# Redes Privadas Virtuales (VPN)


HACKED
©Beyond Security® All rights reserved
www.SecuriTeam.com
BY DALE BRADEN
I'M NOT USING THIS CIRCUIT UNTIL I KNOW FOR SURE THAT OUR CONVERSATION WILL OCCUR OVER A SECURE CONNECTION!
BRADEN

# Redes Privadas virtuales

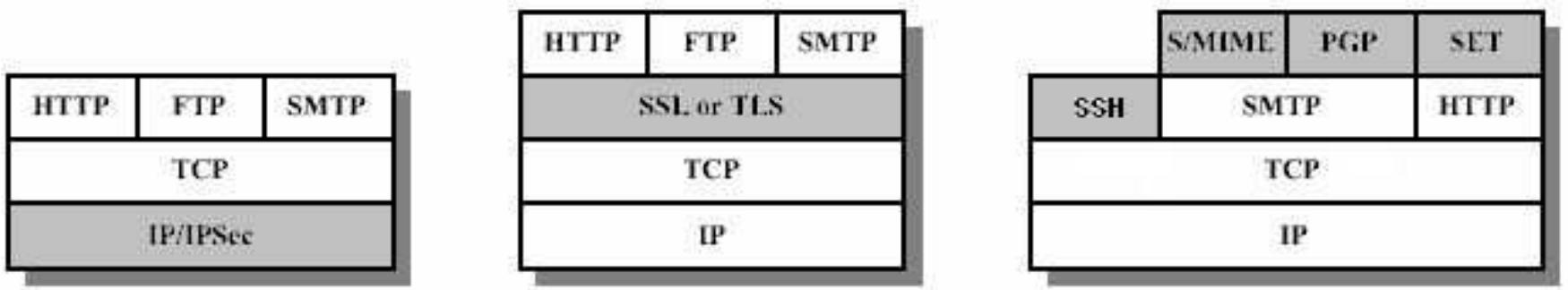
Se pueden definir redes privadas virtuales utilizando protocolos de distinto nivel. Ejemplos mas utilizados:

IPSEC, componente obligatorio en IPv6 , opcional para IPv4

VPN Basada en SSL. Se puede asegurar un servicio  o toda la comunicación entre dos redes.

VPN Basada en SSH. Se pueden hacer túneles para un solo servicio, o túneles dinámicos con socks.

# Protocolos


HTTP
FTP
SMTP
TCP
IP/IPSec
HTTP
FTP
SMTP
SSL or TLS
TCP
IP
S/MIME
PGP
SET
SSH
SMTP
HTTP
TCP
IP

# ¿Por qué IPSEC?

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Verificación de los extremos de la comunicación
  - Verificar que un paquete se originó realmente en la contraparte.
- Integridad de los datos
  - Saber que los paquetes no han sido alterados.
- Confidencialidad
  - Saber que nadie ha visto el contenido de la comunicación.
- Protección contra el Replay
  - A veces no hace falta conocer el contenido del paquete para causar efectos indeseados. Por ejemplo capturar un paquete con el comando de formatear el disco y luego reenviarlo a otro equipo.
- Confidencialidad del flujo de datos
  - Que nadie sepa que estamos comunicándonos.

# IPSEC: Seguridad en IP

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Específico para IP.
  - Opcional en IP v4.
  - Implementación obligatoria en IP v6.
- Transparente para aplicaciones y usuarios.
  - *Virtual Private Network* (VPN).
  - Usuario remoto, puede ser móvil *(road warrior).*

# IPSEC RFCs

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

| | |
|---|---|
| RFC 1828 : IP Authentication using Keyed MD5 | |
| RFC 1829 : The ESP DES-CBC Transform | |
| RFC 1851 : The ESP Triple DES Transform (status experimental) | |
| RFC 2085 : HMAC-MD5 IP Authentication with Replay Prevention | Feb 97 |
| RFC 2104 : HMAC: Keyed-Hashing for Message Authentication | Feb 97 |
| **RFC 2401 : Security Architecture for the Internet Protocol** | **Nov 98** |
| **RFC 2402 : IP Authentication Header** | **Nov 98** |
| **RFC 2403 : The Use of HMAC-MD5-96 within ESP and AH** | **Nov 98** |
| **RFC 2404 : The Use of HMAC-SHA-1-96 within ESP and AH** | **Nov 98** |
| **RFC 2405 : The ESP DES-CBC Cipher Algorithm With Explicit IV** | **Nov 98** |
| **RFC 2406 : IP Encapsulating Security Payload (ESP)** | **Nov 98** |
| **RFC 2407 : The Internet IP Security Domain of Interpretation for ISAKMP** | **Nov 98** |
| **RFC 2408 : Internet Security Association and Key Management Protocol (ISAKMP)** | **Nov 98** |
| **RFC 2409 : The Internet Key Exchange (IKE)** | **Nov 98** |
| **RFC 2410 : The NULL Encryption Algorithm and Its Use With IPSec** | **Nov 98** |
| **RFC 2411 : IP Security Document Roadmap** | |
| **RFC 2412 : The OAKLEY Key Determination Protocol** | **Nov 98** |
| **RFC 2451 : The ESP CBC-Mode Cipher Algorithms** | **Nov 98** |

# Modos de operación


- Transporte: protege los datos de la capa de transporte, **extremo a extremo**.
  → Típicamente, entre *hosts.*


- Túnel: protege el datagrama IP completo, **encapsulándolo** dentro de otro.
  → Obligatorio cuando uno o ambos extremos se comportan como *gateway.*

# Tres protocolos

- *Internet Key Exchange* (IKE).
  - **Negociación** de parámetros e intercambio seguro de **claves**.
- *Authentication Header* (AH).
  - Proporciona **autenticación** e **integridad**.
- *Encapsulating Security Payload* (ESP).
  - Además, puede proporcionar **confidencialidad**.

# IKE

**Internet Key Exchange (IKE)** es el protocolo utilizado en IPSEC para intercambiar claves. Está definido en el RFC 2409. Utiliza intercambio de claves con el método de Diffie-Hellman para definir una clave secreta de sesión. Utiliza UDP/500

IKE opera en dos fases: La fase uno establece un canal para seguro. En la fase dos se eligen los protocolos a usar.

Se pueden utilizar 3 mecanismos de autenticación mutua:

- Preshared-keys
- Pares de claves (firma digital)
- Certificados Digitales.

# Authentication Header (AH)

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Servicios de seguridad:
  - Integridad
  - Autenticación del origen de los datos
  - Anti-replay (opcional)

- Emplea MAC:
  - HMAC-MD5
  - HMAC-SHA1

# Authentication Header (AH)

El protocolo AH se define como el protocolo 51 de IP

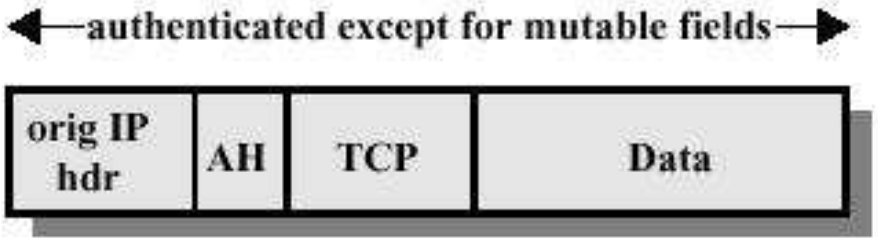
Su objetivo es proveer verificación de la fuente, integridad de paquetes y anti-replay pero no se preocupa por la confidencialidad de los datos o del flujo de datos.

El header de autenticación esta insertado en dos posibles lugares:

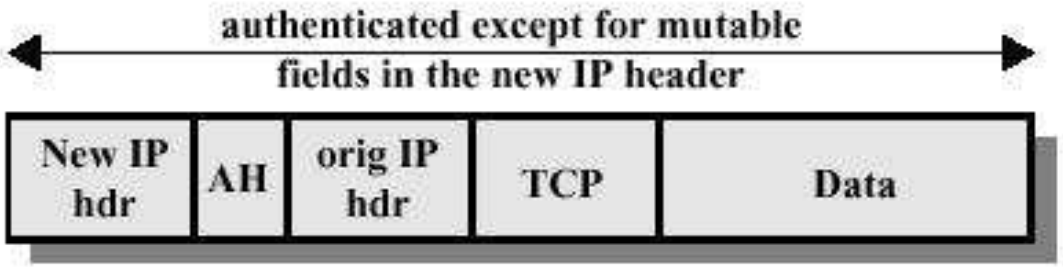
1. Entre el header IP original y el IP data payload (Modo transporte)
2. Como un prefijo del header original al cual se le agrega un nuevo header IP (Modo tunel).

# Authentication Header (AH)


(a) Before Applying AH




(b) Transport Mode



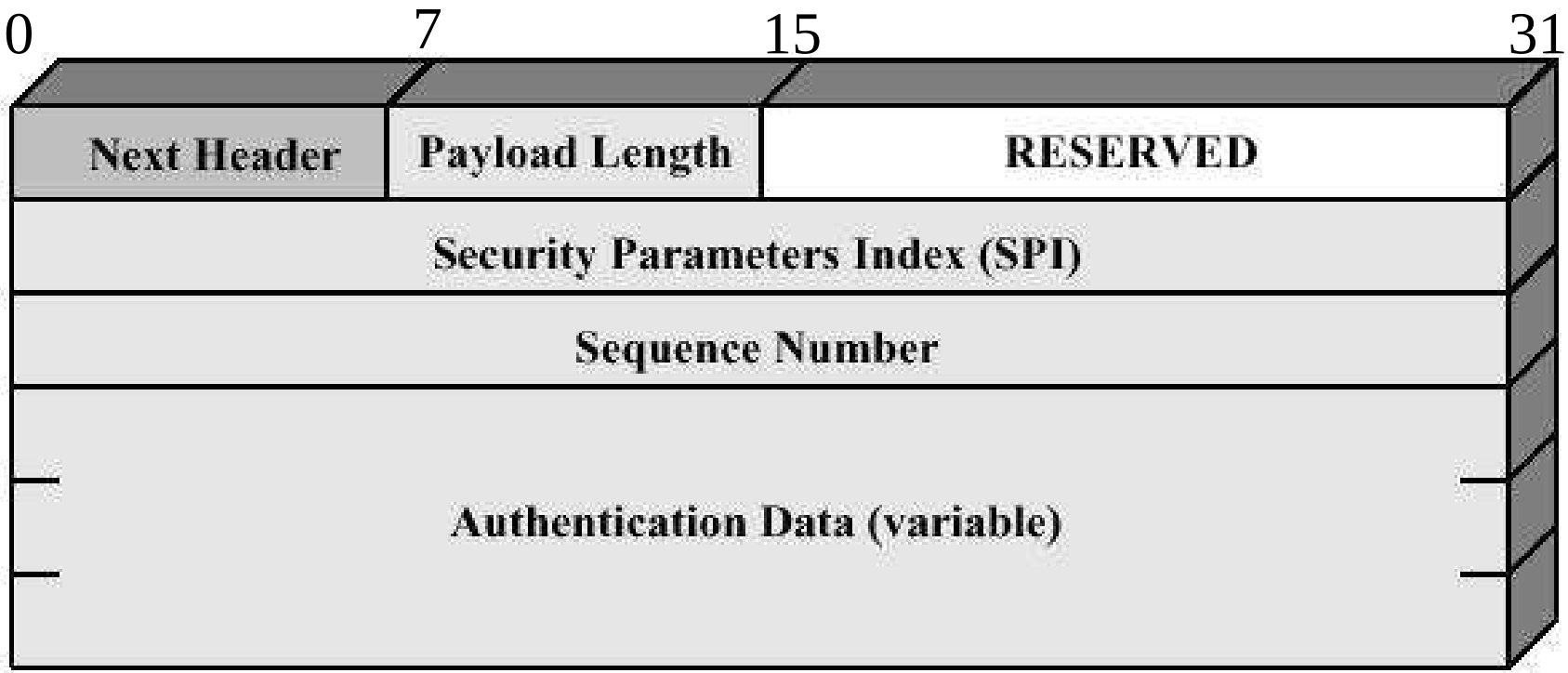

(c) Tunnel Mode

# Authentication Header (AH)

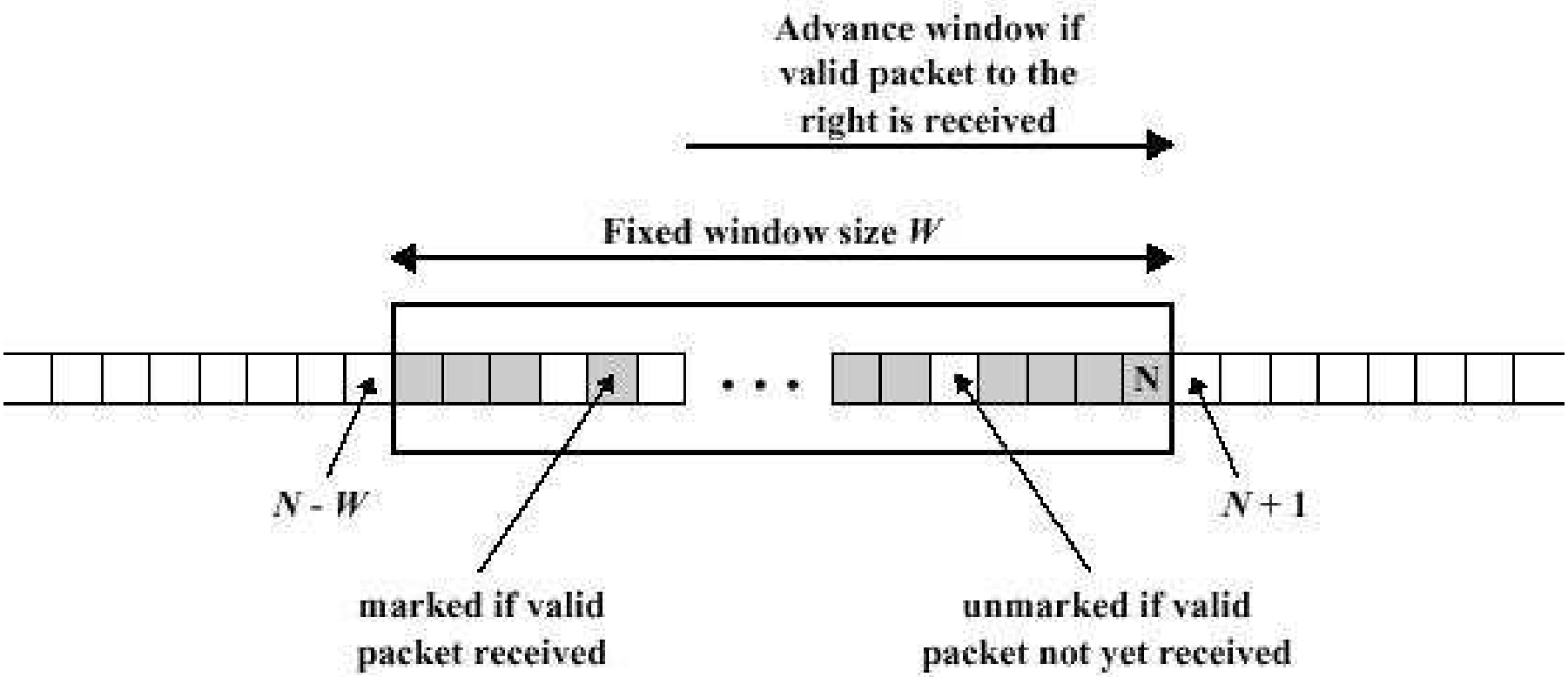
El Header AH contiene los siguientes campos:

- **Next Header**: El tipo de protocolo del paquete (8 bits).
- **Payload Length**: Longitud del AH. (8 bits)
- **Reserved**: 16 bits reservados para uso futuro. Hoy en día debe ser todos ceros.
- **SPI**: 32 bit "Security Parameter Index". Consideraciones:
  - Los valores entre 1 y 255 están reservados para uso futuro.
  - El SPI "0" esta reservado para uso local y no debe ser enviado a la red.
- **Sequence Number**: 32 bits utilizado para protección de replays. Esto tiene un efecto colateral, que limita la cantidad de paquetes a mil millones, una vez pasado ese limite, si la SA esta utilizando protección anti replay, deberá ser reestablecida.
- **Authentication Data**: una cadena de bits de longitud arbitraria para autenticación del paquete.

# Authentication Header (AH)



# Mecanismo anti-replay



# Encapsulating Security Payload (ESP)

Los paquetes ESP son similares a AH, pero en el protocolo 50 de IP.

ESP provee confidencialidad de datos y en las condiciones correctas también puede proveer confidencialidad del flujo de datos.

La posición del header ESP es la misma que la del header AH, dependiendo del modo en que se este utilizando la SA.

ESP puede ser utilizado para proveer la misma funcionalidad que AH utilizando el algoritmo de cifrado “NULL”

# Encapsulating Security Payload (ESP)

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Servicios de seguridad (uno, otro o los dos conjuntos):
  - Integridad
  - Autenticación del origen de los datos
  - Anti-replay (opcional)
    → Son los mismos que presenta AH
  - Confidencialidad
  - Resiste el análisis de tráfico (modo túnel)
    → Solamente disponibles en ESP.

- Emplea MAC:
  - HMAC-MD5
  - HMAC-SHA1

# Encapsulating Security Payload (ESP)

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

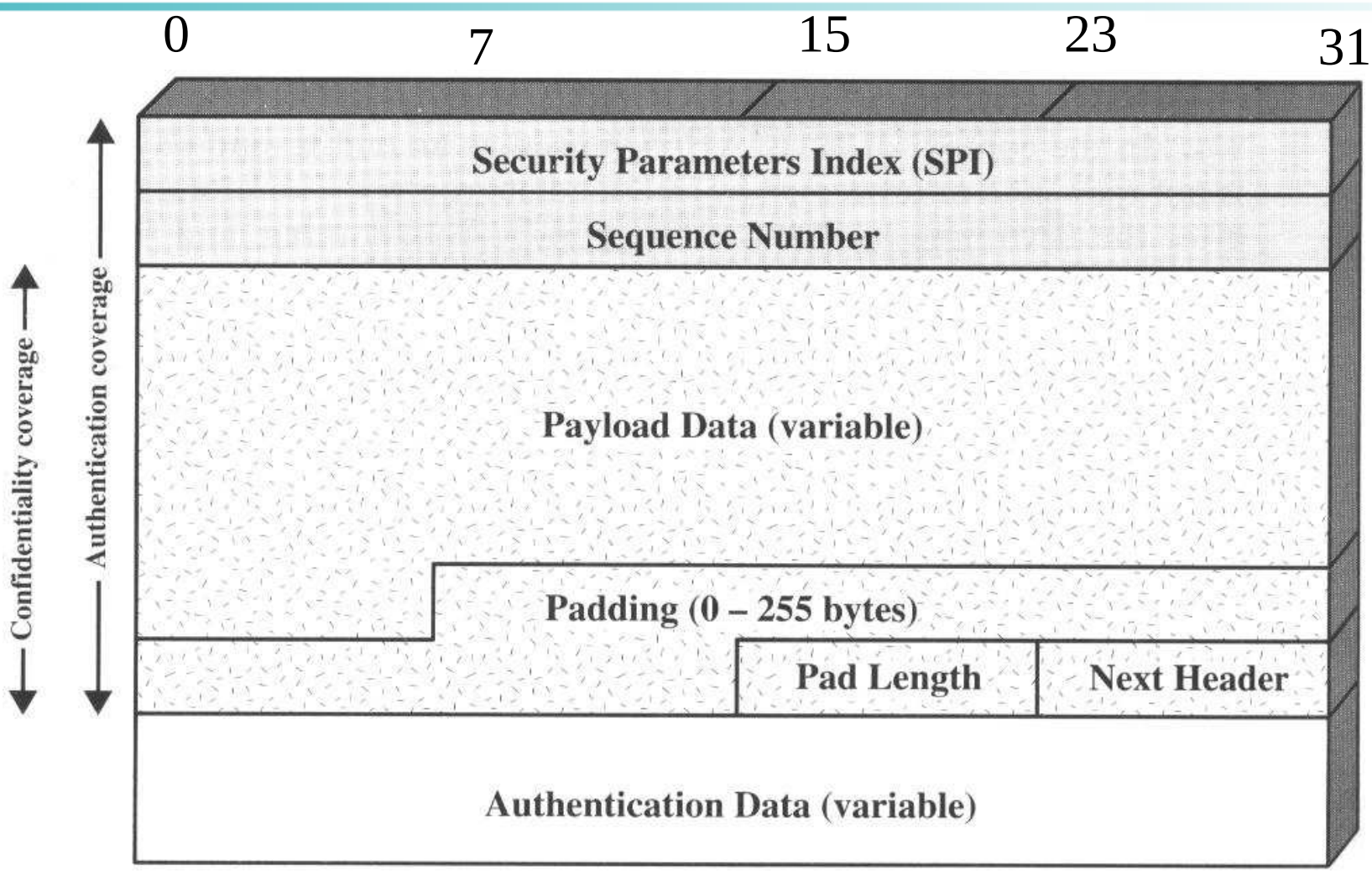
- Cifrado para confidencialidad:
  - Algoritmos obligatorios de cifrado:
    DES (RFC 2405) y NULL (RFC 2410)
  - Algoritmos opcionales de Cifrado (RFC 2451):
    3DES, CAST-128, RC5, IDEA, Blowfish y AES
  - Modo CBC

# Encapsulating Security Payload (ESP)

El header ESP contiene los siguientes campos:

- **SPI**: valor de 32 bits
- **Sequence Number**: 32 bits de protección anti-replay
- **Payload Data**: Longitud variable. Payload cifrado.
- **Padding**: 0-255 bytes. Utilizados para proteger al payload cifrado de análisis criptoanalítico.
- **Pad Length**: 8 bits, indican la longitud del padding.
- **Next Header**: Tipo de protocolo del paquete. 8 bits.
- **Authentication Data**: Un checksum del paquete.

# Encapsulating Security Payload (ESP)



# NAT traversal

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- NAT supone la **manipulación** y por ende **modificación** del paquete IP original para resolver el problema del direccionamiento IP (público/privado)

- IPSec considera la acción de **NAT** como un **ataque** a la integridad del paquete y la conexión no se realiza.

- Nat-Traversal: Solución propuesta definida en RFCs 3947 y 3948 en base a los requerimientos definidos en RFC 3475.

  Consiste en **encapsular** la trama cifrada por IPSec (antes de añadirle la cabecera IP final) con un campo **UDP** y sobre esta cabecera extra se realizan las operaciones de NAT.

# TLS / SSL

# Transport Layer Security (TLS)

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Versión actualizada de **SSL** (Secure Sockets Layer)
  - La última versión de SSL (Netscape) fue 3.0
  - TLS se identifica como SSL v 3.1
  - Similar, pero no compatible directamente.
  - Especificado en RFC 2246 (1999). Extendido posteriormente en RFC 3546 (2003)
- Protege una sesión entre **cliente** y **servidor**.
  - Típicamente, HTTP (navegador y web server).
- Requiere protocolo de transporte confiable.
  - Por ejemplo **TCP**.

# Transport Layer Security (TLS)

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

**TIPS:**

- RFC 2246
- Basado en SSL 3.0.
- Incompatible con este SSL 3.0
- Una de las ventajas que proporciona sobre SSL es que puede ser iniciado a partir de una conexión TCP ya existente, lo cual permite seguir trabajando con los mismos puertos que los protocolos no cifrados.
- SSL es un protocolo incompatible con TCP, lo cual significa que no podemos establecer una conexión de un cliente TCP a un servidor SSL ni al revés, y por lo que es necesario diferenciarlos utilizando distintos puertos.
- Con TLS puede establecerse la conexión normalmente a través de TCP, y luego activar sobre el mismo el protocolo TLS.

# Problemas con SSL v2



- **Utiliza la misma clave para integridad y confidencialidad.**
- **Utiliza una función MAC dónde el secreto es un prefijo y usa MD5, por ende es vulnerable a “length extension attacks”**
- **No protege el handshake, por ende es posible mediante un man-in-the-middle forzar la utilización de algoritmos de cifrado débiles.**
- **No tiene un mensaje de fin de conexión, por ende es posible terminar la conexión con un TCP FIN (IP spoofing).**

## Mecanismos de seguridad de SSLv3 y TLS:

- Numeración secuencial de los paquetes que contienen datos.
- Incorporación de esa numeración al cálculo de los MAC.
- Protección frente a ataques que intentan forzar el empleo de versiones antiguas del protocolo o cifrados más débiles.
- El mensaje que finaliza la fase de establecimiento de la conexión incorpora un hash de todos los datos intercambiados por ambos interlocutores.

## Mecanismos de seguridad adicionales de TLS:

- Se utilizan dos funciones de hash (MD5 y SHA-1) para calcular el MAC.
- La construcción de MAC está basada en el HMAC de RFC 2104.

# TLS



- **La versión actual es la TLS 1.2**

- **No usa md5 ni sha1, sino sha-256**
- **No puede negociar SSLv2**
- **Está soportado por todos los navegadores modernos**

- **Ver https://www.ssllabs.com/downloads/SSL_TLS_Deployment_Best_Practices.pdf**

# Servicios de Seguridad

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Autenticación
  - del servidor frente al cliente;
  - opcionalmente, del cliente frente al servidor.

  →Mediante **certificados** de clave pública.

- Integridad

  →Mediante **MAC** y números de secuencia.

- Confidencialidad
  - opcional

  →Mediante **cifrado** con algoritmo simétrico.

# Fases de SSL

Una comunicación a través de SSL implica tres fases:

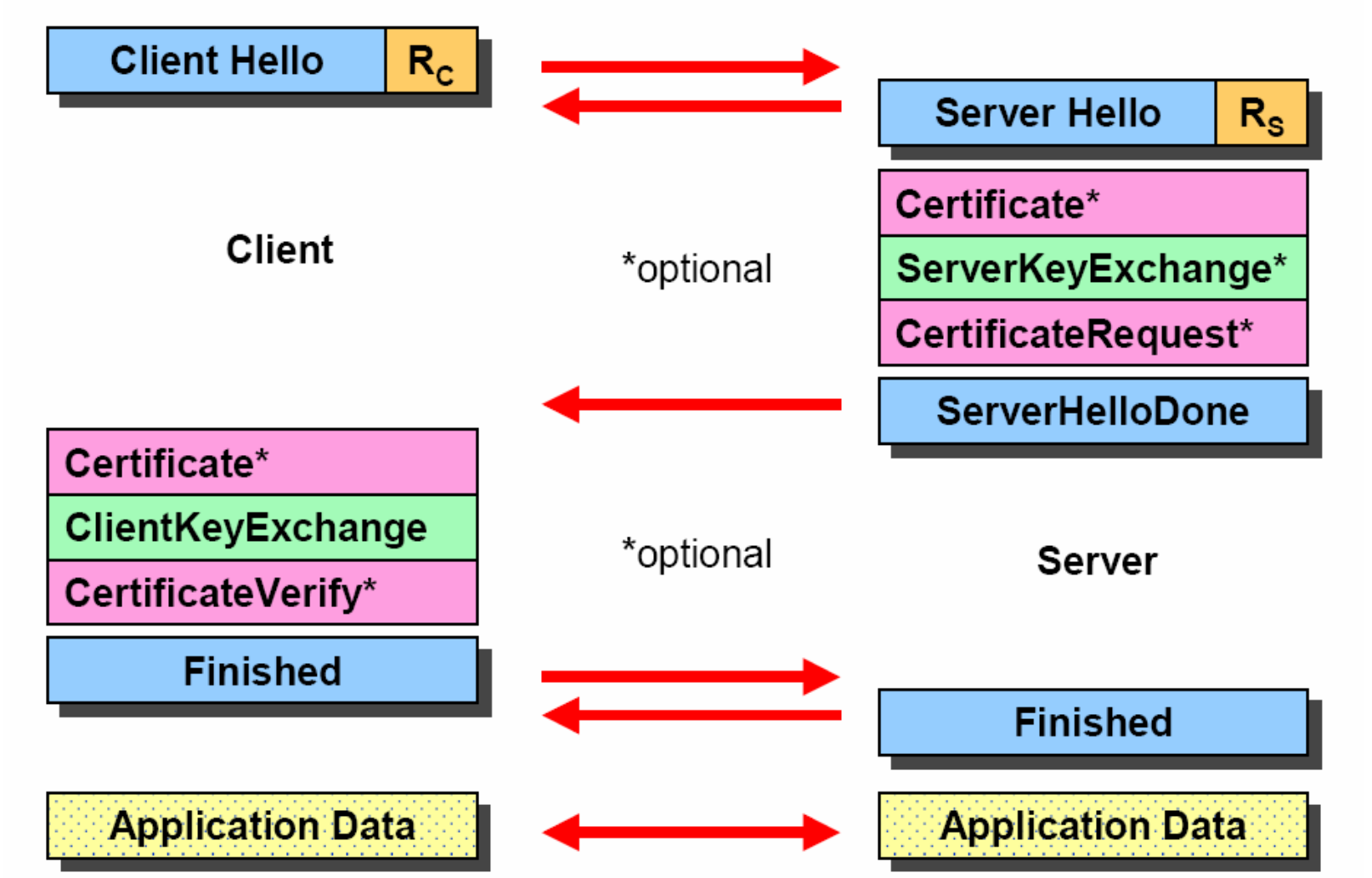
- Establecimiento de la conexión y negociación de los algoritmos criptográficos que van a usarse en la comunicación, a partir del conjunto de algoritmos soportados por cada uno de los interlocutores.
- Intercambio de claves, empleando algún mecanismo de clave pública y autentificación de los interlocutores a partir de sus certificados digitales.
- Cifrado simétrico del tráfico.

# Fases

- Handshake:
  - Negociación de **algoritmos** y parámetros.
  - **Autenticación** (del servidor o mutua).
  - Canal seguro para **compartir** un secreto inicial.
  - Derivación de **claves** en cada extremo.
  - **Integridad** de todo el intercambio.

- Transferencia de datos:
  - Usa las **claves** anteriormente derivadas.
  - Provee **integridad**.
  - Opcionalmente, provee **confidencialidad**.
  - Autentica el **cierre** de cada conexión.

# Handshake


Client Hello
$R_C$
Server Hello
$R_S$
Certificate*
ServerKeyExchange*
CertificateRequest*
Client
*optional
ServerHelloDone
Certificate*
ClientKeyExchange
CertificateVerify*
*optional
Server
Finished
Finished
Application Data
Application Data

# Handshake

El estado de la sesión SSL es controlado por el protocolo handshake de SSL. Se emplea para negociar los atributos de sesión.

Cuándo un cliente SSL y un servidor SSL comienzan a comunicarse, acuerdan:

- algoritmos criptográficos,
- versión del protocolo,
- opcionalmente se autentican
- con algoritmos de cifrado asimétrico generan claves de sesión.

# Handshake

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

1. El cliente comienza con un mensaje de **ClientHello** al cual el servidor debe responder con un mensaje de **ServerHello**. De no ser así, ocurre un error fatal y la conexión falla.
2. Se establecen los siguientes atributos: versión, ID de sesión, conjunto de cifradores y método de compresión.
3. Se generan dos valores al azar los cuales luego serán intercambiados:
   - ClientHello-Random RC y ServerHello-Random RS,
4. Opcionalmente, el servidor envía después su certificado de servidor (X.509).
5. Si no se envía ningún certificado, entonces un mensaje alternativo de **ServerKeyExchange** es enviado conteniendo un secreto "Diffie-Hellman (DH)".

# Handshake

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

6. Si el servidor requiere autenticación del lado del cliente, se añade un mensaje opcional de **CertificateRequest**.
7. El servidor indica el final de la fase de "Hello" enviando un mensaje de **ServerHelloDone**.
8. Si el servidor ha enviado un mensaje de **CertificateRequest**, el cliente debe enviar su certificado de cliente X.509 o bien debe enviar una alarma de "no certificate".
9. Si el cliente ha recibido un certificado del servidor que contenía la llave pública RSA, el cliente cifra un secreto aleatoriamente elegido y lo envía al servidor en un mensaje **ClientKeyExchange**.

# Handshake

10. El cliente puede enviar alternativamente, su clave DH.
11. Cada lado puede ahora formar una clave de sesión.
12. El cliente entonces emite un mensaje de **ChangeCipherSpec** que anuncia que los nuevos parámetros han sido cargados.
13. Luego el cliente emite un mensaje de **Finished** con los nuevos parámetros cifrados.
14. El servidor repite esto ultimo en su lado.
15. La comunicación a partir de ahora es cifrada.

# Algunos algoritmos disponibles

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

| Cipher Suite | Auth | Key Exchange | Encryption | Digest | Number |
|---|---|---|---|---|---|
| TLS_RSA_WITH_NULL_MD5 | RSA | RSA | NULL | MD5 | 0x0001 |
| TLS_RSA_WITH_NULL_SHA | RSA | RSA | NULL | SHA | 0x0002 |
| TLS_RSA_EXPORT_WITH_RC4_40_MD5 | RSA | RSA_EXPORT | RC4_40 | MD5 | 0x0003 |
| TLS_RSA_WITH_RC4_128_MD5 | RSA | RSA | RC4_128 | MD5 | 0x0004 |
| TLS_RSA_WITH_RC4_128_SHA | RSA | RSA | RC4_128 | SHA | 0x0005 |
| TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5 | RSA | RSA_EXPORT | RC2_40_CBC | MD5 | 0x0006 |
| TLS_RSA_WITH_IDEA_CBC_SHA | RSA | RSA | IDEA_CBC | SHA | 0x0007 |
| TLS_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | RSA_EXPORT | DES40_CBC | SHA | 0x0008 |
| TLS_RSA_WITH_DES_CBC_SHA | RSA | RSA | DES_CBC | SHA | 0x0009 |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA | RSA | RSA | 3DES_EDE_CBC | SHA | 0x000A |
| TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA | RSA | DH_DSS_EXPORT | DES_40_CBC | SHA | 0x000B |
| TLS_DH_DSS_WITH_DES_CBC_SHA | DSS | DH | DES_CBC | SHA | 0x000C |
| TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA | DSS | DH | 3DES_EDE_CBC | SHA | 0x000D |
| TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | DH_EXPORT | DES_40_CBC | SHA | 0x000E |
| TLS_DH_RSA_WITH_DES_CBC_SHA | RSA | DH | DES_CBC | SHA | 0x000F |
| TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA | RSA | DH | 3DES_EDE_CBC | SHA | 0x0010 |

# Algunos algoritmos Disponibles

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

| | | | | | |
|---|---|---|---|---|---|
| TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA | DSS | DHE_EXPORT | DES_40_CBC | SHA | 0x0011 |
| TLS_DHE_DSS_WITH_DES_CBC_SHA | DSS | DHE | DES_CBC | SHA | 0x0012 |
| TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA | DSS | DHE | 3DES_EDE_CBC | SHA | 0x0013 |
| TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA | DHE_EXPORT | DES_40_CBC | SHA | 0x0014 |
| TLS_DHE_RSA_WITH_DES_CBC_SHA | RSA | DHE | DES_CBC | SHA | 0x0015 |
| TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA | RSA | DHE | 3DES_EDE_CBC | SHA | 0x0016 |
| TLS_DH_anon_EXPORT_WITH_RC4_40_MD5 | - | DH_EXPORT | RC4_40 | MD5 | 0x0017 |
| TLS_DH_anon_WITH_RC4_128_MD5 | - | DH | RC4_128 | MD5 | 0x0018 |
| TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA | - | DH | DES_40_CBC | SHA | 0x0019 |
| TLS_DH_anon_WITH_DES_CBC_SHA | - | DH | DES_CBC | SHA | 0x001A |
| TLS_DH_anon_WITH_3DES_EDE_CBC_SHA | - | DH | 3DES_EDE_CBC | SHA | 0x001B |
| TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA † | RSA | RSA | DES_CBC | SHA | 0x0062 |
| TLS_DHE_DSS_EXPORT1024_WITH_DES_CBC_SHA † | RSA | RSA | DES_CBC | SHA | 0x0063 |
| TLS_RSA_EXPORT1024_WITH_RC4_56_SHA † | RSA | RSA | RC4_56 | SHA | 0x0064 |
| TLS_DHE_DSS_EXPORT1024_WITH_RC4_56_SHA † | RSA | RSA | RC4_56 | SHA | 0x0065 |
| TLS_DHE_DSS_WITH_RC4_128_SHA † | RSA | RSA | RC4_56 | SHA | 0x0066 |

Ver: https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml

# PFS

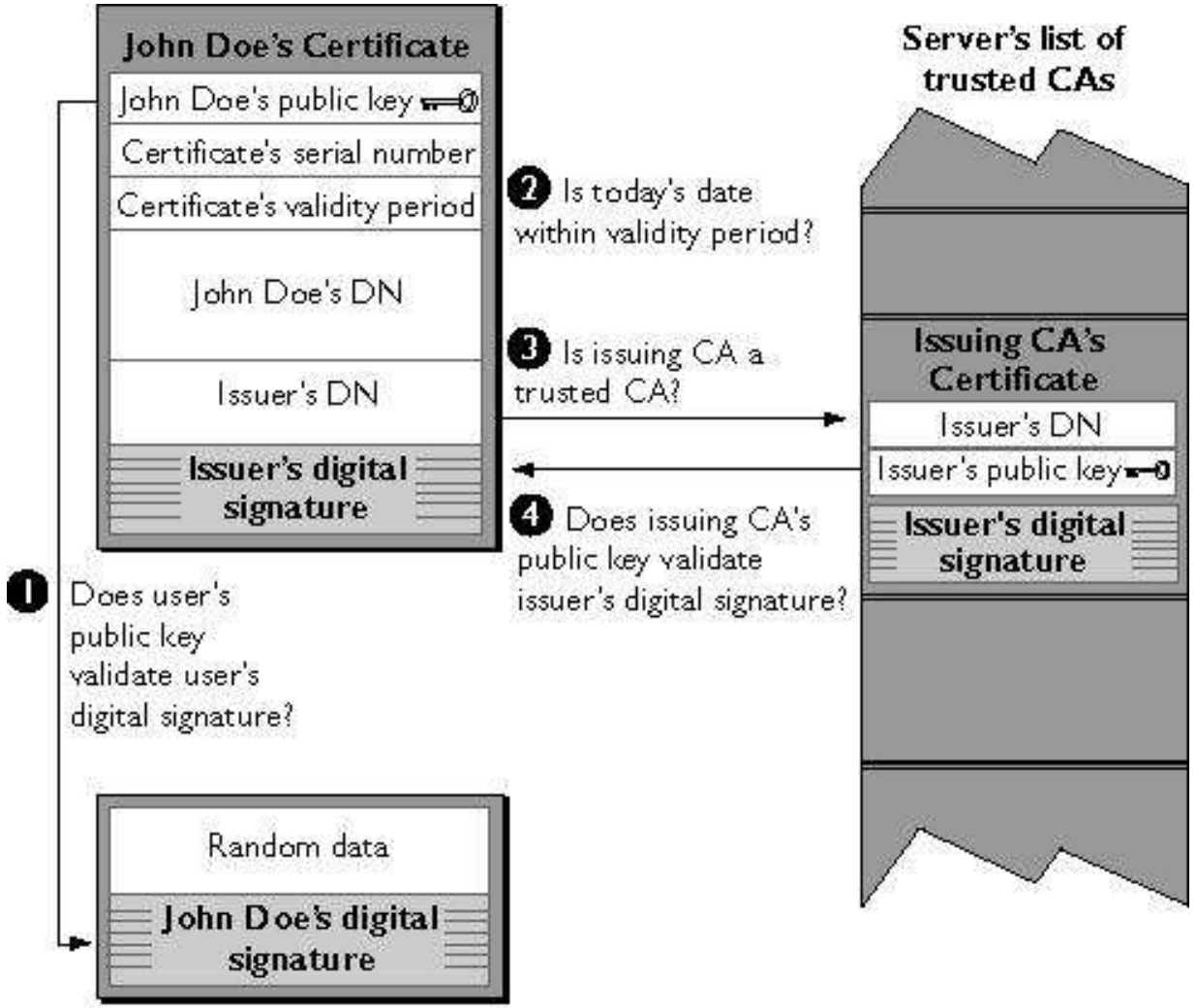
DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- **Perfect Forward Secrecy**
- **RFC 4251: PFS is essentially defined as the cryptographic property of a key-establishment protocol in which the compromise of a session key or long-term private key after a given session does not cause the compromise of any earlier session**
- **En SSL: https://community.qualys.com/blogs/securitylabs/2013/06/25/ssl-labs-deploying-forward-secrecy**

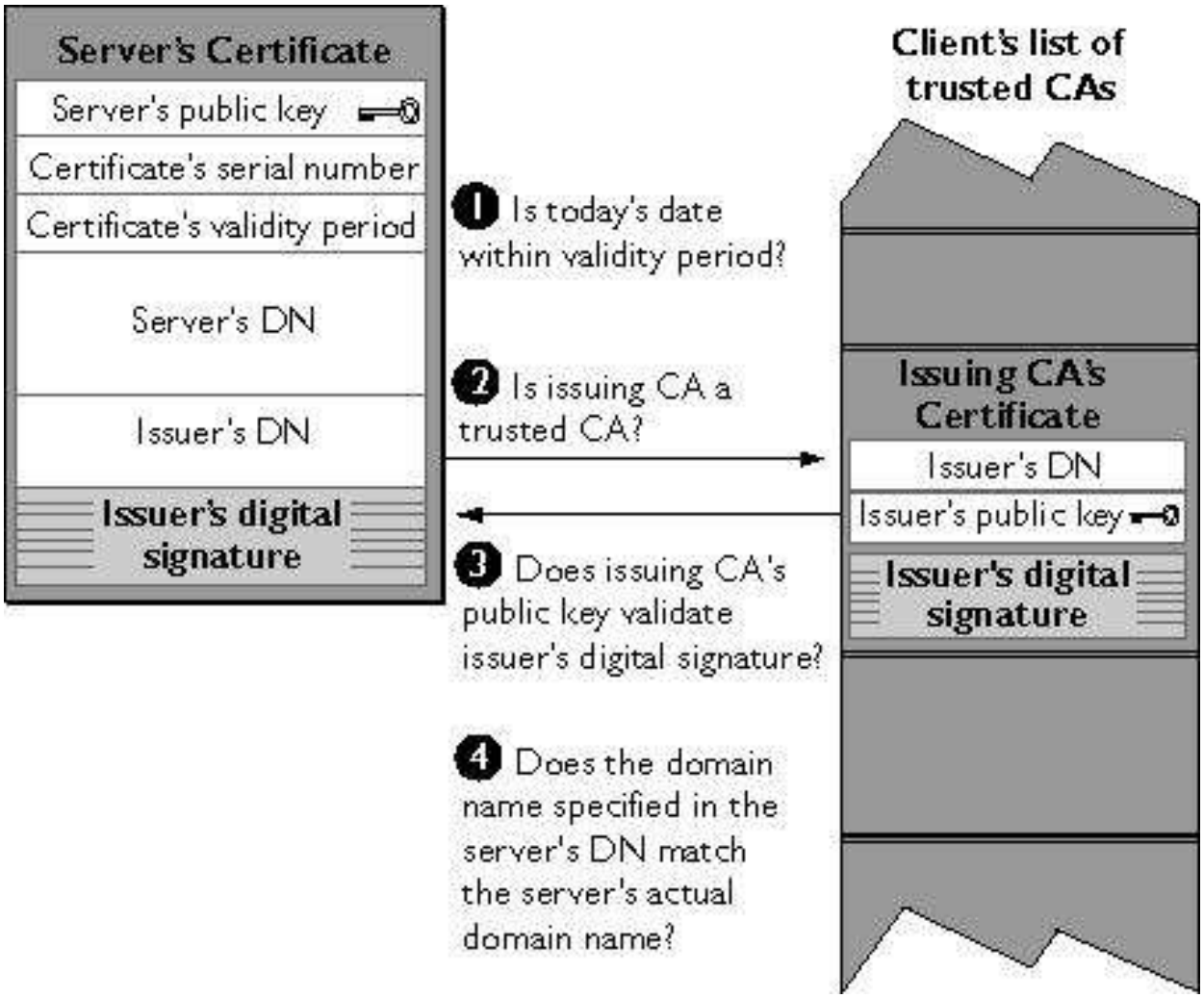
# Autenticación del cliente

- Con el propósito de controlar el acceso, el servidor puede requerir la **autenticación** del **cliente**, solicitándole su certificado de clave pública y verificando que posea la clave privada:
  - Después de presentar su certificado, el servidor se lo solicita al cliente mediante *CertificateRequest.*
  - El cliente lo envía, mediante *Certificate.*
  - Como comprobación de la posesión de su clave privada **KRA**, el cliente envía el mensaje *Certificate Verify*, que consiste en la **firma** de los mensajes de *handshake* intercambiados hasta ese momento.

# Autenticación del cliente



# Verif. Certificado Servidor



# OpenVPN

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

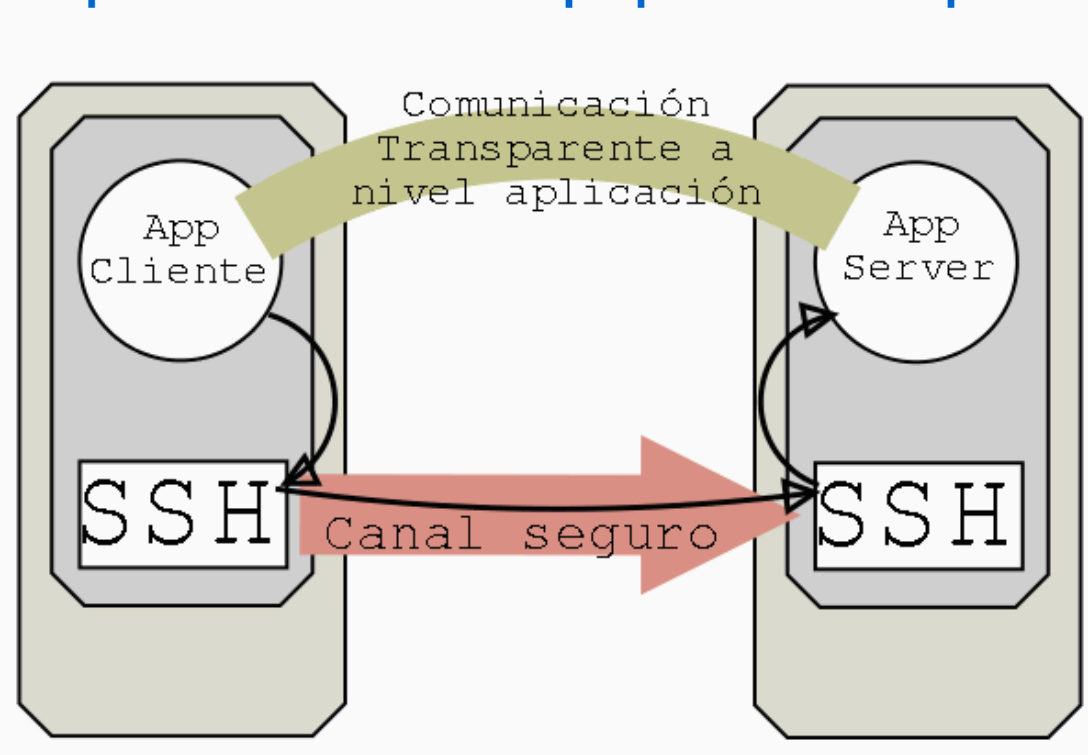
- **Busca implementar VPNs de forma mas portable y sencilla que IPSEC.**
- **No requiere de la complejidad del protocolo IKE.**
- **Puede trabajar en modo bridging o en modo routing**
- **Utiliza la implementación de openssl**
- **www.openvpn.net**

# SSH

# Túneles SSH – Port forwarding

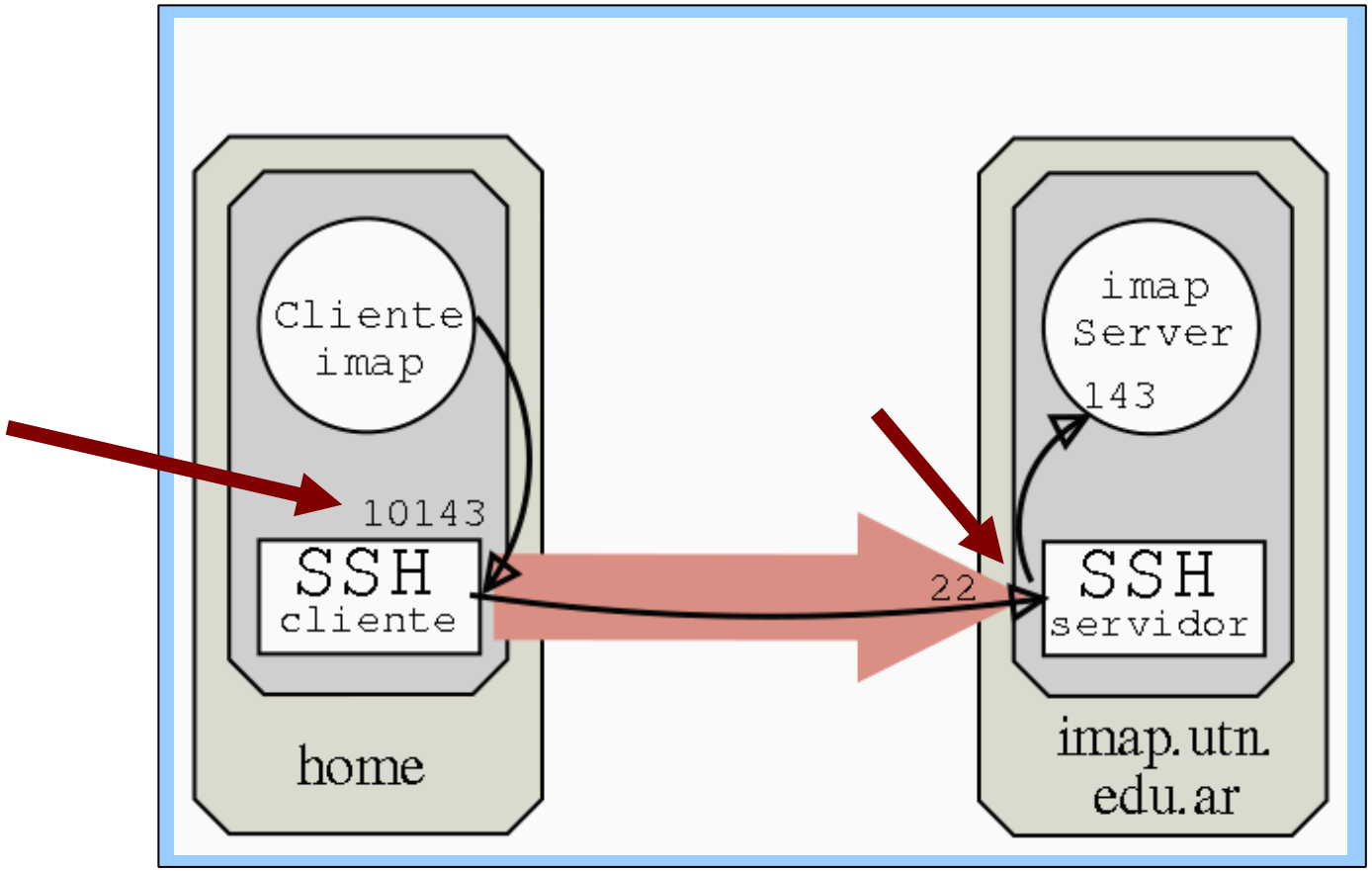
- Ahorrarse el software necesario para una complicada VPN
- Proteger mi clave plana de FTP Telnet, POP3, IMAP, authSMTP, NNTP.....
- Atravesar un firewall donde solo el servicio de SSH está permitido
- Acceder a servicios TCP internos de una LAN con direcciones privadas

**Los túneles SSH no permiten forwardear paquetes UDP o protocolos no IP**



# Túneles SSH – Local simple

**home:~$> ssh -L 10143:localhost:143 imap.utn.edu.ar**



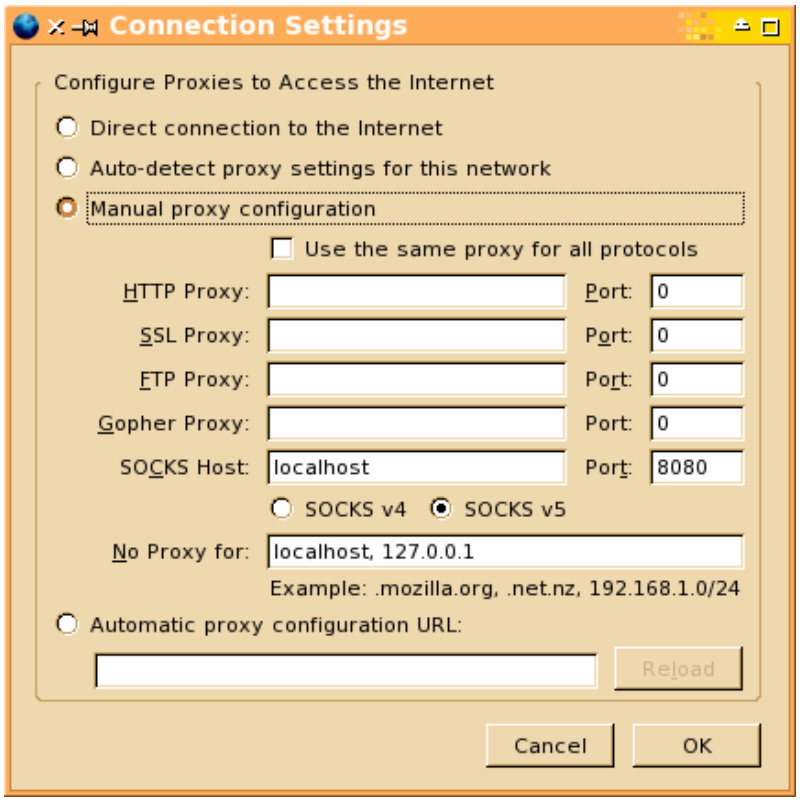
# Túneles con SSH – SOCKS (dynamic forwarding)



- SOCKet Secure es una forma de conectarse a una red, un proxy server
- El modificador *-D* genera un túnel local “dinámico”, haciendo que el cliente SSH se transforme en un SOCKS server

- Para emular un SOCKS server en el puerto 8080:

```
$> ssh -D8080 <user>@server
```




- La aplicación a forwardear debe ser SOCKS aware, o hay que usar un “proxifier”:
- proxychains.sourceforge.net
- ksb.sourceforge.net

# OpenSSH's built in tunneling



- **OpenSSH has built-in TUN/TAP support using -w<local-tun-number>:<remote-tun-number>. Here, a layer 3/point-to-point/ TUN tunnel is described. It is also possible to create a layer 2/ethernet/TAP tunnel.**

- **Soportado desde versión 4.3**

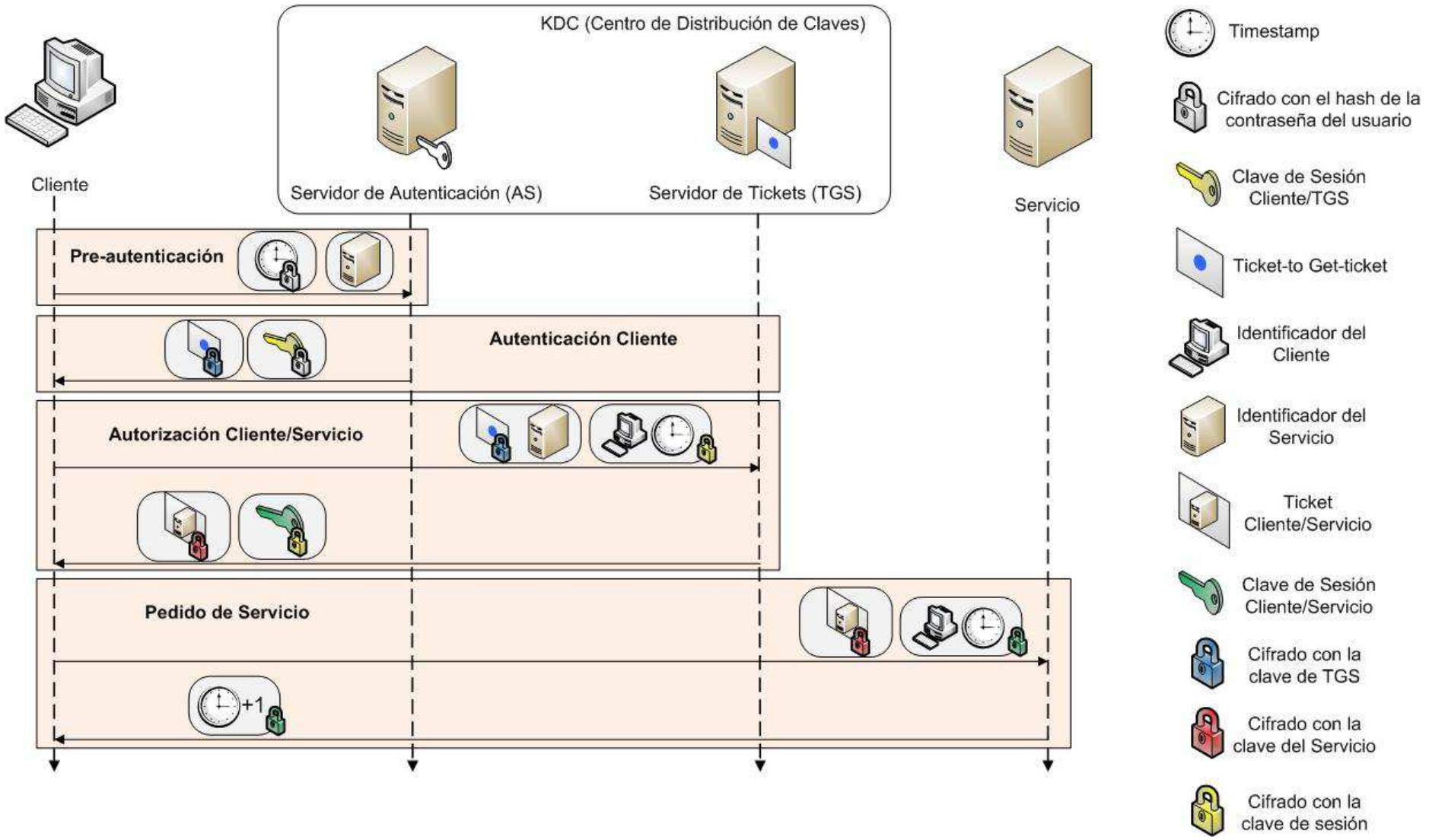
# Kerberos

# Kerberos

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Protocolo diseñado para proveer autenticación sobre un canal inseguro, desarrollado por el MIT.
- Utiliza criptografía simétrica y requiere de un tercero confiable (con quien todas las partes involucradas comparten un secreto).
- Provee autenticación mutua, y los mensajes empleados por el protocolo se encuentran protegidos contra ataques de replay y de eavesdropping.
- La versión 4 se publicó a finales de los ’80, la versión actual (v5) se estandarizó en 1993 (RFC 1510) y se modificó en 2005 (RFC 4120).
- Windows 2000 y superior utilizan Kerberos  v5 como su mecanismo de autenticación por defecto.

# Kerberos: Diagrama



# Kerberos: ¿Qué contienen los tickets?

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- ***Ticket-to Get-Ticket*:** Identificador del cliente, dirección de red del cliente, timestamp, período de validez del ticket, clave de sesión C*liente/TGS.*

- ***Client-to-service ticket:*** Identificador del cliente, identificador del servicio, dirección de red del cliente, timestamp, período de validez del ticket, clave de sesión *Cliente/Servicio.*

# Kerberos: Cracking de Contraseñas Off-line

DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

- Durante el proceso de pre-autenticación, el cliente envía un timestamp cifrado utilizando el hash de su contraseña como clave (dependiendo de la implementación la clave de cifrado puede ser distinta, pero siempre se deriva de la contraseña de usuario mediante algún algoritmo de derivación de clave conocido).

- Para hacer un ataque de cracking de contraseñas, probamos descifrar el timestamp con cada clave posible (derivadas de cada contraseña que deseamos probar) y verificamos su estructura. Si cumple con el formato de un timestamp (***YYYYMMDDHHMMSSZ)***, encontramos la contraseña del usuario.

- Referencia:
http://www.frankodwyer.com/papers/feasibility_of_w2k_kerberos_attack.htm