Securing Desktop Access to Host Systems – Protecting Critical Data

Eric Raisters

Security Technical Lead

WRQ, Inc.

ericr@wrq.com

HP WORLD 2002
Conference & Expo

What’s the problem?

- 70 - 80% of security breaches come from inside the firewall. (FBI and CSI surveys as recent as 2000)
- New regulations (e.g. HIPAA) require confidential data to be transmitted securely.
- Popularity of the Internet increases risk.
  - B2B/Web services
  - Working from home

Assumptions

- Securing the desktop is a whole topic on it’s own.
- Both desktop and host systems have a base level of security.
- The network between them is not secure.
- This is geared towards desktops running Windows, but could equally apply to other desktops.

Proprietary Products - Pluses

- More difficult to crack if don’t know algorithm; have to reverse-engineer
- Usually better administration utilities, services, and documentation

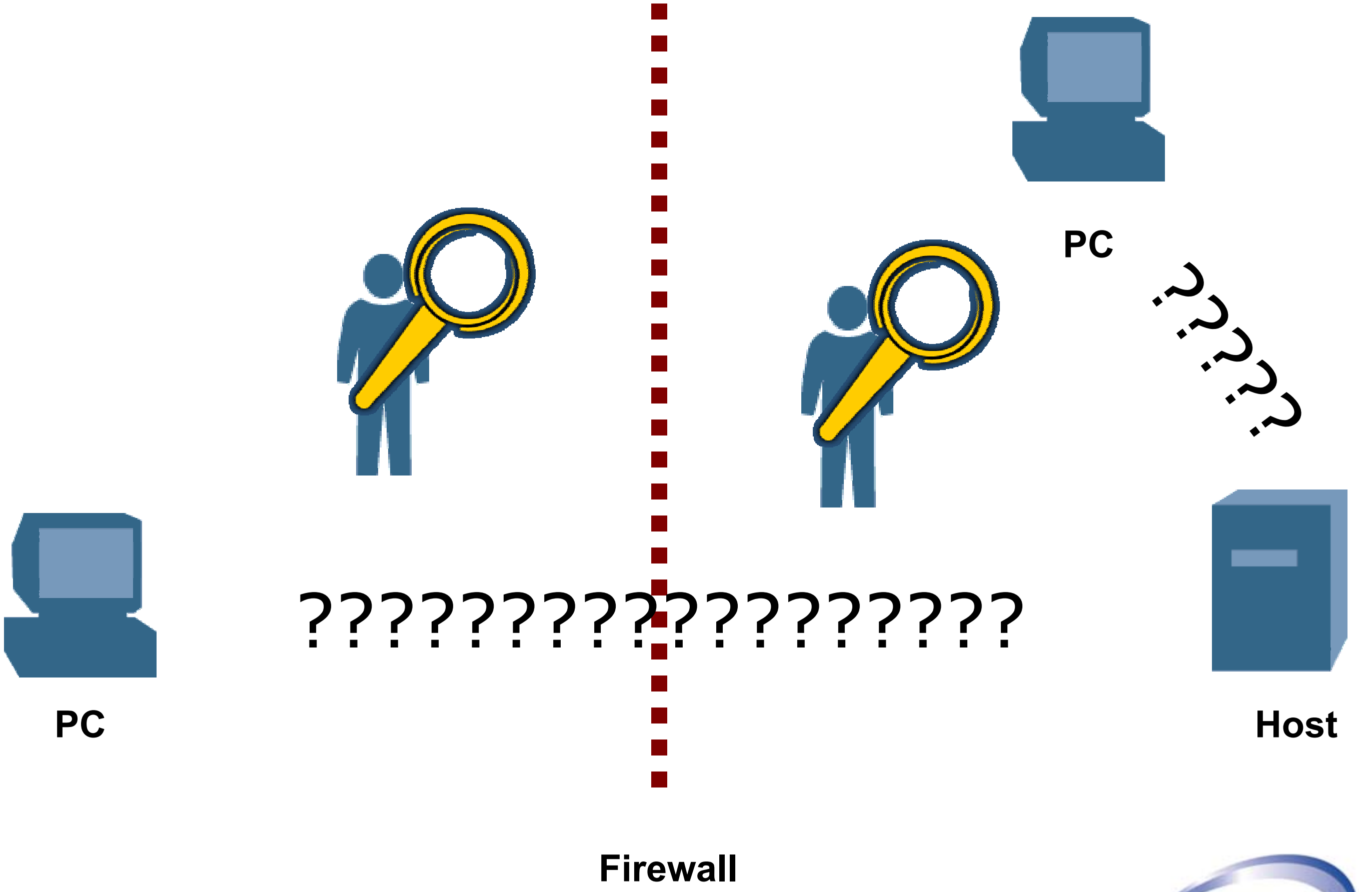
Open Source/Standard - Pluses

- Lots of eyes and hands working on it
- Not dependent on the trustworthiness of several programmers or one company
- Algorithms proven to be cryptologically sound

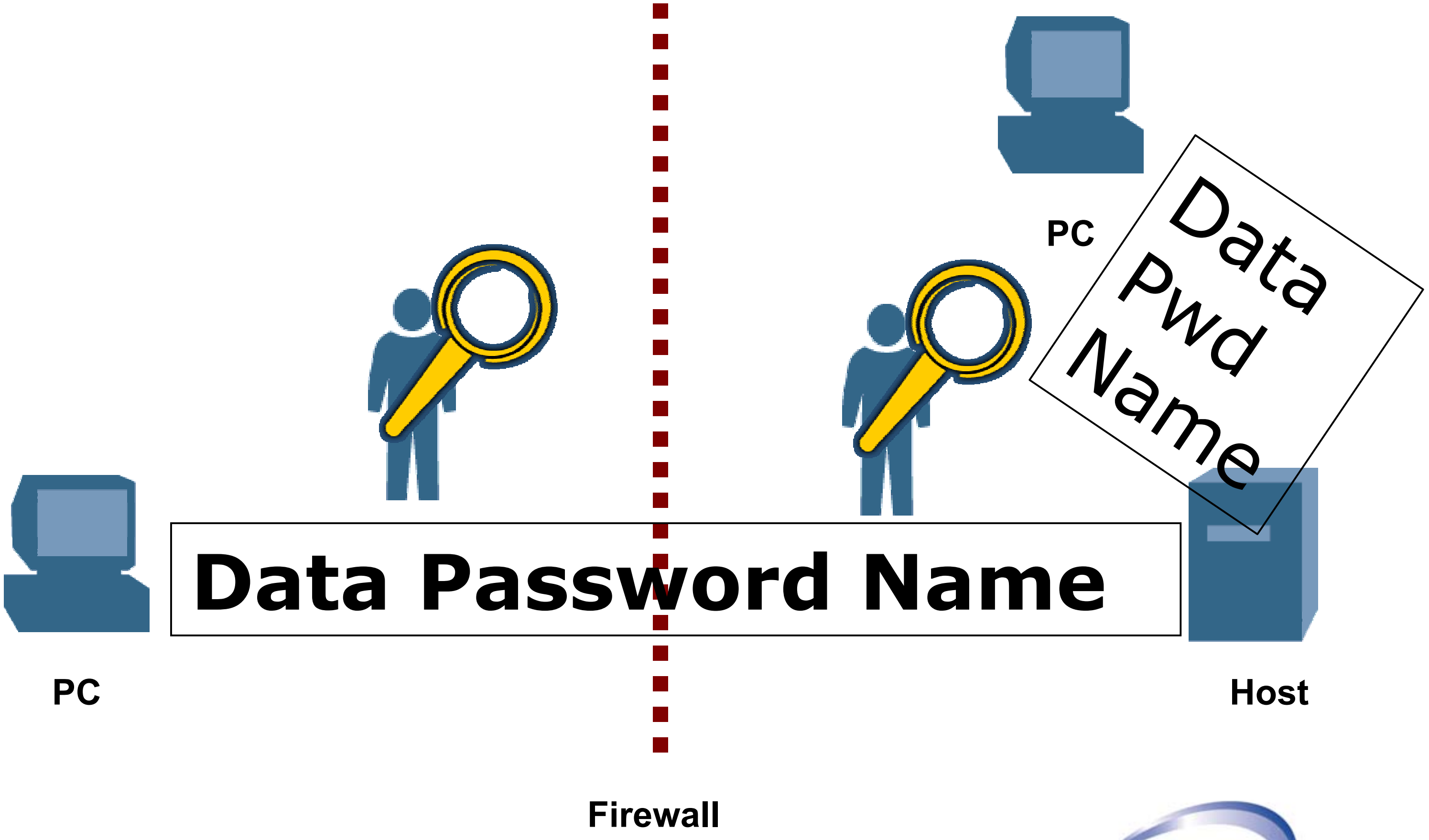
The Three A’s of Network Security

- Authentication
  - – proving who you are
  - – getting proof back (mutual authentication)
- Authorization
  - – proving what resources you may use
- Audit
  - – logging who has done what
  - – primarily a server-side responsibility
- [Administration]

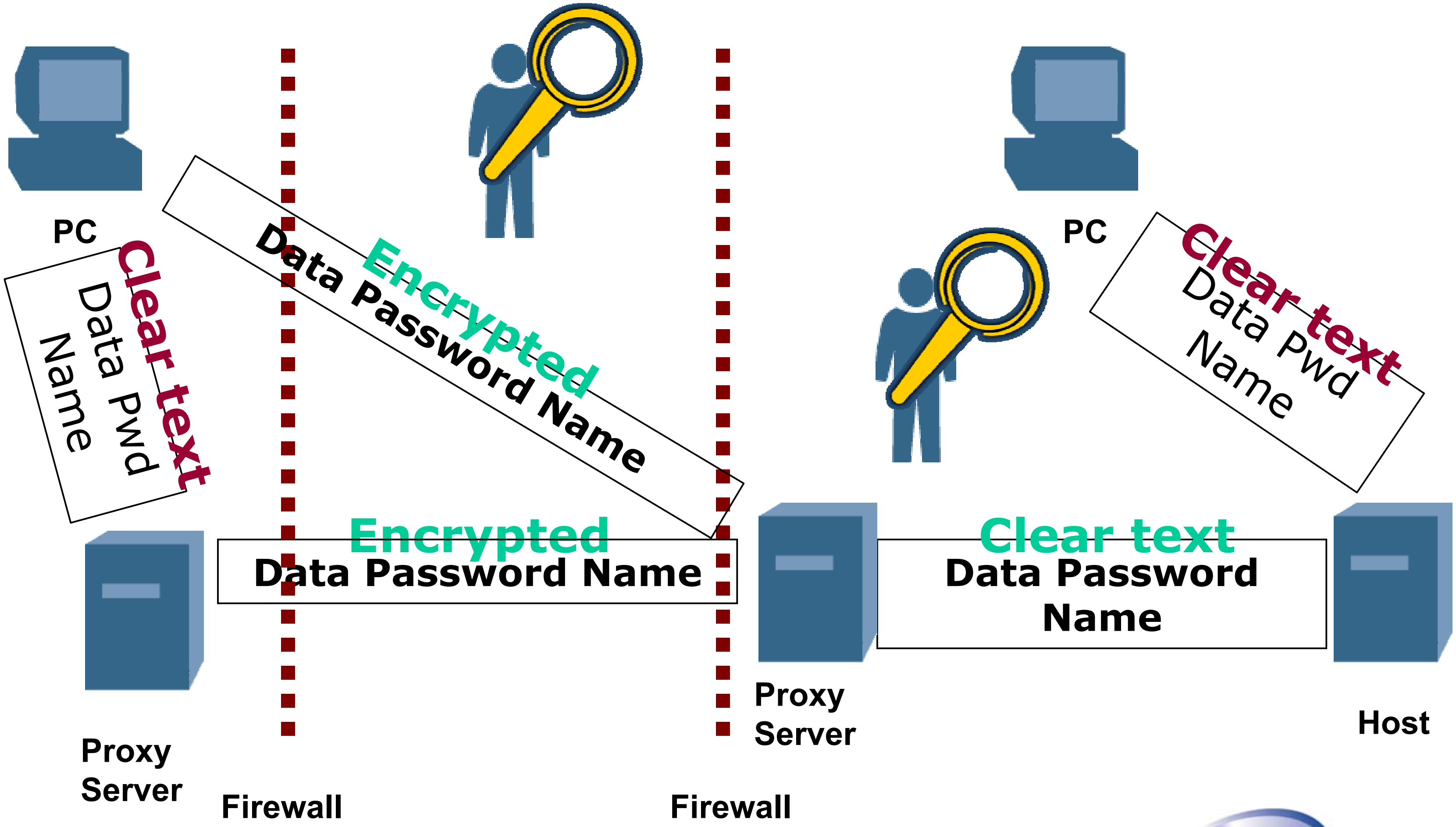
How Do We Securely Communicate?



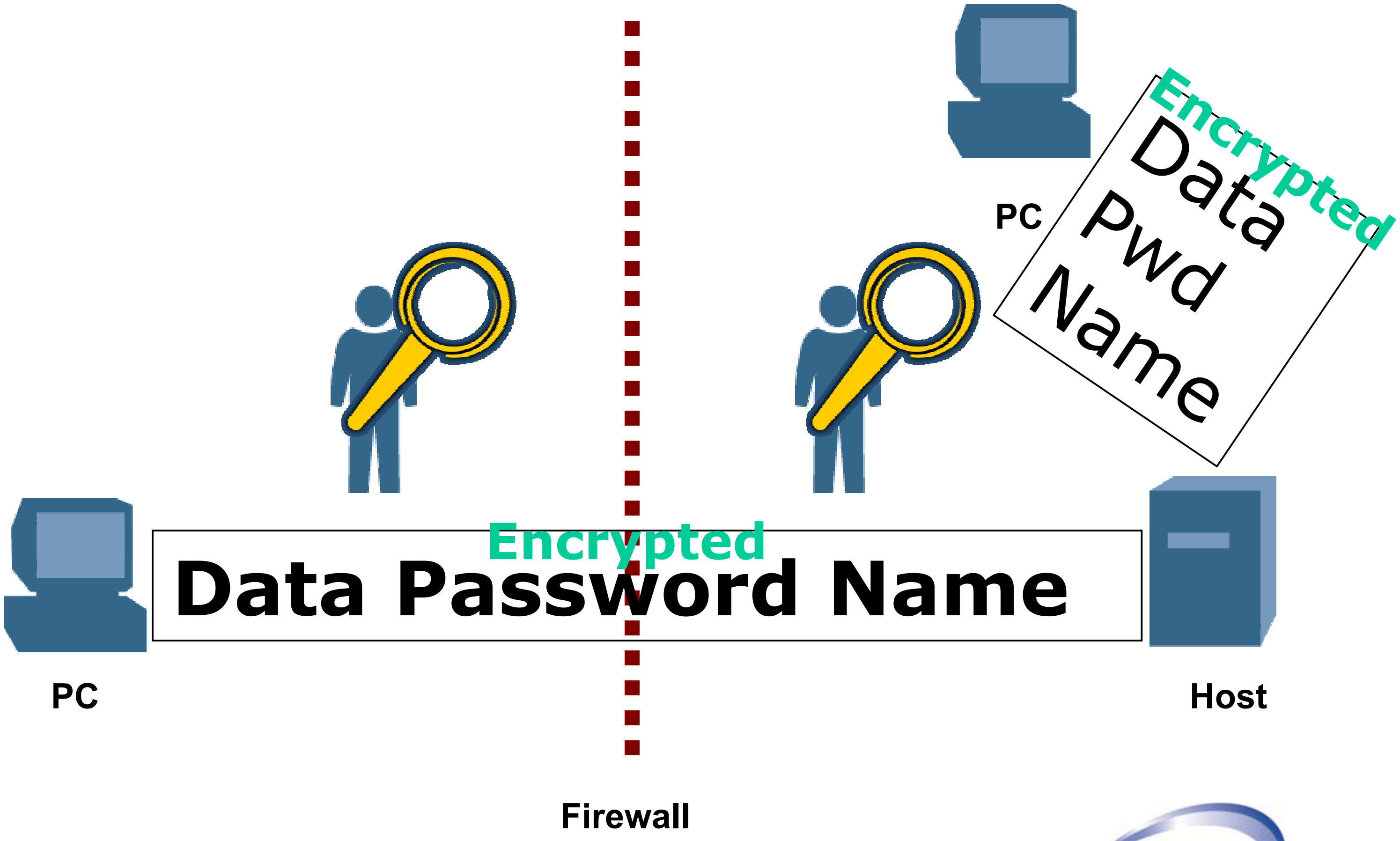
Historically - Clear Text
"Security Through Obscurity"



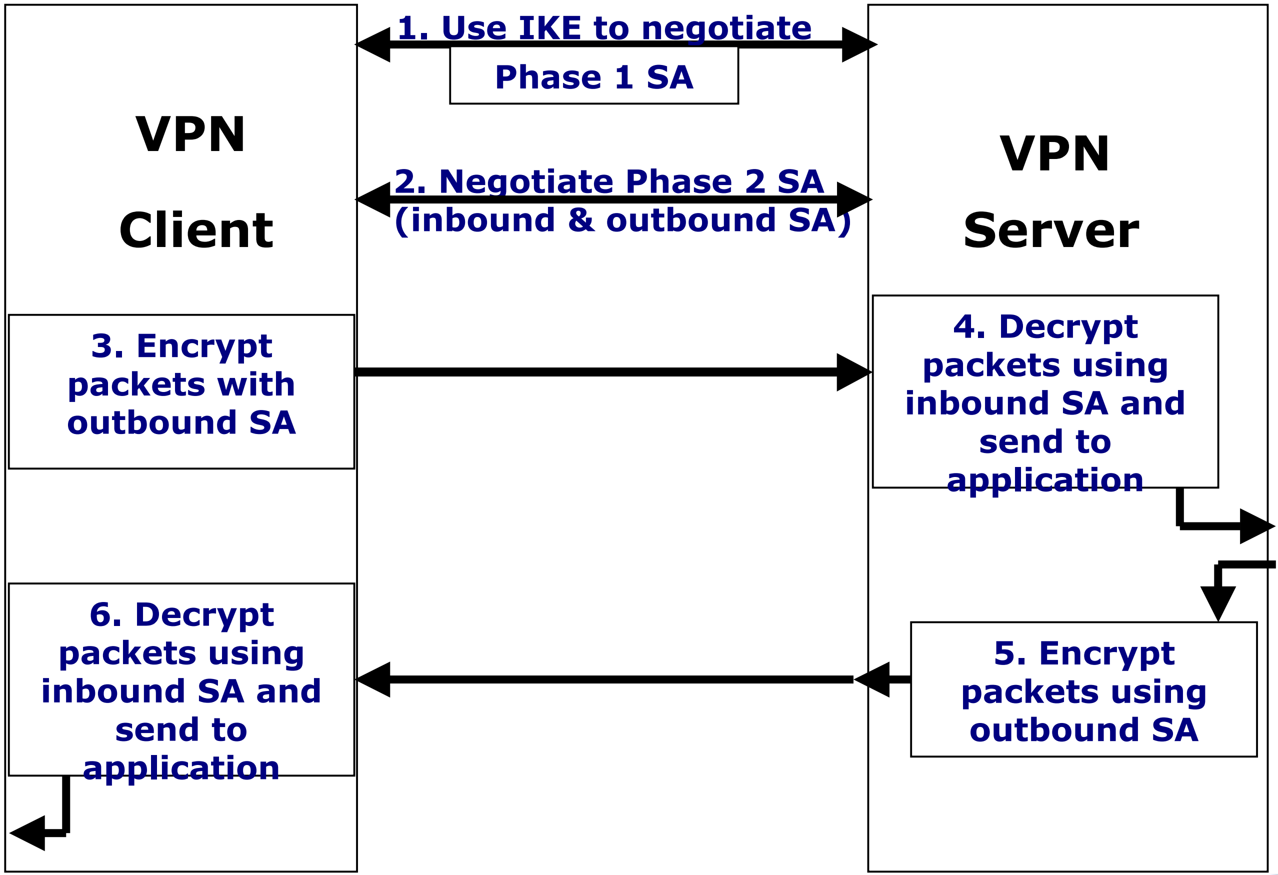
Middleware Model (Proxy)



Encrypted Tunnels Model (Direct)



VPN (via IPSec) Basics



VPN Client
VPN Server
1. Use IKE to negotiate
Phase 1 SA
2. Negotiate Phase 2 SA (inbound & outbound SA)
3. Encrypt packets with outbound SA
4. Decrypt packets using inbound SA and send to application
5. Encrypt packets using outbound SA
6. Decrypt packets using inbound SA and send to application

SOCKS Clients - Features

- Standard protocol developed by NEC
  - Application client makes a request to SOCKS to communicate with the application server.
  - On behalf of the application client, SOCKS establishes a proxy circuit to the application server, then relays the application data between the client and the server.
- Designed for traversing TCP-based client/server applications
- Version 5 provides secure authentication and encryption with GSSAPI

SOCKS Client - Pluses

- Standards-based protocol
- Facilitates firewall traversal
- Widely available in client programs
- Can be used regardless of the protocol the application uses
- Imposes little overhead on network communications

SOCKS Client - Minuses

- May require *identd* running on client PC (requires the use of DNS server or relay)
- Requires that applications be modified to become “SOCKSified”
- Additional server to administer that may be separate from network servers

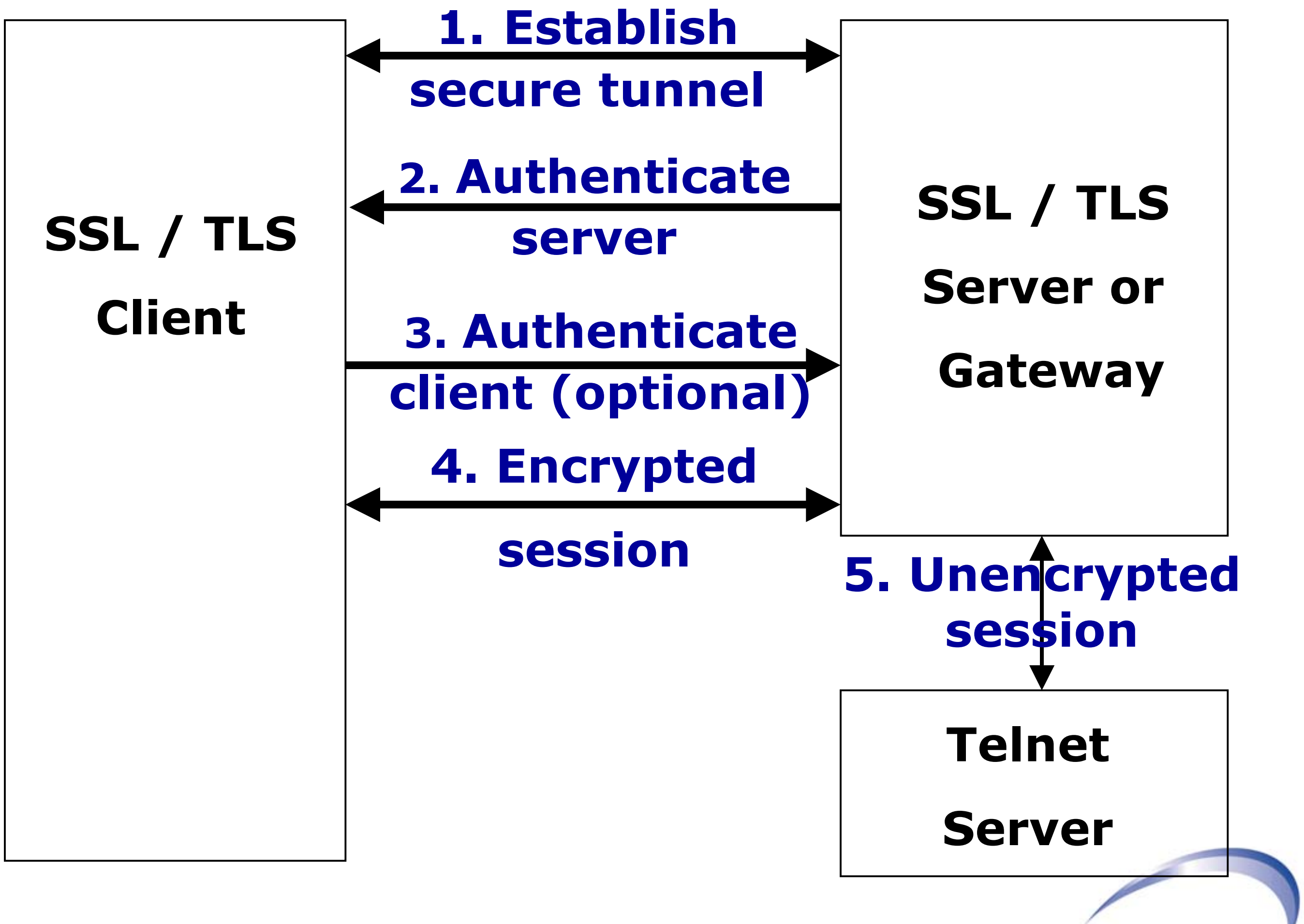
SSL / TLS / OpenSSL

- SSL (Secure Sockets Layer) v3.x
  - proprietary protocol originally developed by Netscape for Web (HTTP) security
  - the de facto security standard for the Web
- TLS (Transport Layer Security) v1.0
  - standards-based version that uses open-source algorithms
  - currently an IETF draft

SSL / TLS - Features

- Uses public key cryptography and X.509 certificates to authenticate
- Negotiates session keys for symmetric encryption
- Includes 56-bit DES, 128-bit RC-4, 168-bit 3DES encryption
- Provides data integrity and encryption

SSL / TLS Basics



SSL / TLS - Pluses

- Proven technology for securing the Web
- IETF standard coming (IBM is pushing)
- OpenSSL available for UNIX/Linux

SSL / TLS - Minuses

- Open standard may not interoperate with proprietary SSL — different key negotiation and encryption algorithms
- Certificates difficult to administrate
- Not many choices for Telnet or FTP server vendors (primarily IBM big iron)
- Possible trademark and royalty issues

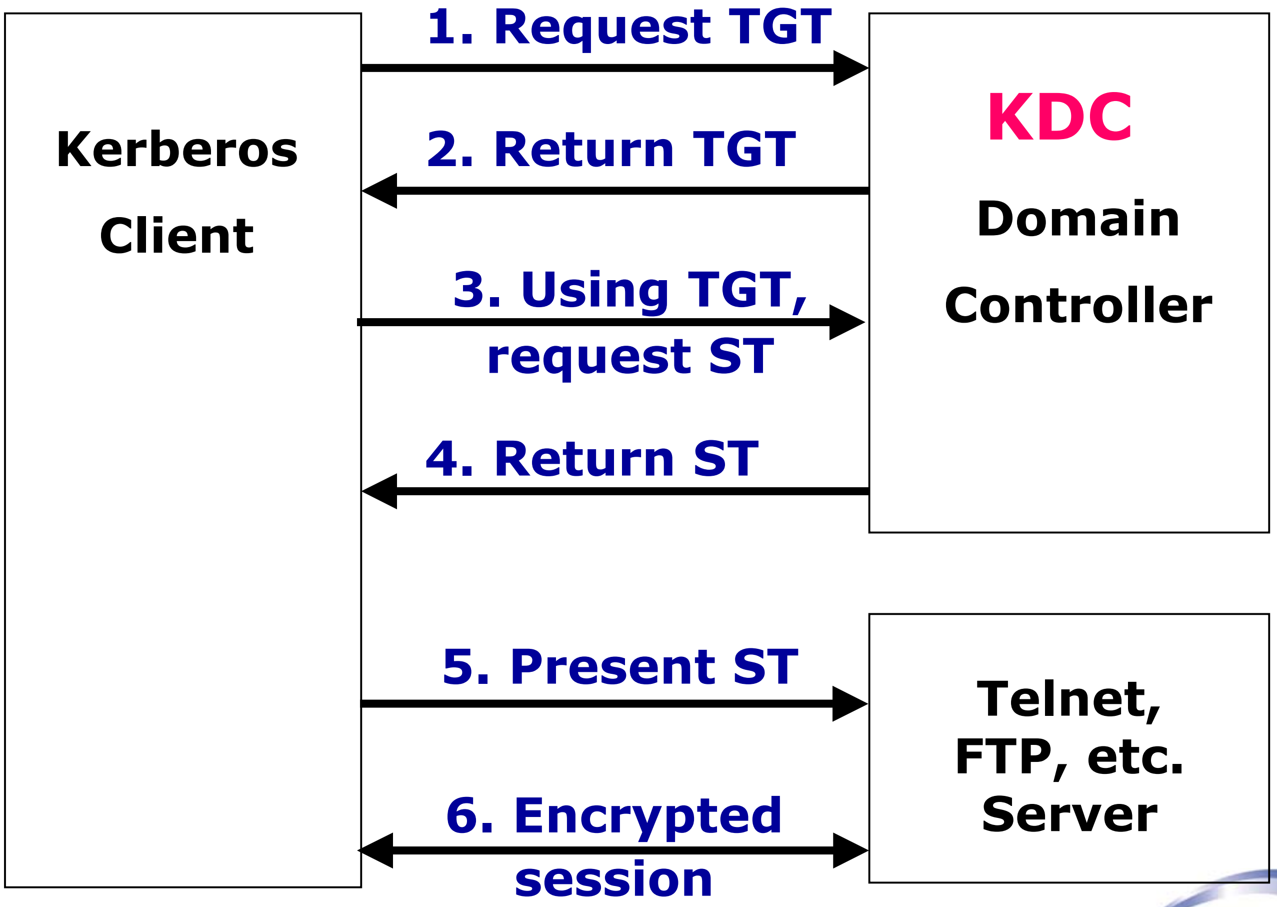
Kerberos

- Created at MIT in the early 1980s
- Current open-standard version is 5.0
- Used for authentication, data integrity, and encryption
- Implemented in Windows 2000 and XP via the Security Service Provider Interface (SSPI)

Kerberos - Features

- Secure authentication
  - Password never travels over the network
  - Memory-only credentials caches
- Data stream protections
  - Detection of data stream modification
  - 56-bit DES or 168-bit 3DES encryption
  - Telnet, FTP, *rlogin*, *rcp*, *rsh* protocols

Kerberos Basics



Kerberos - Pluses

- Mature, open standard that’s never been broken
- Minimal administration and server overhead
- Programmatic access - GSSAPI
- Widely available for UNIX/Linux, Windows, Unisys, OpenVMS,
- No patent or royalty encumbrances

Kerberos - Minuses

- The KDC(s) must be secured
- Prone to offline attacks on TGT; brute force attacks feasible on 56-bit keys
- Significant cost of implementation
  - Requires applications be “kerberized”
  - Administrators require specialized training

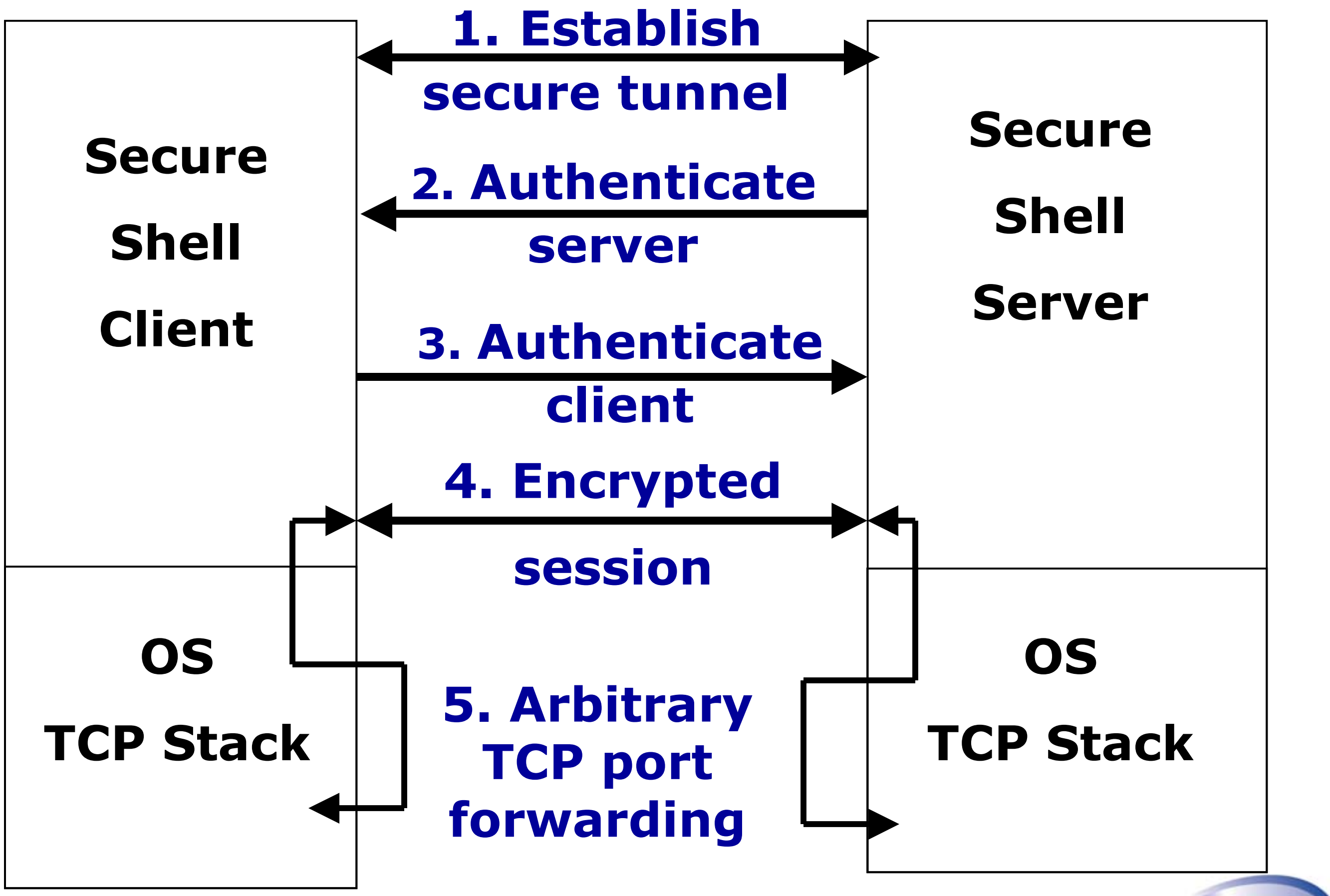
Secure Shell (SSH)

- Provides strong authentication - password, public key, Kerberos
- SSH-1 (deprecated) and SSH-2
- Replaces Telnet, *rlogin*, *rsh*, and *rcp*
- Secure forwarding of TCP connections, including X-11 protocol
- FTP replacement *sftp* in SSH-2

Secure Shell - Features

- 56-bit DES, 168-bit 3DES, 128-bit Arcfour, 128-bit CAST, 443-bit Blowfish and AES algorithms up to 256-bits
- OpenSSL libraries used for SSH-1 compatibility

Secure Shell Basics



Secure Shell - Pluses

- Internet draft, open-source standard
- Only one firewall port open
- No patent or royalty encumbrances
- Protocol-independent
- Available on UNIX/Linux, OpenVMS, Windows

Secure Shell - Minuses

- Administration problems
  - Certificates difficult to manage in timely manner if using user key
  - Specialized administration required if using Kerberos
- Requires regular security updates as bugs and holes are identified and fixed in the open-source implementation.

Security Availability

	SOCKS	SSL/		Secure
	VPN	TLS	Kerberos	Shell
UNIX/Linux	●	*	●	●
OpenVMS			●	●
MPE	Only through middleware servers			
Mainframe	●	●	*	●
Unisys	●		●	
Windows	●	●	●	●

Questions?

Thank you!