Sistema de comunicaciones seguras con segmentación virtual de dominios

Avance de proyecto

Alberto Lange

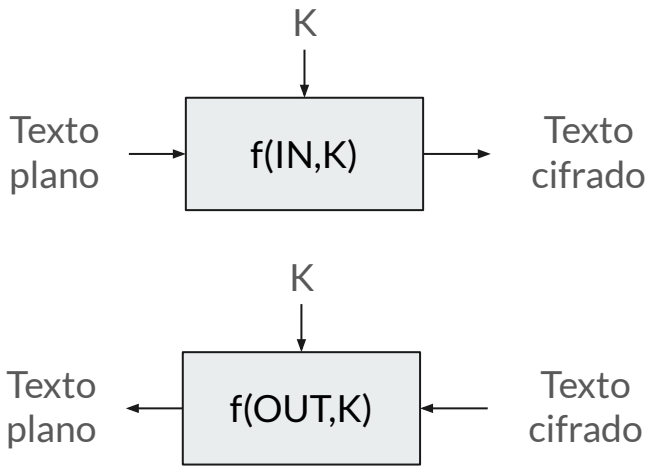
Temas

- Encriptación simétrica y asimétrica
- Intercambio de claves
- Wireguard
- Framework NOISE

Encriptación

Simétrica

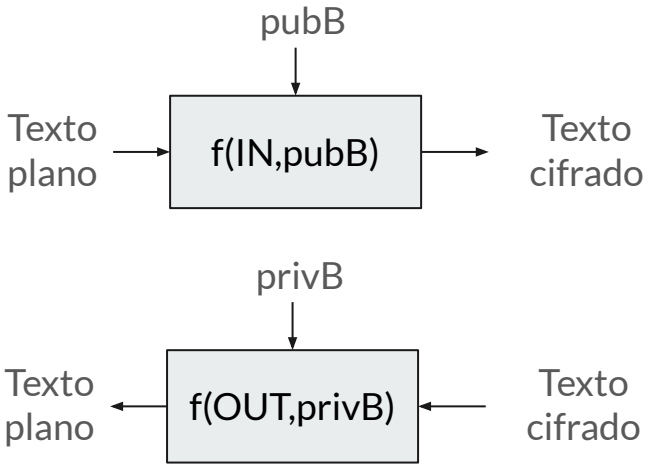
- Clave única.
- Método eficiente.
- Requiere un canal seguro para el intercambio de la clave.
- La confidencialidad y autenticación dependen tanto de A como de B.



Encriptación

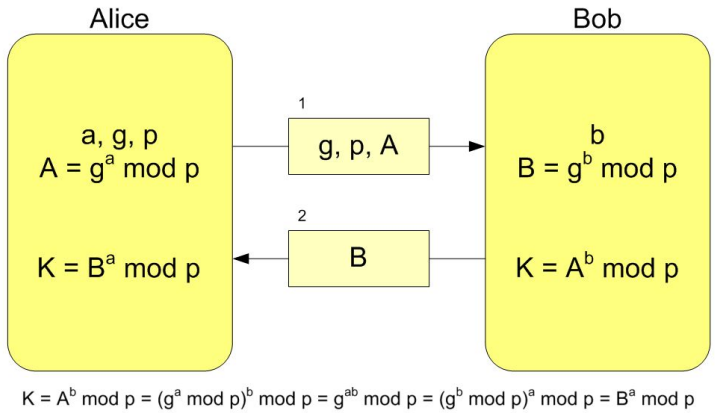
Asimétrica

- Par de claves.
- Mantener la autenticidad y confidencialidad de lo que recibe A depende solo de A.
- Mayor costo computacional.



Acuerdo de claves Diffie-Hellman

- Método para generar una clave simétrica sin transmitirla por el canal.
- Mitiga uno de los problemas de la encriptación simétrica.
- Pueden implementarse acuerdos con autenticación (firmas digitales).



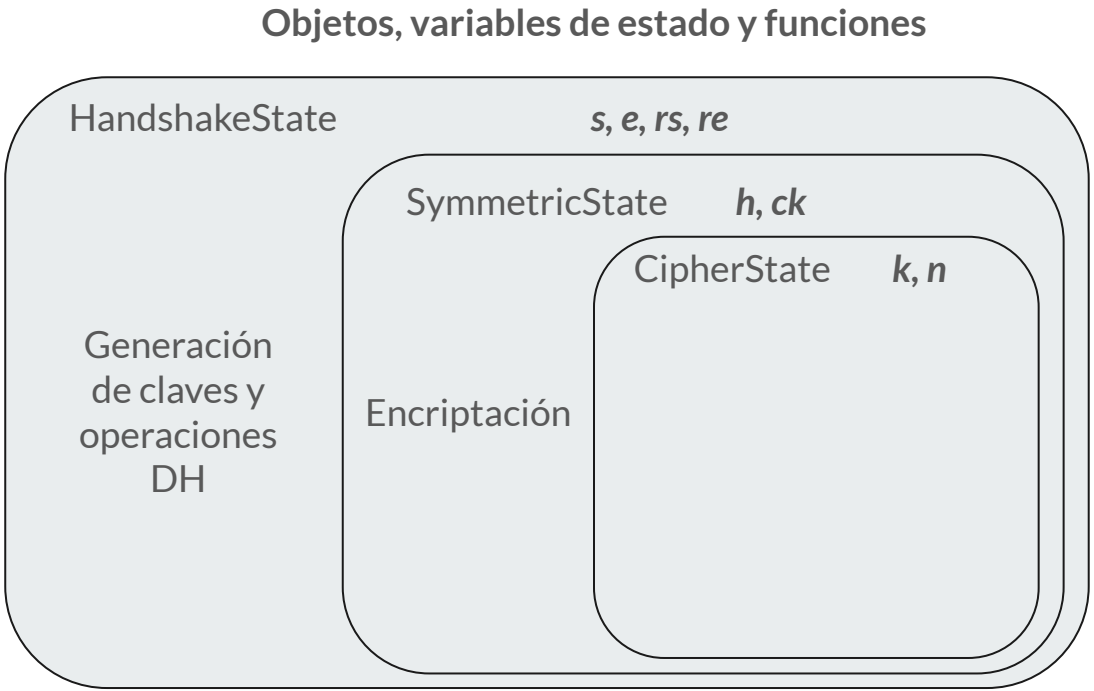
NOISE

- Permite implementar protocolos para el intercambio de claves públicas (mensajes de handshake) y derivación de una clave simétrica con la cual encriptar mensajes de transporte.
- Propone patrones de handshake validados según ciertos criterios de autenticación y confidencialidad.

IK:

<- s
...
-> e, es, s, ss
<- e, ee, se

NOISE


Objetos, variables de estado y funciones
HandshakeState
s, e, rs, re
SymmetricState
h, ck
CipherState
k, n
Generación de claves y operaciones DH
Encriptación

NOISE: handshake

A	IK	B
e, s, re, rs	<- s	e, s, re, rs
h, ck	...	h, ck
k, n	-> e, es, s, ss	k, n
	<- e, ee, se	

NOISE: handshake

A	IK	B
e, s, re, **rs**	<- **s**	e, **s**, re, rs
h, ck	…	h, ck
k, n	-> e, es, s, ss	k, n
	<- e, ee, se	

NOISE: handshake

A		IK		B
e, s, re, rs h, ck k, n	---->	<- s ... -> e, es, s, ss <- e, ee, se	---->	e, s, re, rs h, ck k, n

NOISE: handshake

A	IK	B
e, s, re, rs	<- s	e, s, re, rs
h, ck	...	h, ck
k, n	-> e, es, s, ss	k, n
	<- e, ee, se	

NOISE: handshake

A		IK		B
e, s, re, rs h, ck k, n	--(k)-->	<- s ... -> e, es, s, ss <- e, ee, se	--(k)-->	e, s, re, rs h, ck k, n

NOISE: handshake

A	IK	B
e, s, re, rs	<- s	e, s, re, rs
h, ck	...	h, ck
k, n	-> e, es, s, ss	k, n
	<- e, ee, se	

Autenticación grado 1: la autenticación de A es vulnerable a KCI.
Confidencialidad grado 2: FS solamente para A. B es vulnerable a replay attacks.

NOISE: handshake

IK

<- s

...

-> e, es, s, ss

<- e, ee, se

type $:= 0x1$ (1 byte)	reserved $:= 0^3$ (3 bytes)
sender $:= I_i$ (4 bytes)	
ephemeral (32 bytes)	
static ($\widehat{32}$ bytes)	
timestamp ($\widehat{12}$ bytes)	
mac1 (16 bytes)	mac2 (16 bytes)

Primer mensaje de handshake - Wireguard

NOISE: handshake

A		IK		B
e, s, re, rs h, ck k, n	<----	<- s ... -> e, es, s, ss <- e, ee, se	<----	e, s, re, rs h, ck k, n

NOISE: handshake

A	IK	B
e, s, re, rs	<- s	e, s, re, rs
h, ck	...	h, ck
k, n	-> e, es, s, ss	k, n
	<- e, ee, se	

NOISE: handshake

A	IK	B
e, s, re, rs	<- s	e, s, re, rs
h, ck	...	h, ck
k, n	-> e, es, s, ss	k, n
	<- e, ee, se	

Autenticación grado 2: resistente a KCI.
Confidencialidad grado 4: weak forward-secrecy. → Mensajes posteriores (transporte) adquieren grado 5, strong forward-secrecy.

NOISE: handshake

IK

<- s

...

-> e, es, s, ss

<- e, ee, se

type := 0x2 (1 byte)	reserved := 0^3 (3 bytes)
sender := I_r (4 bytes)	receiver := I_i (4 bytes)
ephemeral (32 bytes)	
empty ($\widehat{0}$ bytes)	
mac1 (16 bytes)	mac2 (16 bytes)

Segundo mensaje de handshake - Wireguard

NOISE: post-handshake

type := 0x4 (1 byte)	reserved := 0^3 (3 bytes)
receiver := $I_{m'}$ (4 bytes)	
counter (8 bytes)	
packet ($\widehat{\|P\|}$ bytes)	

Mensaje de transporte - Wireguard

Wireguard

- VPN que opera en capa 3.
- Asocia claves públicas con direcciones IP.
- Intercambio de claves basado en Noise IK.
- ChaCha20 y Poly1305 como algoritmos de cifrado y autenticación.
- Utiliza el concepto de máquina de estado sincrónica.
- Diseñado según simplicidad, código auditable y alta velocidad.

Wireguard: Cryptokey routing

Source: 10.0.0.1
Destination: 10.0.0.2 → Encripta con clave simétrica asociada

Desencripta usando la clave simétrica asociada al identificador del header

IP → wg0 → (UDP) → wg0 → IP

Source: 10.0.0.1
Destination: 10.0.0.2

Source: 10.0.0.1's endpoint
Destination: 10.0.0.2's endpoint

Si el paquete desencriptado es IP y proviene de 10.0.0.1 se acepta. Se actualiza el endpoint de A usando el source de UDP.

Interface Public Key	**Interface Private Key**	**Listening UDP Port**
HIgo...8ykw	yAnz...fBmk	41414
Peer Public Key	**Allowed Source IPs**	
xTIB...p8Dg	10.192.122.3/32, 10.192.124.0/24	
TrMv...WXX0	10.192.122.4/32, 192.168.0.0/16	
gN65...z6EA	10.10.10.230/32	

Wireguard: Protocolo

- Intercambio de claves 1-RTT.
- Par de claves simétricas para mensajes de transporte.
- Uso de timestamps para prevenir replay attacks (retransmisión de mensajes de handshake).
- Uso de cookies para prevenir DoS attacks (procesamiento de handshake y validación del iniciador).

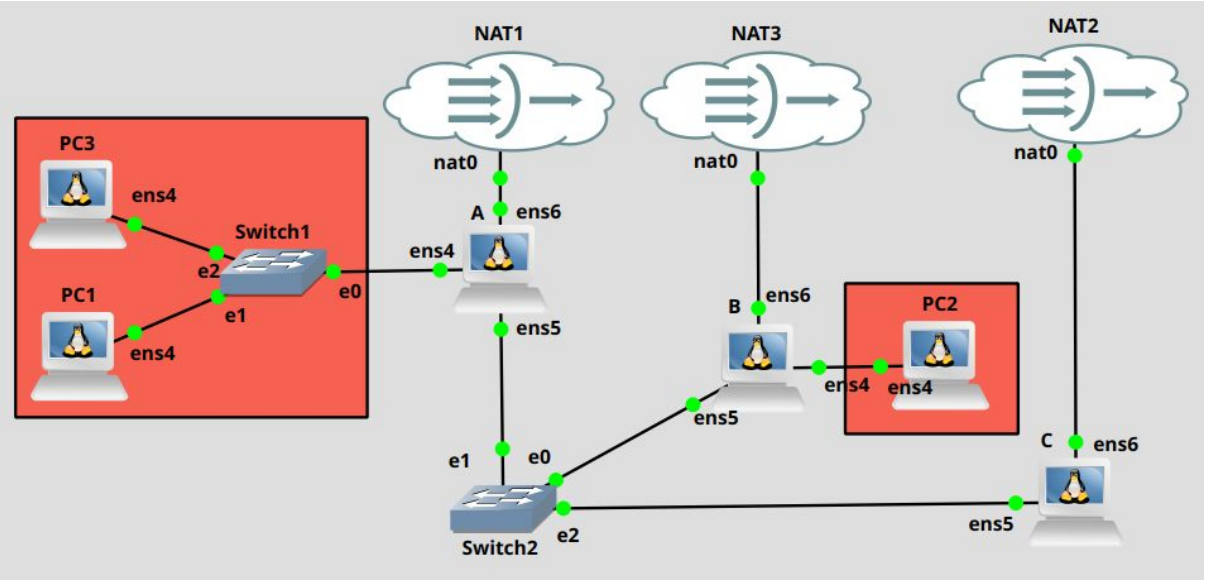
Wireguard: Timers

- Rotación de claves.
- Retransmisión de mensaje inicial de handshake.
- Passive Keepalive.

Symbol	Value
REKEY-AFTER-MESSAGES	2^{60} messages
REJECT-AFTER-MESSAGES	$2^{64} - 2^{13} - 1$ messages
REKEY-AFTER-TIME	120 seconds
REJECT-AFTER-TIME	180 seconds
REKEY-ATTEMPT-TIME	90 seconds
REKEY-TIMEOUT	5 seconds
KEEPALIVE-TIMEOUT	10 seconds

Sistema de comunicaciones seguras

Esquema general



Segmentación virtual: Arquitectura

Dominios rojo/negro.

VMs