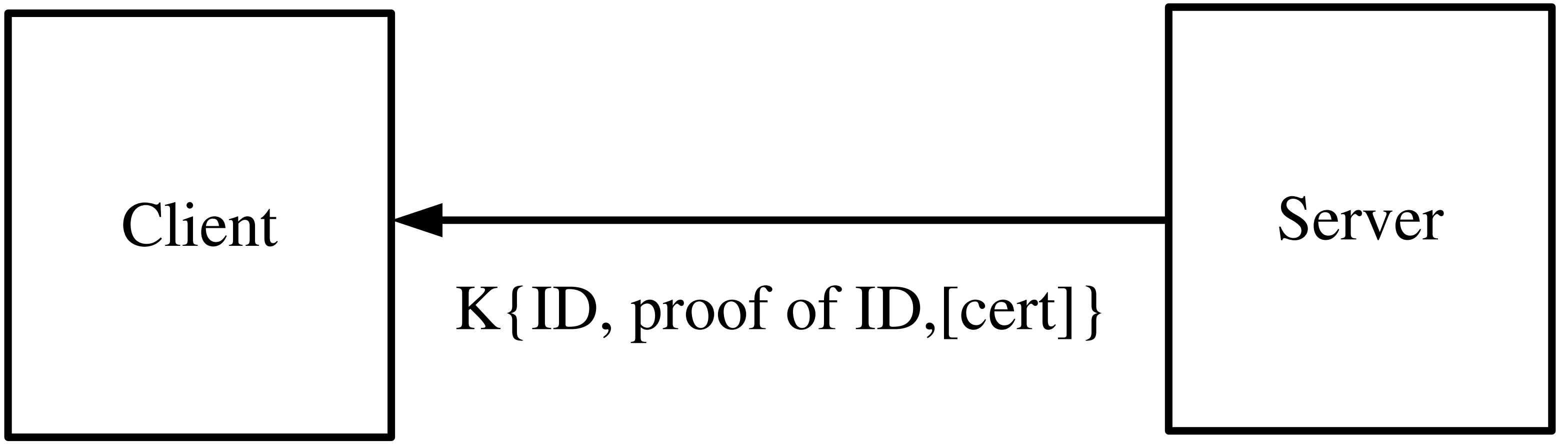
nonces allow same Diffie-Hellman private value for many transactions

proof of ID: signature on a hash of ID, DH values, nonces, crypto choices

IKE

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms



authenticate, encrypted

IKE

Internet Key Exchange Phase 1:

Proof of Identity: Some hash of the key associated with the identity, the Diffie-Hellman values, nonces, cryptographic choices, and cookies.

Problem: choice of cryptographic suite by server is not encrypted. A man-in-the-middle might actually replace a good choice with a poor (crackable) choice then decrypt and impersonate server from then on.

Stateless cookies? No, must remember crypto proposals

Duplicate connection identifiers? Possible to have two connections with the same crypto parameters

IKE

Internet Key Exchange Phase 1:

Crypto Parameters:

1. Encryption algorithm (DES, 3DES, IDEA)
2. Hash algorithm (MD5, SHA)
3. Authentication method (RSA signature, DSS...)
4. Diffie-Hellman group ((g,p), elliptic curves)

IKE

Internet Key Exchange Phase 1:

Certificates: Client nor Server can ask the other side for a certificate. If they do not know the other side's public key they cannot use the protocol. If certificates are sent in first two messages then identities are revealed.

IKE

Internet Key Exchange Phase 1:

Three keys:

Encryption: $K_e = f(K, K_a \mid g^{ab} \bmod p \mid \text{cookie}_a \mid \text{cookie}_b \mid 2)$

Authentication: $K_a = f(K, K_d \mid g^{ab} \bmod p \mid \text{cookie}_a \mid \text{cookie}_b \mid 1)$

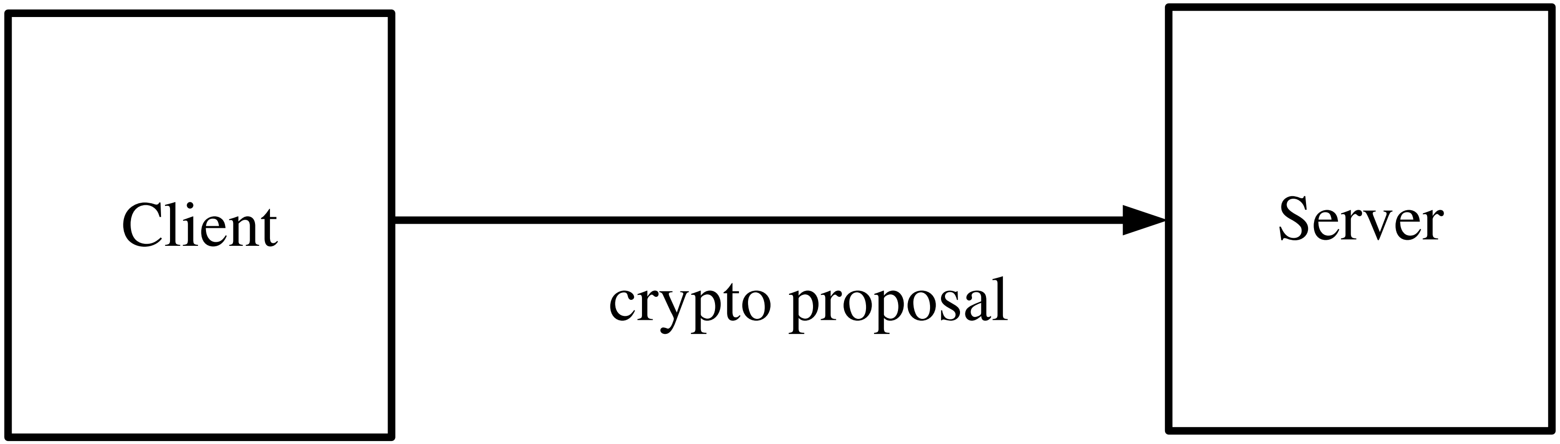
Non-IPSec: $K_d = f(K, g^{ab} \bmod p \mid \text{cookie}_a \mid \text{cookie}_b \mid 0)$

These keys will be used to protect the last phase 1 transaction and all the phase 2 transactions

IKE

Internet Key Exchange Phase 1:

Main Mode Revised: requires a single private key operation on either side.

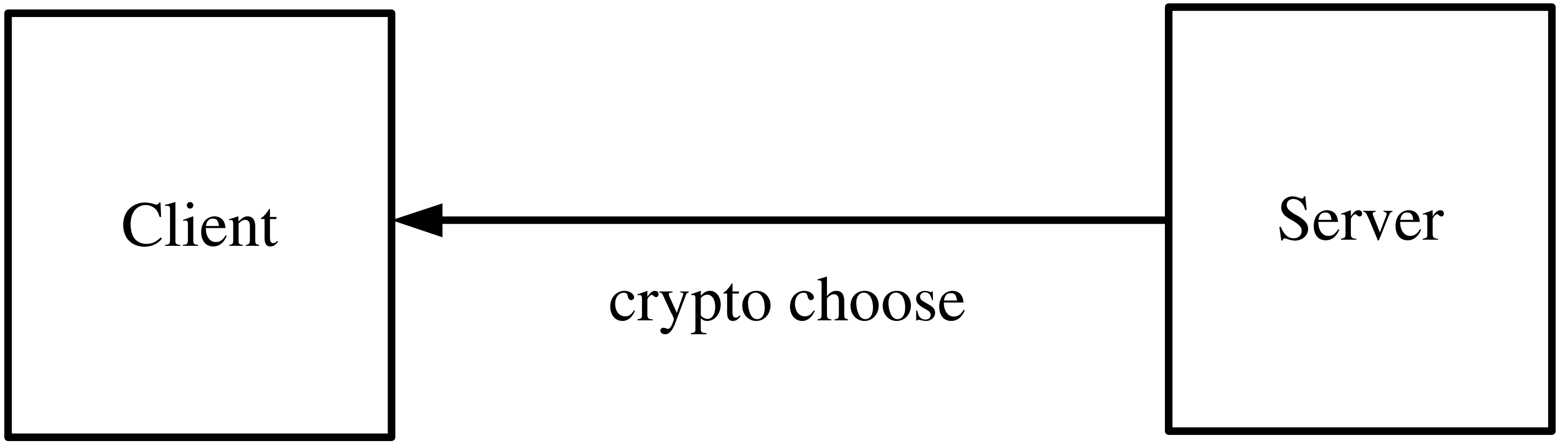


Parameter negotiation
Starts out as before

IKE

Internet Key Exchange Phase 1:

Main Mode Revised:

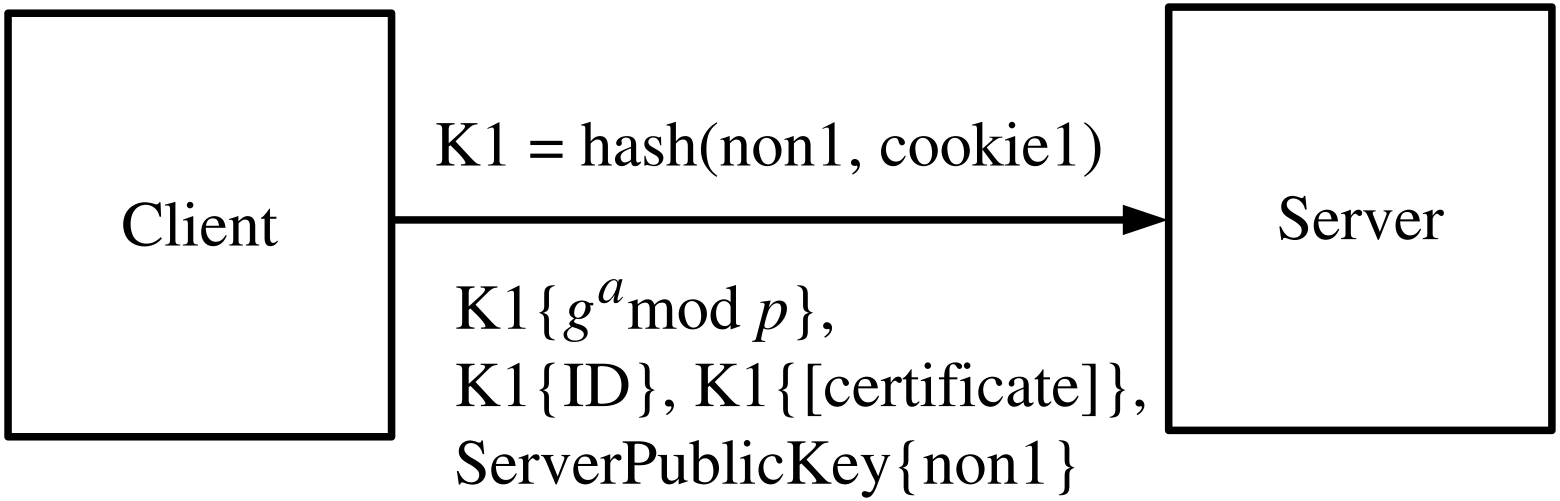


Parameter negotiation
No change yet

IKE

Internet Key Exchange Phase 1:

Main Mode Revised:

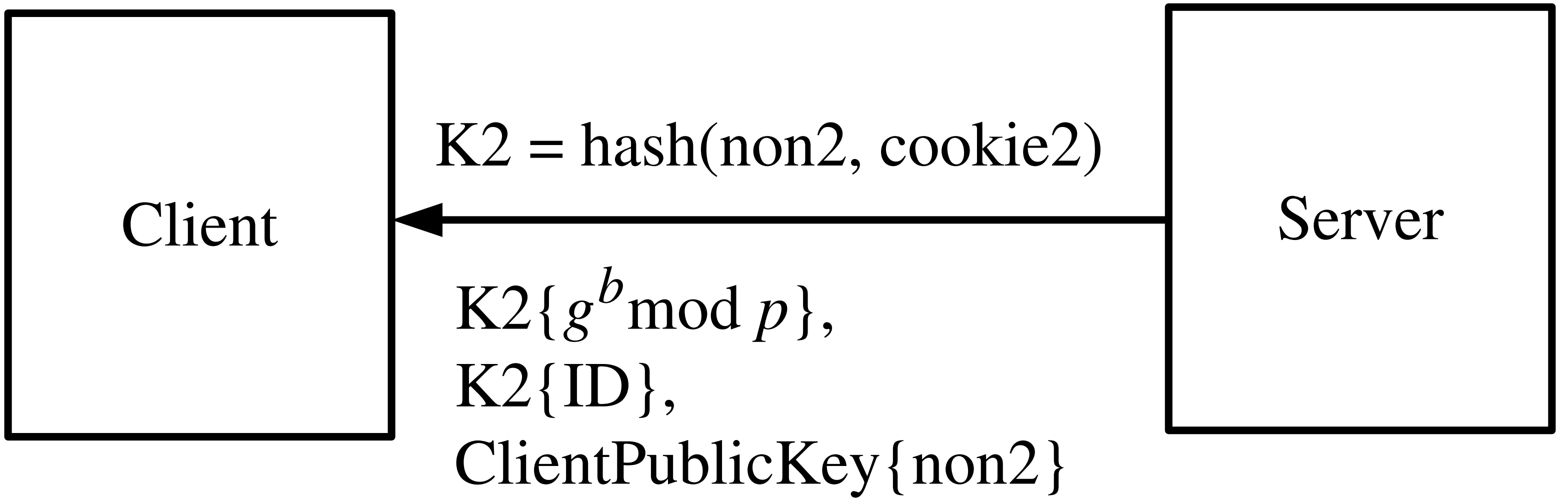


Diffie-Hellman exchange
Server uses private key to decrypt non1 then determines K1
then decrypts ID, and everything else

IKE

Internet Key Exchange Phase 1:

Main Mode Revised:



Client

K2 = hash(non2, cookie2)

Server

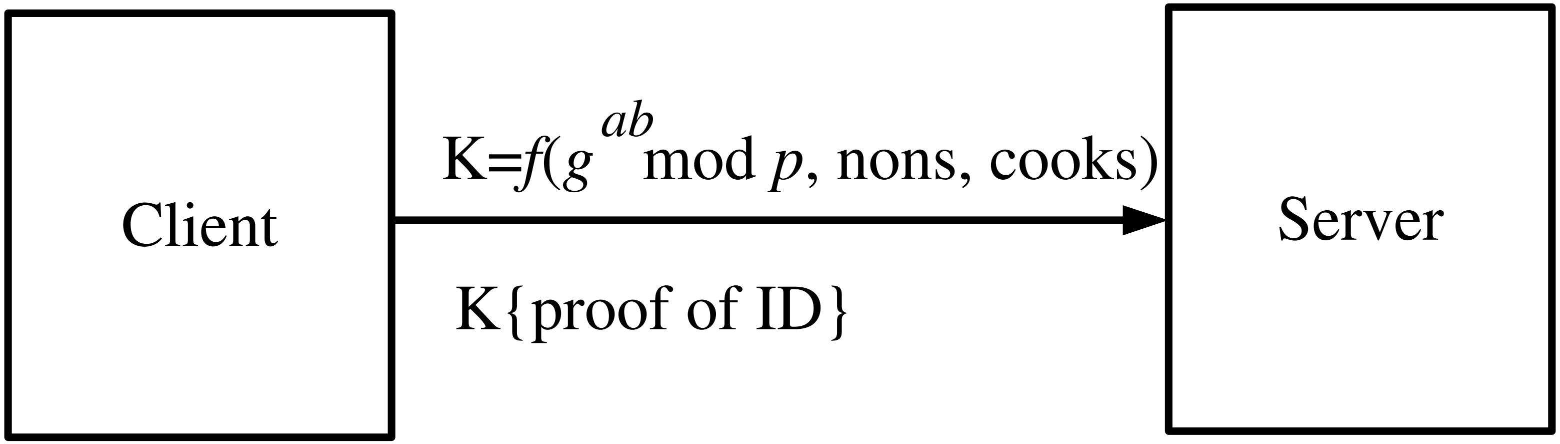
K2{g^bmod p},
K2{ID},
ClientPublicKey{non2}

Diffie-Hellman exchange

IKE

Internet Key Exchange Phase 1:

Main Mode Revised:

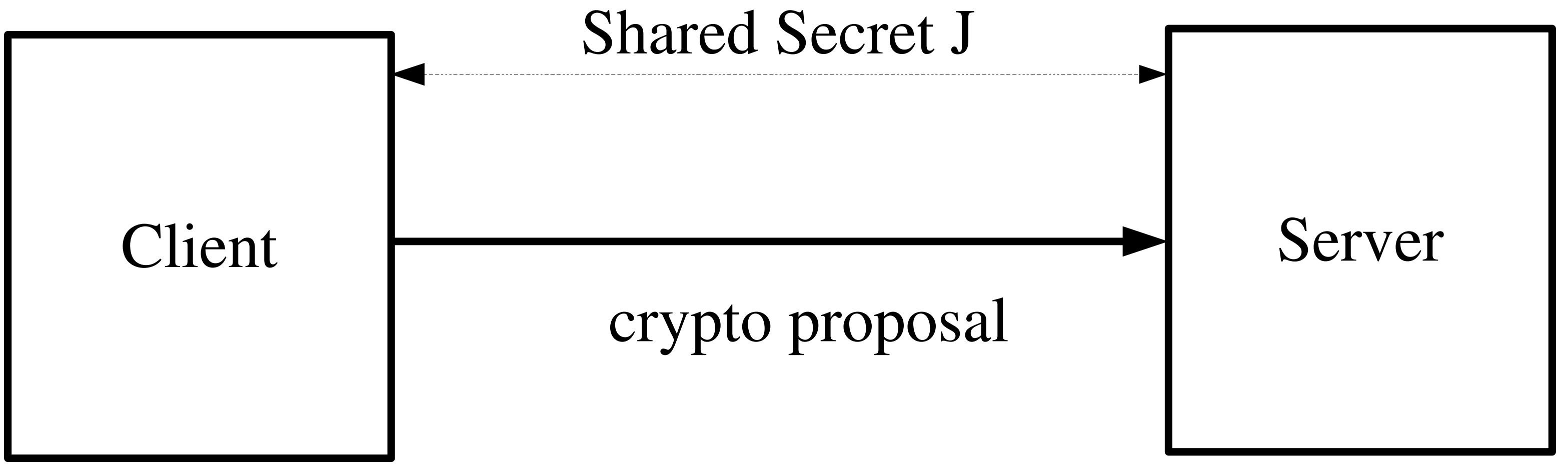


authenticate, encrypted

IKE

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

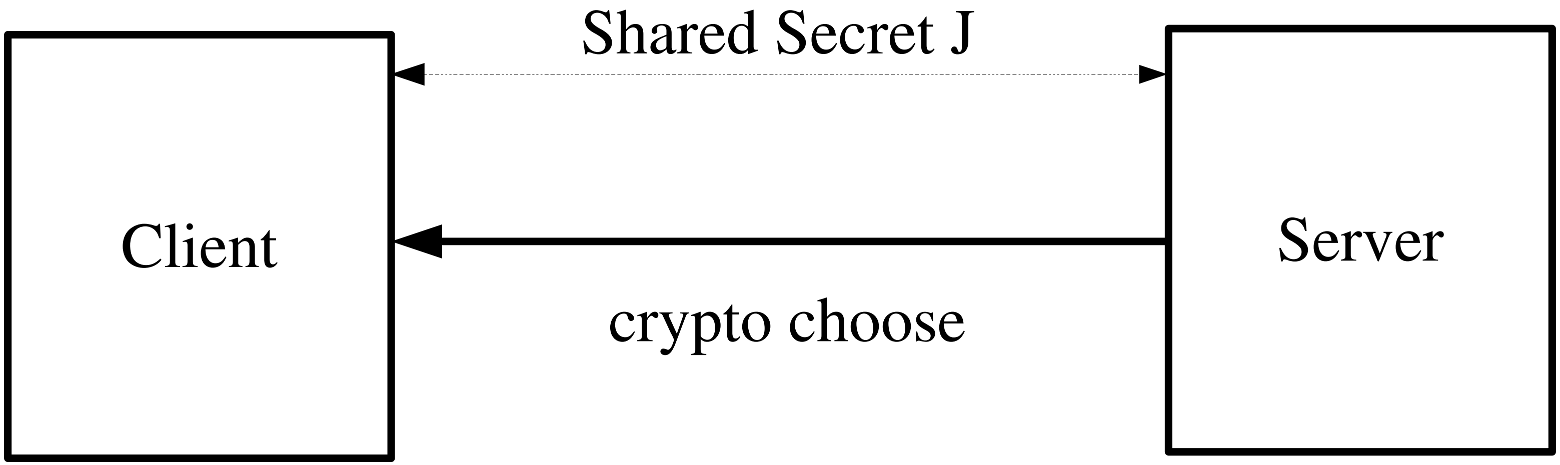


Parameter negotiation

IKE

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

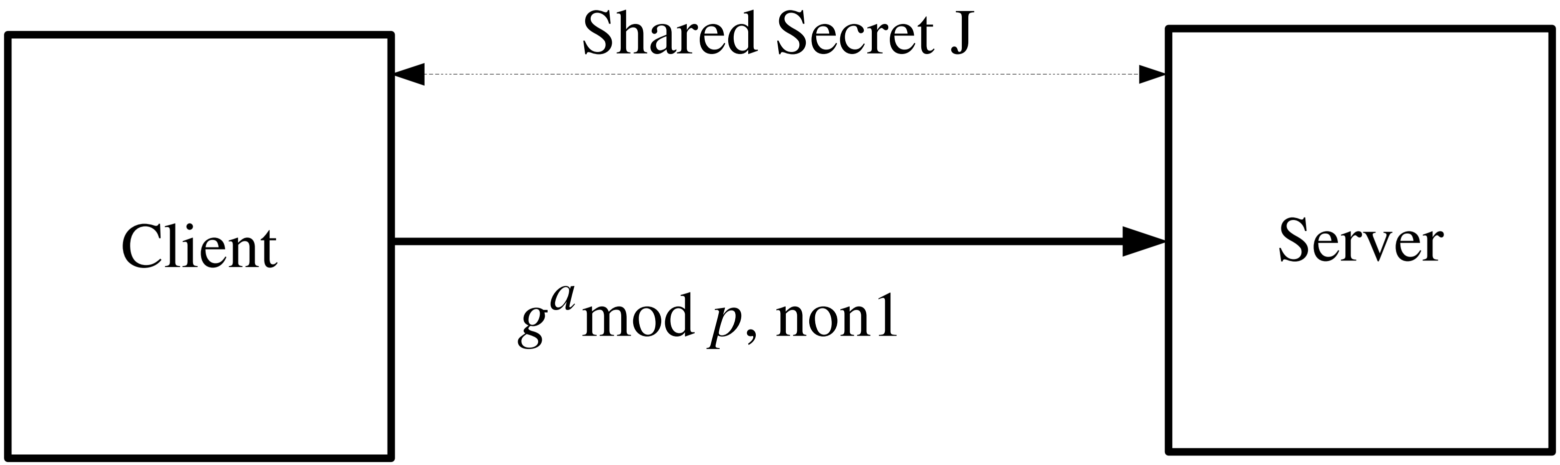


Parameter negotiation

IKE

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

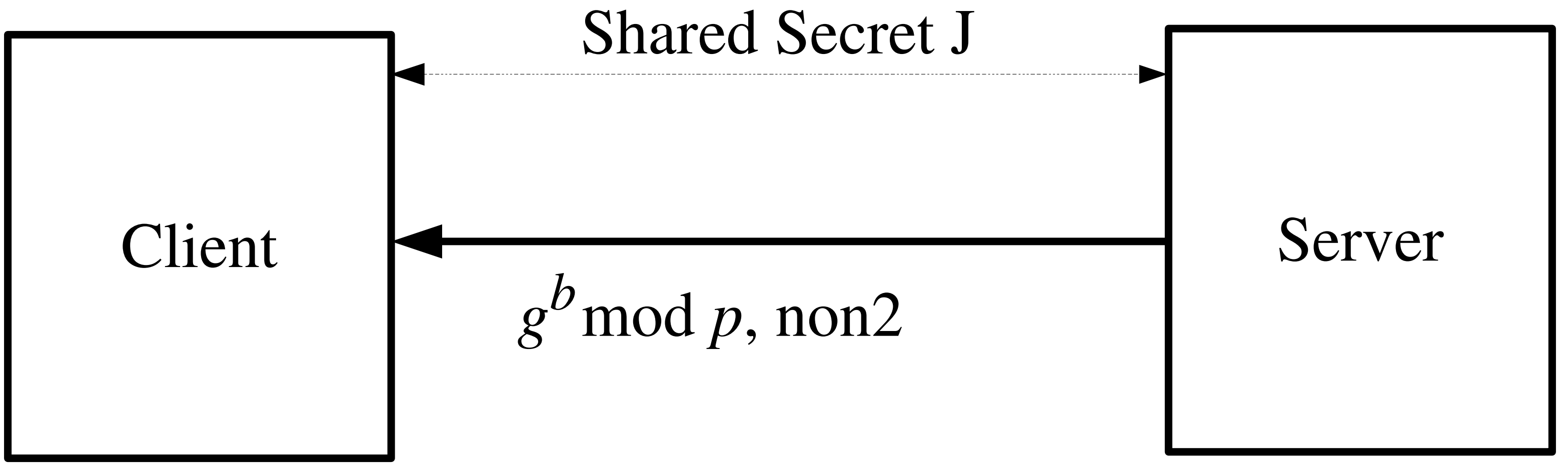


Diffie-Hellman

IKE

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

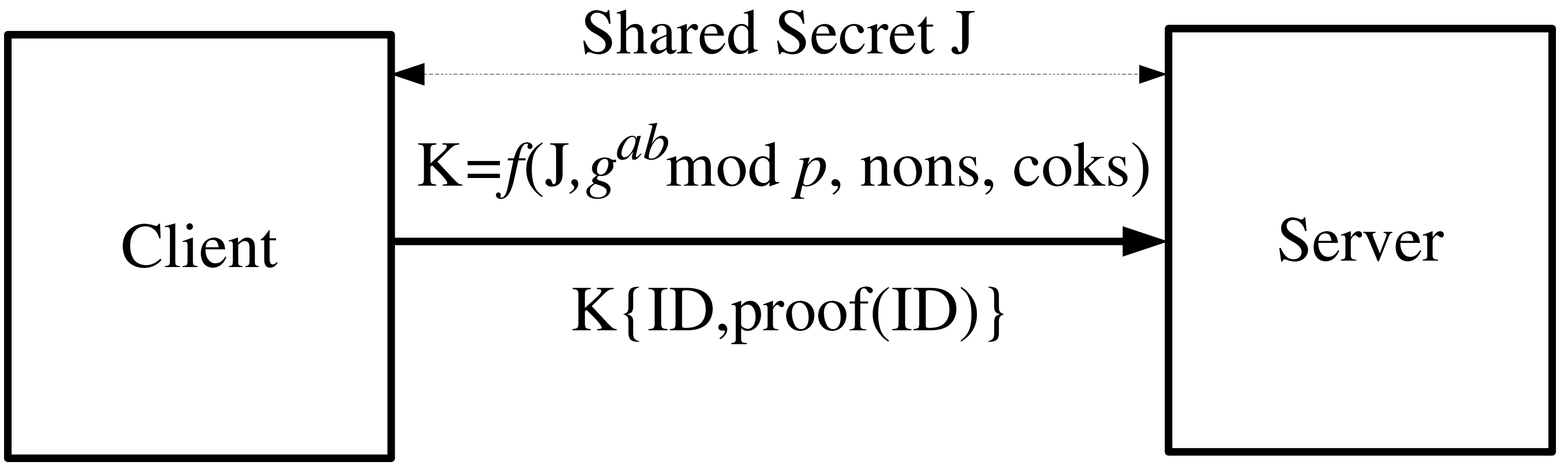


Diffie-Hellman

IKE

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?



Shared Secret J

$K=f(J, g^{ab} \bmod p$, nons, coks)

Client

Server

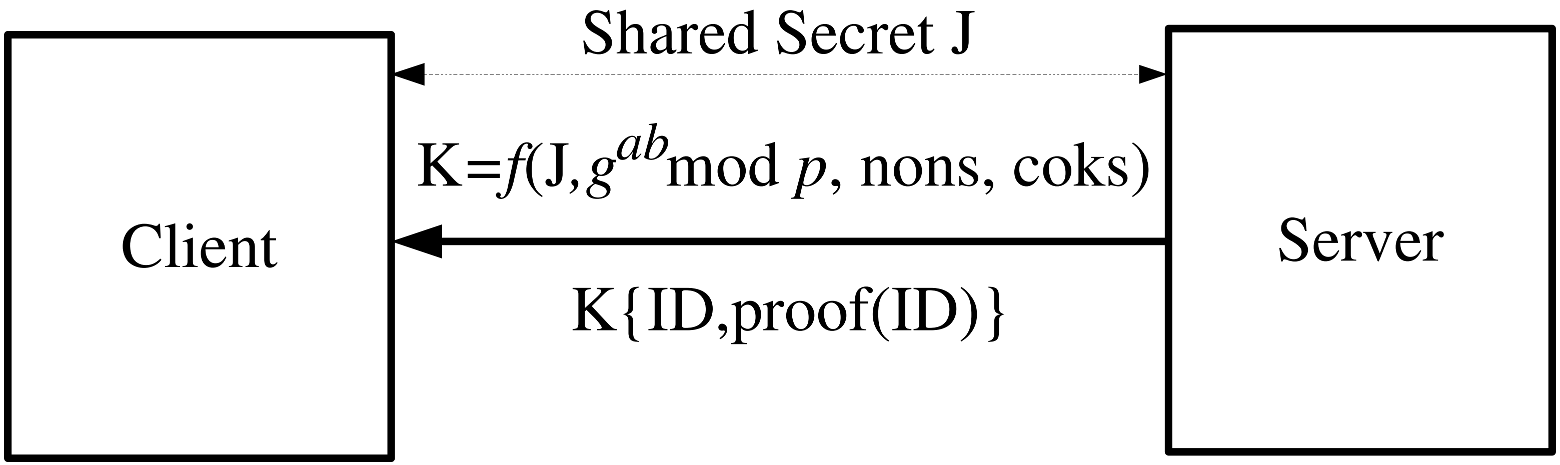
K{ID,proof(ID)}

authentication

IKE

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?



Shared Secret J

K=f(J,g^{ab}mod p, nons, coks)

Client

Server

K{ID,proof(ID)}

authentication

IKE

Internet Key Exchange Phase 1:

Problems:

1. Client sends identity in message 5 encrypted with key K which is a function of shared secret J. Server cannot decrypt that message to find out who the Client is unless it knows J. But that means Server must know who the Client is in the first place! So the specification requires that identities are IP addresses.

2. If identities must be IP addresses, this protocol cannot seriously be used in road warrior application

IKE

Internet Key Exchange Phase 1:

Problems:

1. Client sends identity in message 5 encrypted with key K which is a function of shared secret J. Server cannot decrypt that message to find out who the Client is unless it knows J. But that means Server must know who the Client is in the first place! So the specification requires that identities are IP addresses.

2. If identities must be IP addresses, this protocol cannot seriously be used in road warrior application

Fix:

Do not make K a function of J. OK since J is included in the hash which is proof of identity.

IKE

Internet Key Exchange Phase 1:

Negotiating Cryptographic Parameters:

encryption algorithm: DES, 3DES, IDEA

hash: MD5, SHA

authentication method: pre-shared-keys, RSA signing, RSA encryption, DSS

Diffie-Hellman type: p,g

Session Keys:

Two established: integrity, encryption for protecting the last phase 1 transaction and all the phase 2 transactions

$$K_d = f(K, g^{ab} \bmod p \mid \text{cookie}_a \mid \text{cookie}_b \mid 0)$$

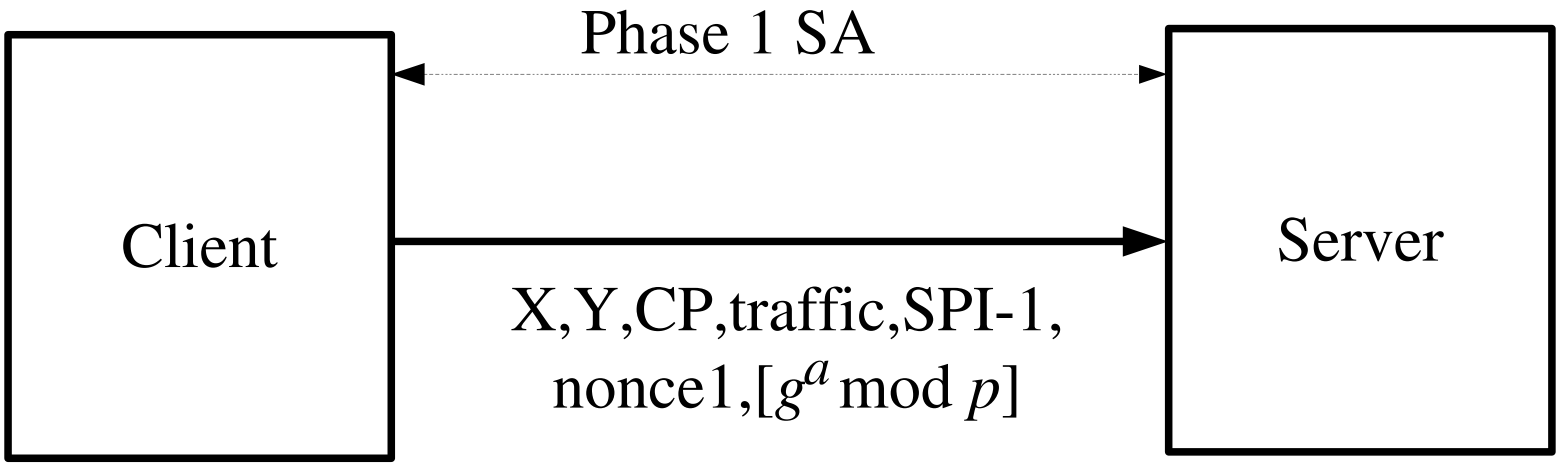
$$K_a = f(K, K_d \mid g^{ab} \bmod p \mid \text{cookie}_a \mid \text{cookie}_b \mid 1)$$

$$K_e = f(K, K_a \mid g^{ab} \bmod p \mid \text{cookie}_a \mid \text{cookie}_b \mid 2)$$

IKE

Internet Key Exchange Phase 2:

Setting up IPSec SAs: All messages are encrypted with Phase 1 SA's encryption key K_e and integrity protected with phase 1 SA's integrity key K_a.



X is a pair of cookies from phase 1

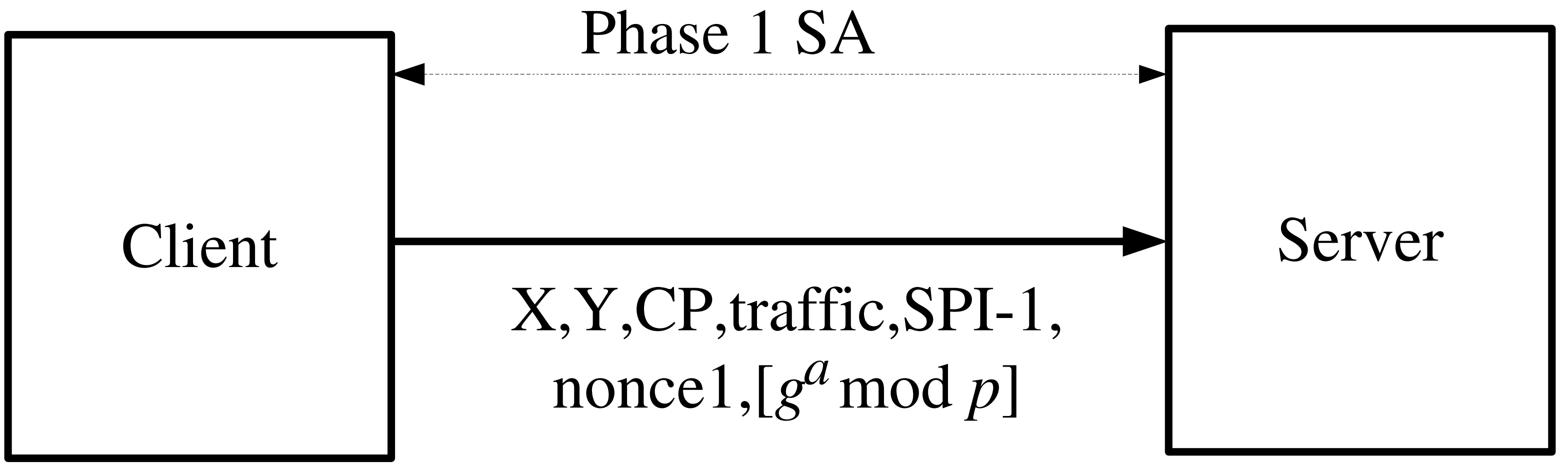
Y is a 32 bit number chosen to distinguish this setup from others that may be setup simultaneously in phase 1.

X and Y are unencrypted.

IKE

Internet Key Exchange Phase 2:

Setting up IPSec SAs: All messages are encrypted with Phase 1 SA's encryption key K_e and integrity protected with phase 1 SA's integrity key K_a.



Rest of message: crypto parameters, optional Diffie-Hellman values for Perfect Forward Secrecy, optional description of traffic.

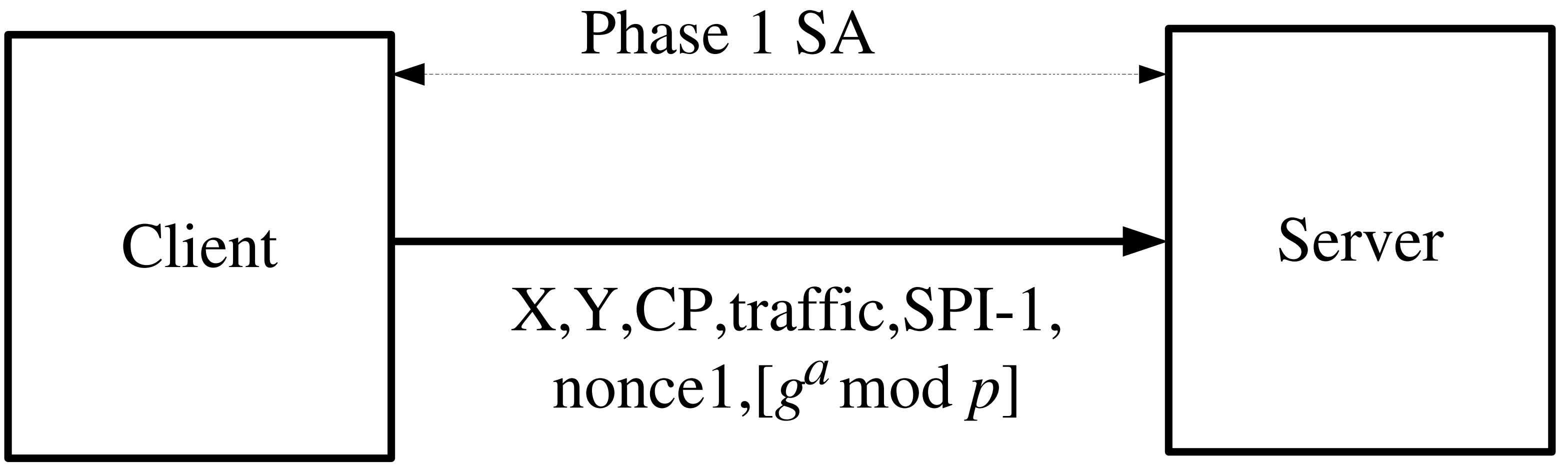
Integrity Protected: with K_a

Encrypted: with K_e

IKE

Internet Key Exchange Phase 2:

Setting up IPSec SAs: All messages are encrypted with Phase 1 SA's encryption key K_e and integrity protected with phase 1 SA's integrity key K_a.

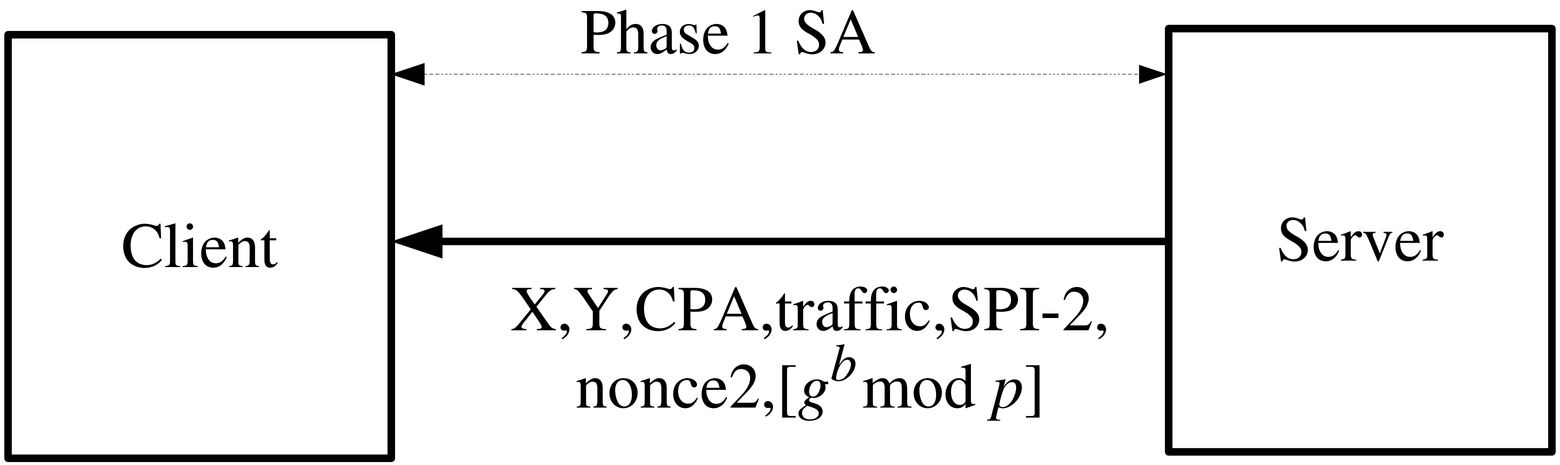


Encryption: Initialization vector is the final ciphertext block of last message of phase 1 hashed with Y.

IKE

Internet Key Exchange Phase 2:

Setting up IPSec SAs:

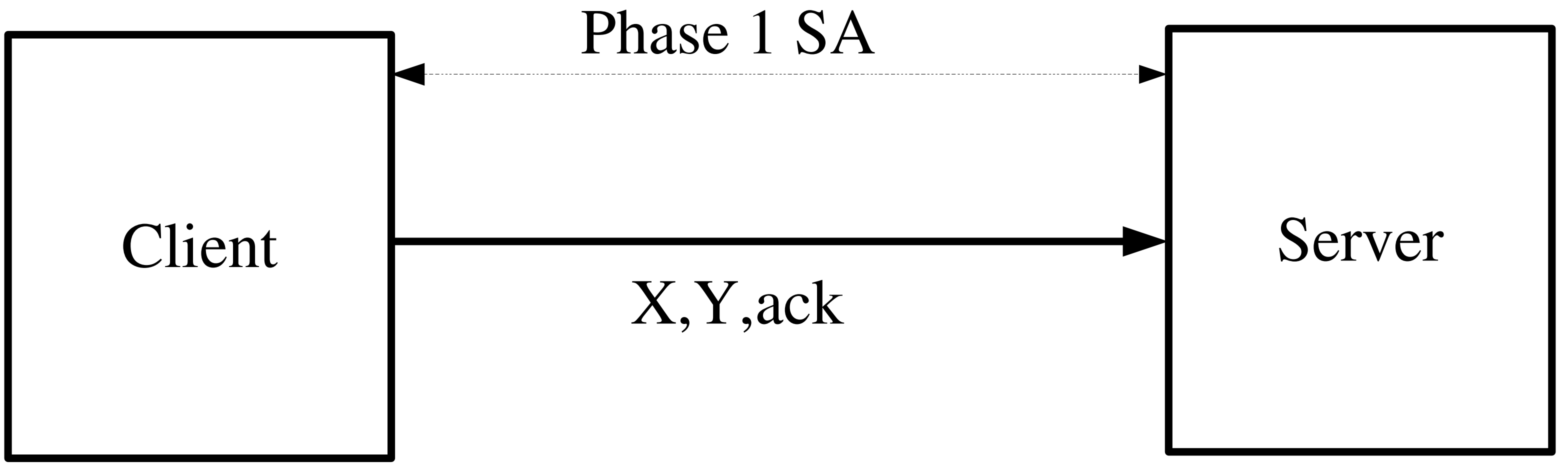


Encryption: Initialization vector is the final ciphertext block of last message of previous phase 2 message hashed with Y.

IKE

Internet Key Exchange Phase 2:

Setting up IPSec SAs:



Encryption: Initialization vector is the final ciphertext block of last message of previous phase 2 message hashed with Y.

IKE

Internet Key Exchange Phase 2:

Results:

New Keying material:

Keymat = $f(K_d$, protocol | [$g^{xy} \bmod p$ |] SPI | nonce1 | nonce2)

parties decide how to use the keying material to generate six keys for the session.

IKE

Internet Key Exchange Phase 2:

Problems:

1. Can be vulnerable to replay:
   a. If Y is "random" instead of based on a sequence #, to detect a replay attack one must remember all Y's generated.
   b. If headers and session keys are the same in both directions, attacker can replay easily.
2. Can be vulnerable to reflection attack.

What to do:

Use different keys in different directions.

Use sequence numbers instead of message IDs.

IKE

ISAKMP/IKE Encoding:

Messages have a fixed header followed by a sequence of payloads. Each payload starts with "type of next payload" and "length of this payload".

IKE

Fixed Header:

(64 bits)	initiator's cookie	
(64 bits)	responder's cookie	
(8 bits)	next payload type	
(32 bits)	version	exchange type
(80 bits)	flags	message ID
(64 bits)	message length	

payload type:

End, SA, Proposal, Transform (crypto choices), Key Exchange, ID, Certificate, Certificate Request, Checksum (hash), signature, nonce, Notification, delete (closing the SPI), vendor ID (for telling the Implementation being used)

IKE

Fixed Header:

(64 bits)	initiator's cookie	
(64 bits)	responder's cookie	
(8 bits)	next payload type	
(32 bits)	version	exchange type
(80 bits)	flags	message ID
(64 bits)	message length	

exchange type: base - adds extra message to aggresive mode to allow DH negot.
identity protection (main mode)
authentication only
aggressive
informational

flags: encrypted, commit, authentication only (set only during phase 2),

messageID: differentiates messages with same phase 1 SA

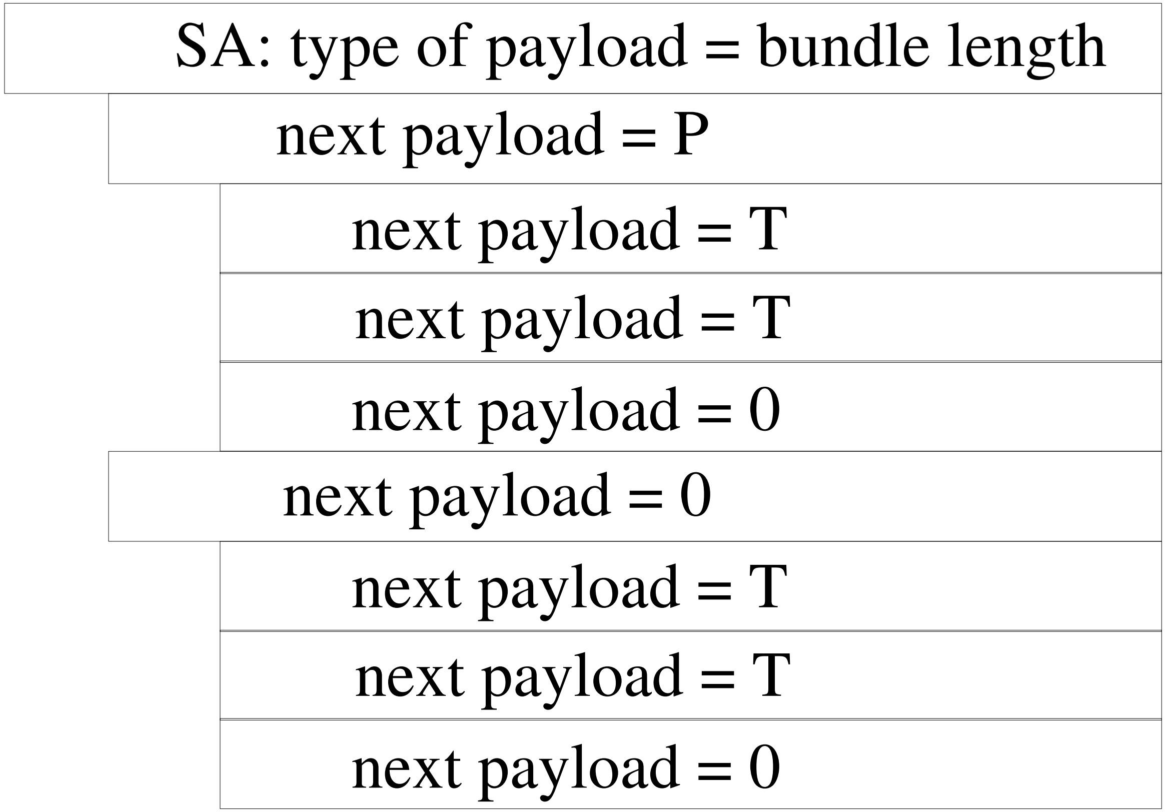
IKE

Payload, starting fields:

(8 bits)	next type of payload
(8 bits)	unused
(16 bits)	length of this payload

IKE

Example, cryptochoices:



SA: type of payload = bundle length

next payload = P

next payload = T

next payload = T

next payload = 0

next payload = 0

next payload = T

next payload = T

next payload = 0