Global Information Assurance Certification Paper

How Safe is Remote Security Reporting Using bv-Control for UNIX® and Check Point™ FireWall-1 NG? A Case Study

Wayne Davis
GIAC Security Essentials Certification (GSEC)
Practical Assignment Version 1.4b
Option 2: Case Study in Information Security
Submitted 11/17/2003

Introduction

The need for centralized security monitoring is driven by current business trends which often require the integration of diverse networks due to corporate mergers, acquisitions and start-up operations. This case study documents the security issues associated with performing vulnerability assessment and platform management of UNIX/Linux systems from remote locations over publicly accessible networks. The tests were undertaken to prove the viability of a solution which allows a popular UNIX vulnerability reporting tool to be employed in a unique way and to verify that security concerns are properly addressed. In the paper I examine the ways in which the product of choice addresses the requirement to provide secure off-site vulnerability reporting. I analyze the way in which the product operates and attempt to demonstrate the extent to which it addresses communication related security issues. I then apply the information gathered to create a firewall configuration which will help to protect internal information resources while allowing Security Analysts to access increased levels of information from remote locations.

The environment of concern consists of several groups of UNIX/Linux servers located at physically diverse locations which provide various services to authorized users. Security, platform management, and other auditing and reporting is currently addressed on a local basis through the use of Administrator created scripts. Changes in corporate structure have produced the requirement to extend the ability to monitor the security and operational status of these servers while enabling the establishment of a coherent enterprise wide security policy which can be monitored and enforced from a central location.

In order to provide a workable solution, it was necessary to define the parameters and scope of tests to be performed and information to be derived. These included (1) determining the desired operating parameters (i.e., customer mandates); (2) understanding the security and technical details of the resources involved (how bv-Control for UNIX® communicates and authenticates, and how to install and configure Check Point™ FireWall-1 NG® to allow its proper operation); and (3) a general knowledge of security issues pertaining to the objective and how they could be addressed (i.e., protection of data flowing across an insecure medium, encryption and key exchange techniques, etc.).

Summary

BindView's bv-Control for UNIX® and Check Point™ FireWall-1® NG software were installed and configured in an arrangement designed to simulate the planned environment. With the test environment operational and the UNIX targets protected by the firewalls, several hundred queries were completed. The data provided by the tests was utilized to demonstrate the security measures employed by the reporting tool and firewall combination.

Background

In the existing environment, several geographically distinct and administratively autonomous pockets of UNIX servers exist. Each server group is locally administered, protected by a firewall, connected to the Internet and providing various services to authorized users. The firewalls are configured to hide the actual IP addresses of the internal networks and servers via Network Address Translation (NAT) and other security rules are in place to prevent unauthorized access to the systems. Security, patch monitoring, and other platform management is accomplished by local UNIX Administrators through locally created scripts and operating system utilities. Since reporting and management are performed locally, report information is not routinely passed over the network to any remote locations. For the same reason, reporting and system management is not consistent across the enterprise, but is implemented somewhat differently at each location, within overall corporate guidelines.

Testing

Parameters and scope of tests

In this case, the parameters and scope of work are defined by a corporate security group and the state of Internet security in general.

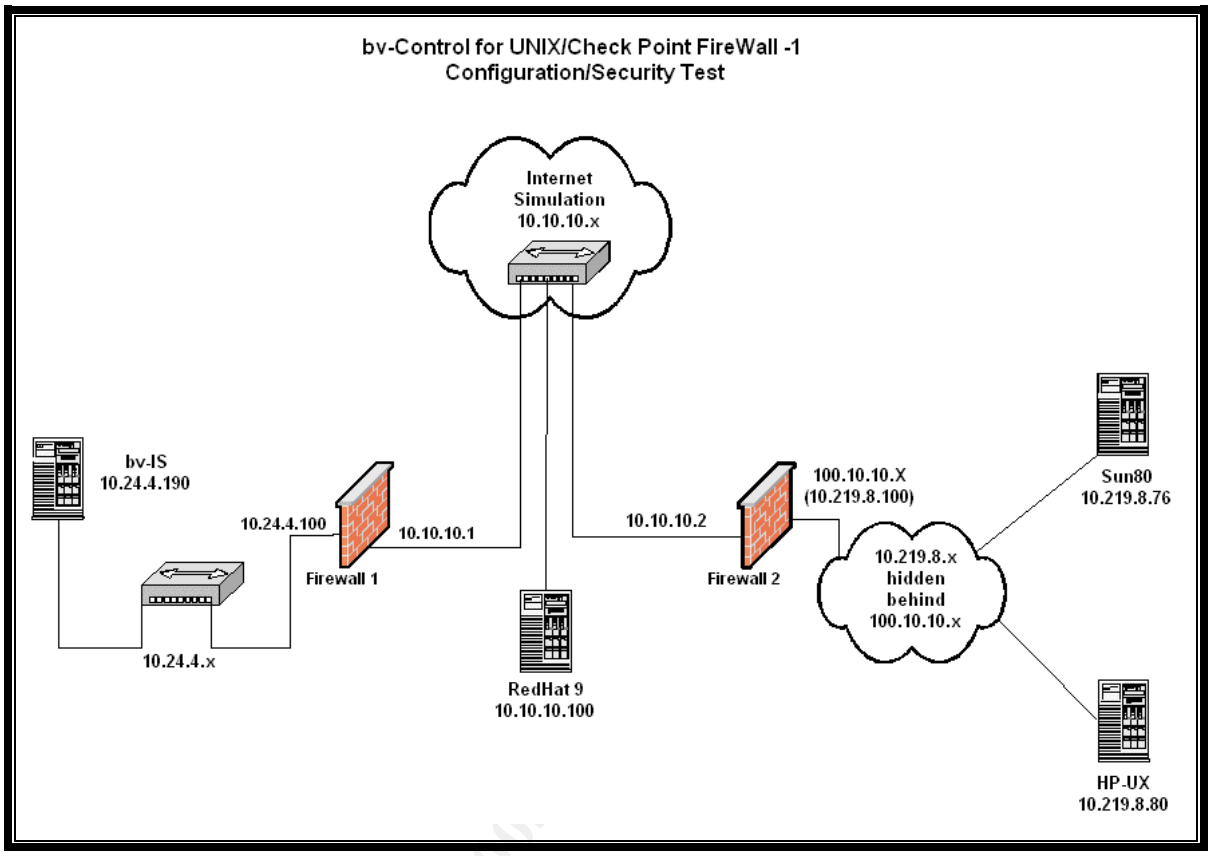
Requirements

- deploy bv-Control for UNIX to provide standardized and scalable enterprise wide reporting for UNIX security vulnerability and platform management at local and remote locations.
- deploy Check Point™ FireWall-1 NG® FP2 on each perimeter firewall, with Network Address Translation (NAT) enabled at the UNIX server locations.
- develop firewall rules to ensure protection will be maximized at all times and security will be maintained during application setup and data acquisition.
- examine bv-Control for UNIX operation to verify that secure access and authentication methods are employed between the UNIX targets and the monitoring console.
- verify that queries and other data are protected from acquisition by unauthorized persons while in transit between the monitoring console and the UNIX targets.

Network Configuration

Testing was performed in a closed lab environment with no connection to the Internet or production network. A self-contained network with three subnets was

configured to simulate two private networks and one Internet hop. Refer to *bv-Control for UNIX/Check Point™ FireWall-1 NG* test configuration.



bv-Control for UNIX / Check Point FireWall-1
Configuration / Security Test

Overview of test procedure

Prior to installing Check Point™ FireWall-1® NG software on the firewall hosts, the network was connected and bv-Control for UNIX was installed on the BindView Information Server (bv-IS) and a UNIX target machine. Static routes were defined on the firewall hosts and the product was tested to ensure connectivity and operational integrity. The BindView software was then removed and Check Point™ FireWall-1® NG software was installed on the firewall hosts.

The BindView Information Server/RMS Console (bv-IS) host and the UNIX targets were defined as Objects to the Check Point™ firewalls through which communications were to be routed. In addition, a TCP Service was defined on each firewall to handle query requests from the BindView Information Server. Network Address Translation (NAT) was configured on the firewall nearest the UNIX targets. Once this was accomplished, two security rules were defined on the firewalls, one to allow queries from the Information Server, and one to allow

application setup from the UNIX targets. Logging was enabled on each firewall and a RedHat Linux host running Snort Intrusion Detection System in packet sniffer mode was connected between the firewalls to monitor and log network traffic. The Solaris Snoop network packet sniffer utility was also used to monitor network activity and to capture setup and query communications for analysis.

With the firewalls functional and rules defined, the BindView Information Server/RMS Console and bv-Control for UNIX software were installed on the bv-IS. The BindView Agent (UNIX daemon) was installed on the UNIX targets and registered with the Information Server.

Again at the RMS Console, four bv-Control for UNIX tasks were created, each consisting of approximately twenty queries chosen at random from the product's list of Pre-Defined queries. Using the Windows Scheduler, the tasks were scheduled to run at thirty minute intervals.

Over 1,800 queries were performed during the test period. Using Snort and Snoop, thousands of network packets were captured for later analysis. Check Point™ FireWall-1® logged firewall throughput, and each of the UNIX systems was configured to log to its syslog file. Query information was also logged by bv-Control for UNIX in its application log files on both the UNIX target and the Information Server.

Log files and captured network packets were later analyzed to verify the security and integrity of the protection mechanisms, as well as to refine the firewall rule sets.

bv-Control for UNIX

Architecture

bv-Control for UNIX is a vulnerability assessment tool designed to report on operational and security aspects of UNIX/Linux targets[1]. It is an integral part of BindView Corporation's multi-faceted Risk Management System (RMS) and operates as a snap-in to BindView's RMS Console. Operationally, the tool consists of a GUI (BindView RMS Console), a Microsoft SQL database, and a collection of background processes which constitute the BindView Information Server (bv-IS), all of which reside on a Windows host. An agent resides on each UNIX target and consists of a bvcontrold process (listener daemon) with support libraries.

[1] "bv-Control for UNIX." URL: http://www.bindview.com/Products/VulnMgmt/AssesmentandSecurity/bv-Control_Unix.cfm (13 Nov. 2003).

Installation and Setup

During *installation and setup*, the BindView RMS Console and Information Server software is installed into a Microsoft Management Console (MMC) on the Windows host, and the bv-Control for UNIX software is installed, or snapped in, to the RMS Console. The Console uses a SQL database and requires that either MSDE or Microsoft SQL Server is installed and operational prior to initiating the Console installation. On the UNIX targets, the daemon software, support libraries, etc., are installed using standard UNIX package manipulation commands and a shell script (setup.sh) is invoked to configure the agent, register the target with the Information Server and start the daemon.

Once the required information is input to the shell script, a *Triple DES* encrypted *SSL* connection is established between the UNIX target and the bv-Control for UNIX Service, located on the Information Server. Registration information is transferred to the bv-IS and communication credentials are requested. The Information Server responds by generating and transferring a certificate and private key to the target for use in future authentication sessions.

Operations

When an authorized RMS Console user initiates a query, the Information Server opens a dialog with the bvcontrold daemon listening on port 1236 of the UNIX target. The daemon then forks a second, stand-alone process which establishes the SSL link. The Information Server authenticates to the UNIX agent and negotiates new session keys using the *Diffie-Hellman* Key Exchange algorithm (see *bv-Control for UNIX Query Communications*). The forked process services the query by gathering the requested data, formatting it into record length XML strings, encrypting the strings and transmitting the information back to the bv-IS. The Information Server decrypts, collects and formats the data into a report which can be displayed, printed, written to file, or passed to a customer-defined post-processing mechanism.

bv-Control for UNIX Tasks

A *task* is a collection of queries that have been grouped together to enable them to be invoked more easily. During testing, four tasks, composed of approximately twenty queries each were scheduled to run at thirty minute intervals.

Check Point™ FireWall-1® NG

Overview

A firewall serves as a concentration point for traffic entering and leaving a protected network. It allows administrators and security specialists to limit

information transfer to desirable communications by acting as a choke point for applying various inspection and filtering techniques, called rule sets. The rule sets developed for this application were determined by monitoring the performance of bv-Control for UNIX mainly through the use of network sniffers and log files.

Stateful Inspection

Check Point™ FireWall-1® NG is one of the industry's leading firewall products. Check Point™ is credited by Webopedia with coining the term, "Stateful Inspection"[2]. Stateful inspection refers to a firewall's ability to derive information not only from the header, but from all parts of the packet, as well as from its context.

In operation, packets are captured as they enter the firewall at the Network layer, the first software layer above the hardware (Data Link Layer). They may then be examined all the way up through the Application Layer and compared to the Administrator defined rule set to determine their validity.

As part of its stateful inspection mechanism, Check Point[3] creates a state table in which it stores information regarding ongoing communication sessions. Using this state table, packets may be examined in relation to previous communications which have traversed the firewall. In this manner, the state of an ongoing communication session may be used to determine whether a packet will be allowed through without the need to review the entire rule set. Session information is removed from the state table once the firewall sees a Fin or Rst packet, requiring subsequent traffic to be authenticated via the rule set.

Network Address Translation

Under RFC 1597[4], groups of IP addresses were set aside for use in private networks and by convention are not routed by Internet routers. Network Address Translation (NAT), as defined in RFC 1631[5], is essentially a routing function in which a machine with two network interface devices is connected to a private, internal net by one interface, and to the Internet by the other. Hosts on the internal network are assigned IP addresses chosen from the group set aside

---

[2] "Stateful Inspection." 18 Aug 2003. URL: http://www.webopedia.com/TERM/S/stateful_inspection.html (12 Nov. 2003).

[3] "Stateful Inspection™ Firewall Technology." URL: http://www.sofaware.com/downloads/Technical/Stateful_Inspection.pdf (12 Nov. 2003).

[4] Rekhter, Yakov, Moskowitz, Robert G., Karrenberg, Daniel, Jan de Groot, Geert. "RFC 1597-Address Allocation for Private Internets." March, 1994. URL: http://www.faqs.org/rfcs/rfc1597.html

[5] Egevang, Kjeld and Paul Francis. "RFC 1631-The IP Network Address Translator (NAT)". May, 1994. URL: http://www.faqs.org/rfcs/rfc1631.html (17 Nov. 2003).

under RFC 1597, while the second connection is assigned a globally unique address valid on the Internet. When a message is sent from the private net to the Internet, the NAT router simply removes the source address and adds its own public address such that the packets appear to originate with the NAT router. It then builds a connection table with the original source and destination information so replies can be returned to the originating host on the private net. The destination device believes itself to be communicating directly with the NAT device and uses its public address for any return messages.

(NAT) is employed by many types of routers and firewalls and serves a couple of important roles. First, NAT allows multiple hosts within a private network to share Internet access using only one public IP address. This conserves the finite number space afforded under the IPv4 guidelines. Second, hosts outside the perimeter of the private network are not aware of specific internal addresses, effectively hiding the internal hosts from public view and access. Instead, external hosts address communications to the NAT device (firewall or router), providing a certain amount of protection for the private net. After all, if you can't address a host, it is pretty difficult to gain access to it.

System Requirements

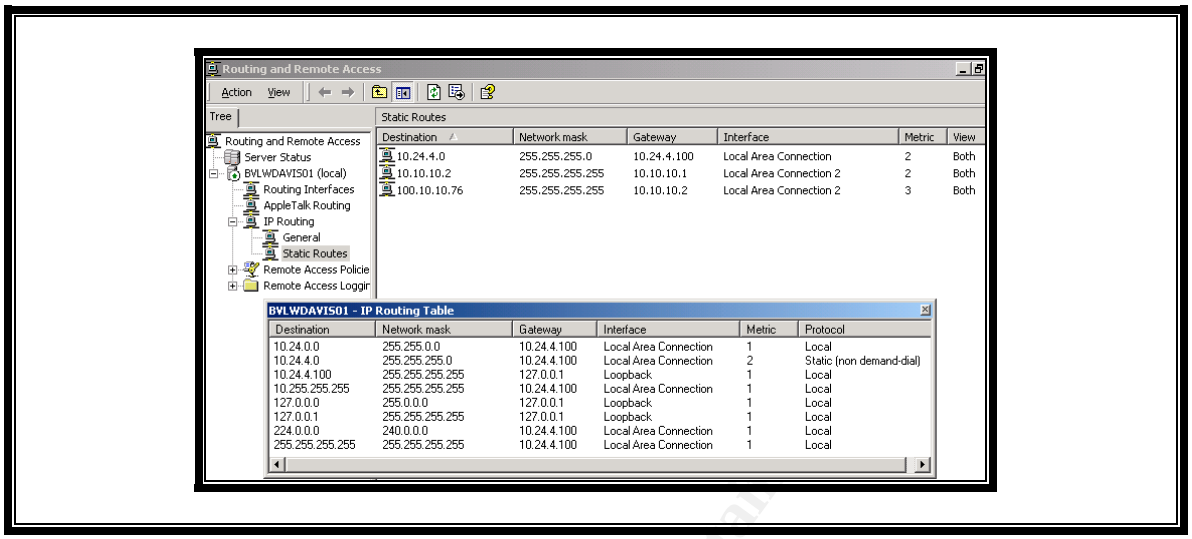
ENFORCEMENT MODULE
Operating Systems Windows 2000 Server
(SP1,SP2, SP3)
Windows 2000 Advanced Server
(SP1, SP2)
Windows NT 4.0 (SP6a)
Sun Solaris 8 (32 or 64 bit mode)
Sun Solaris 9 (64 bit mode)
Red Hat Linux 7.0, 7.2, 7.3
Nokia IPSO
Check Point SecurePlatform
Disk Space 40 MB
Memory 128 MB

Firewall hardware for the test consisted of two x86 PCs, each with dual network interface cards.

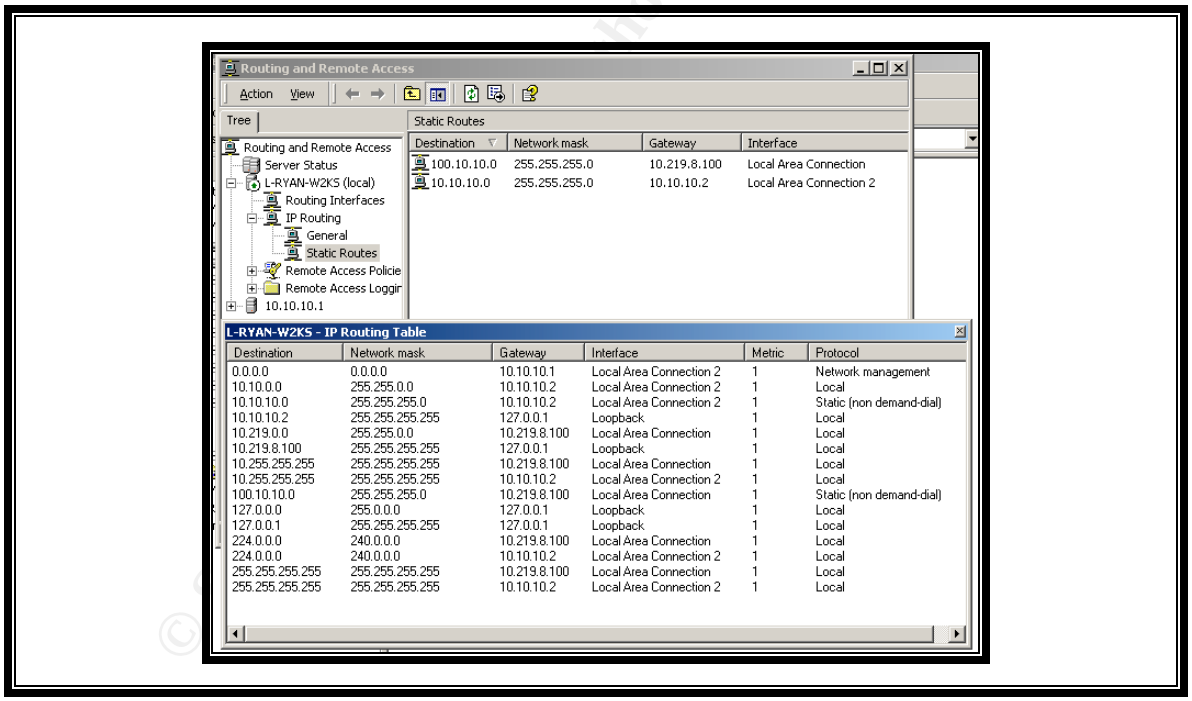
The Operating System software on the two firewall machines was Microsoft Windows 2000 Server installed using Windows default installation. No services or applications were disabled or added after the operating system installations were completed. Service packs were installed to bring the systems into compliance with Check Point™ requirements (above).

The only significant change to the firewall host configuration prior to installing and configuring the firewall software was to define static routes through the machines

to steer the bv-Control for UNIX traffic. The static routes were defined and verified prior to Check Point™ installation.



Firewall 1 Static Route and Routing Table



Firewall 2 Static Route and Routing Table

Check Point Installation and Configuration

Check Point ’s installation was performed using default settings, with a minimal amount of additional configuration. A fairly representative collection of security,

configuration, logging and reporting modules were installed in the default location, no remote management consoles were chosen, backward compatibility was not installed, and the respective firewall hosts served as their own Certificate Authorities.

Installation Procedure

- Autorun the installation CD. Select Next on the Welcome display.
- Read and accept the End User License Agreement (EULA).
- Select Server/Gateway Components from the Product Menu.
- Select the desired products from the Server/Gateway Components Menu.
- Verify the products to be installed from the InstallShield display and select Next.
- Select Enforcement Module & Primary Management from the Product Type Menu for installation on the local machine.
- Choose the installation destination (I chose default location).
- Install without backward compatibility for a new installation.
- Select the components to be installed.
- Add the license into the License Manager GUI.
- Add at least one Administrator ID and designate permissions for each.
- Specify remote clients from which an Administrator can log onto this management console.
- Enter random text into the Key Hit Session until a random seed has been created.
- Initialize and start the Internal Certificate Authority.
- Enter the Fully Qualified Domain Name (FQDN) of the Management Station.
- Send the FQDN to the Internal Certificate Authority.
- The Management Server fingerprint (generated above) is displayed and may be exported to a file for later examination.
- cpconfig hardens the OS until the first custom policy is installed.

Results of Testing

The following information was derived from research and testing and was used to demonstrate the extent to which bv-Control for UNIX addresses communication related security issues. The information gathered was used to create a firewall configuration which will help to protect internal information resources while allowing Security Analysts to access increased levels of information from remote locations.

BindView Communication Requirements

Communication during installation – During the software installation phase, there is no requirement for communications between the BindView Information

Server/RMS Console host and the UNIX targets. That is, the BindView infrastructure and the bv-Control for UNIX snap-in may be loaded from the distribution CDROM onto the bv-IS using standard Windows installation techniques. The same is true for installation of the agent software on the UNIX targets. The agent software can be loaded from a local CDROM drive and installed using standard UNIX/Linux commands. As an alternative, the agent software can be copied (binary ftp or the like) onto the target and installed from a local hard drive rather than from the distribution CD.

Communication during setup – During the setup phase, the Unix agent is configured, registered with the Information Server, and started. A command line Bourne Shell script, setup.sh, is executed to register the agent on each UNIX target. The script prompts the user for the IP address (or DNS name) of the BindView Information Server and other information with which to configure the snap-in on the bv-IS. The script validates the Information Server address by issuing an ICMP Echo Request (ping). If the ping response is not returned, an invalid address message is displayed and setup will not continue until the proper address is entered.

Once the bv-IS address is validated and the other required information has been entered, an SSL connection is established with the bv-Control for UNIX service listening on Windows port 1236. The sole function of this service is to monitor the network for target registrations. The registration information is passed to the bv-IS and a certificate and private key are returned. The bv-Control for UNIX daemon (bvcontrold) process is then started and listens on UNIX port 1236 for query requests from the Information Server.

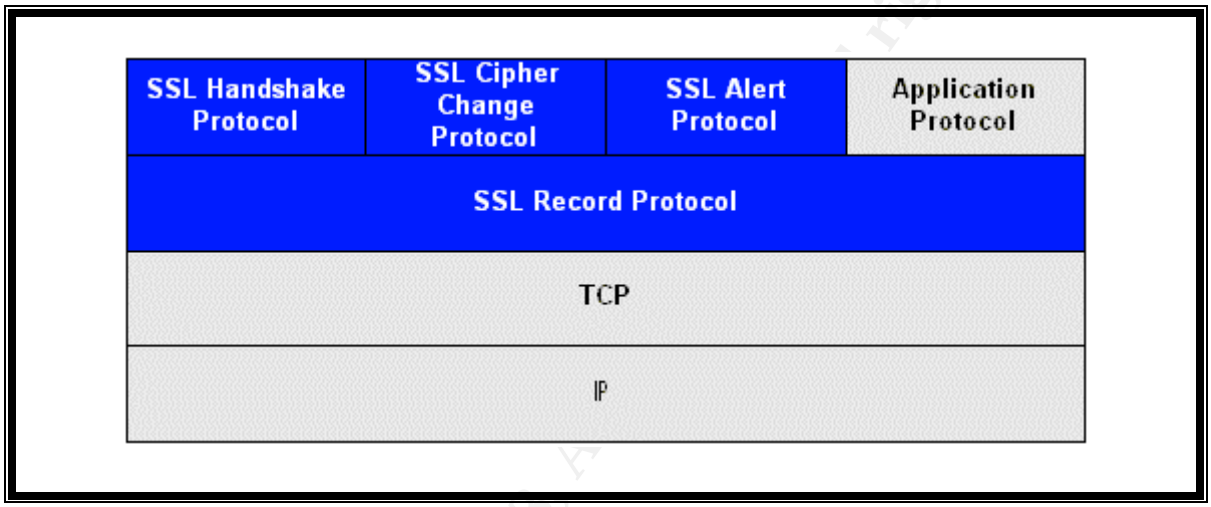
Communication during queries – When a query is initiated, the Information Server establishes a TCP connection with the target and sends a clear text "Hello" to the bvcontrold process. If the daemon can not verify that the source is the bv-IS defined during the setup phase, the connection is dropped. Both the bv-IS and UNIX target must be assigned static IP addresses for this reason. Once the address is verified, the SSL connection request is accepted. All subsequent communications are over the encrypted channel. Credentials are authenticated and session keys updated prior to launching the query.

Encryption and Communication

As previously stated, bv-Control for UNIX employs 3DES encrypted SSL communications and Diffie-Hellman Key Exchange algorithms to update private keys each time a communication session is initiated. The following overview examines these techniques to determine just how secure the data transmissions are between the bv-IS and the Unix agent.

SSL Overview

The Secure Socket Layer (SSL) communication protocol was originally developed by Netscape[6] for use with its Internet browser, but has since been applied to many different applications which require secure transmission (free from unauthorized reception or tampering) of sensitive information. SSL consists of four underlying protocols, SSL Record Protocol, SSL Handshake Protocol, SSL Cipher Change Protocol and SSL Alert Protocol, all of which enhance the security of standard TCP/IP communications.



SSL Protocol[7]

When an SSL connection is desired, the Handshake Protocol is invoked to establish the link. The Handshake Protocol is the most complex of the four protocols. The first step in setting up the connection is for the client and server to establish TCP communications and exchange Hello's which include SSL server versions, session IDs, cipher suites, a pre-master secret key and compression methods. In the case of application communications, many components other than the pre-master secret, session IDs, and other minor items may be hard coded and therefore remain constant. The pre-master secret is generated from random data, timestamp and other information.

Once this information is exchanged, SSL certificates are validated by examining expiration date, issuing authority and digital signature. The last step in the

---

[6] "Introduction to SSL." URL: *http://developer.netscape.com/docs/manuals/security/sslin/contents.htm#1046261* (12 Nov. 2003).

[7] Greenfield, David. "SSL and TLS." Network Magazine.com. 04 Dec2002. URL: *http://www.networkmagazine.com/shared/article/showArticle.jhtml;jsessionid=GSSF0IBPHZ1AQQSNDBCCKHQ?articleId=8703479&pgno=2* (12 Nov. 2003).

authorization process is for the client to verify the server's actual domain against the domain named in the server's certificate.

This final step is to protect against the "man-in-the-middle" attack in which all communications between the client and server are intercepted by an unauthorized host between the two. This host surreptitiously captures server to client communications, repackages the packets and forwards them to the client, posing itself as the server. It then captures replies from the client, repackages and forwards them on to the server in like manner. Both client and server believe they are in communication with each other. It is therefore important that the client verify that the server domain from which it receives the connection is indeed the same domain identified in the server's certificate[8]. If not, the session is aborted.

The job of the SSL Record Protocol is to package all information for secure transmission and to ensure its integrity. The cipher suite negotiated during the handshake is used to encrypt the data plus a Message Authentication Code (MAC). First, the information to be transmitted is broken down into 16 kb (or smaller) packets. The packet may be padded as necessary to meet size requirements. The MAC is created by hashing the secret key, data, padding and the sequence number to create a kind of super checksum. The data and MAC are then encrypted, a header is added and the packet is passed down the stack for transmission. At the destination the message is decrypted. A local version of the MAC is created and compared to the version contained in the packet to verify that the message has not been modified in transit.

The Alert Protocol conveys information about the session and consists of two bytes. The first byte is a severity code indicating (1) warning, or (2) fatal. The second byte contains a pre-defined error code. In the event of a message labeled fatal, the session is immediately terminated.

The Change Cipher Spec Protocol is used to signal agreement on the currently used set of protocols. Once agreed, the protocol set is carried through to the termination of the session.

Triple DES Encryption

The Data Encryption Standard (DES)[9] employs a 64 bit key to encrypt data for transmission across public channels. The DES encryption standard, adopted as a national standard in the United States in 1977, is no longer considered secure

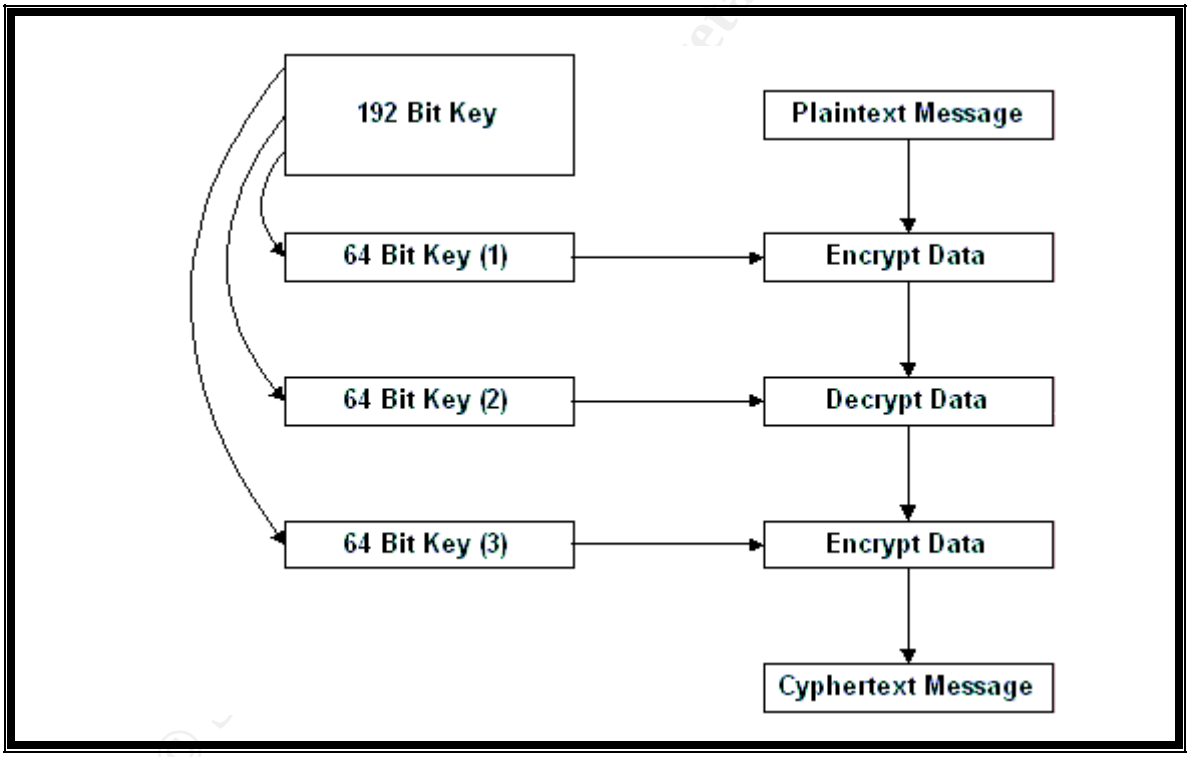
---

[8] "Introduction to SSL." URL: http://developer.netscape.com/docs/manuals/security/sslin/contents.htm#1046261 (16 Nov. 2003).

[9] "Data Encryption Standard (DES)" Federal Information Processing Standards Publication 46-2. 30 Dec. 1993. URL: http://www.itl.nist.gov/fipspubs/fip46-2.htm (12 Nov. 2003).

since studies have shown that it can be broken in a reasonable amount of time, given appropriate resources[10].

A derivative of the original standard is Triple DES (3DES), which uses three 64 bit keys for a total of 192 bits to encrypt the data. In operation, the key is broken into three parts and the data is encrypted three times using a 64 bit key each time. For that reason, Triple DES encryption is about three times slower than original DES, but can be immensely stronger due to the combined key length and techniques involved. In practice, the least significant bit of each byte is a parity bit, used to produce odd parity. The result is that the effective key length is reduced to 56 bits for an overall key length of 168 bits. Even so, Triple DES is still considered to be a very effective encryption technique and was endorsed by the National Institute of Standards and Technology for strong data encryption prior to the adoption of the new Advanced Encryption Standard. In fact, Netscape, the inventor of SSL says, "Triple DES is the strongest cipher supported by SSL"[11].



Triple DES Encryption

[10] "RSA Code-Breaking Contest Again Won by Distributed.Net and Electronic Frontier Foundation (EFF)." 19 Jan 1999. URL: *http://www.eff.org/Privacy/Crypto_misc/DESCracker/HTML/19990119_deschallenge3.html* (16 Nov. 2003).

[11] "Introduction to SSL." URL: *http://developer.netscape.com/docs/manuals/security/sslin/contents.htm#1046261* (12 Nov. 2003).

Diffie-Hellman Key Exchange

The Diffie-Hellman Key Exchange protocol (RFC 2631)[12] was first described by Whitfield Diffie and Martin Hellman in their publication, "New Directions in Cryptography"[13], and was considered a major advancement in cryptography. The protocol is widely used and provides the ability to exchange private (symmetric) keys across insecure medium, such as the Internet. Diffie-Hellman employs public key (asymmetric) cryptography to securely transfer and update the symmetric keys which are actually used for authentication and data encryption. In this way, parties to a conversation may be quickly authenticated using their private keys and still realize the ability to secure future communications by changing keys on a regular basis. With many applications, such as bv-Control for UNIX, the keys are updated during each session, providing enhanced protection against the "man-in-the-middle" or other types of network sniffer attacks as shown in the illustration below.

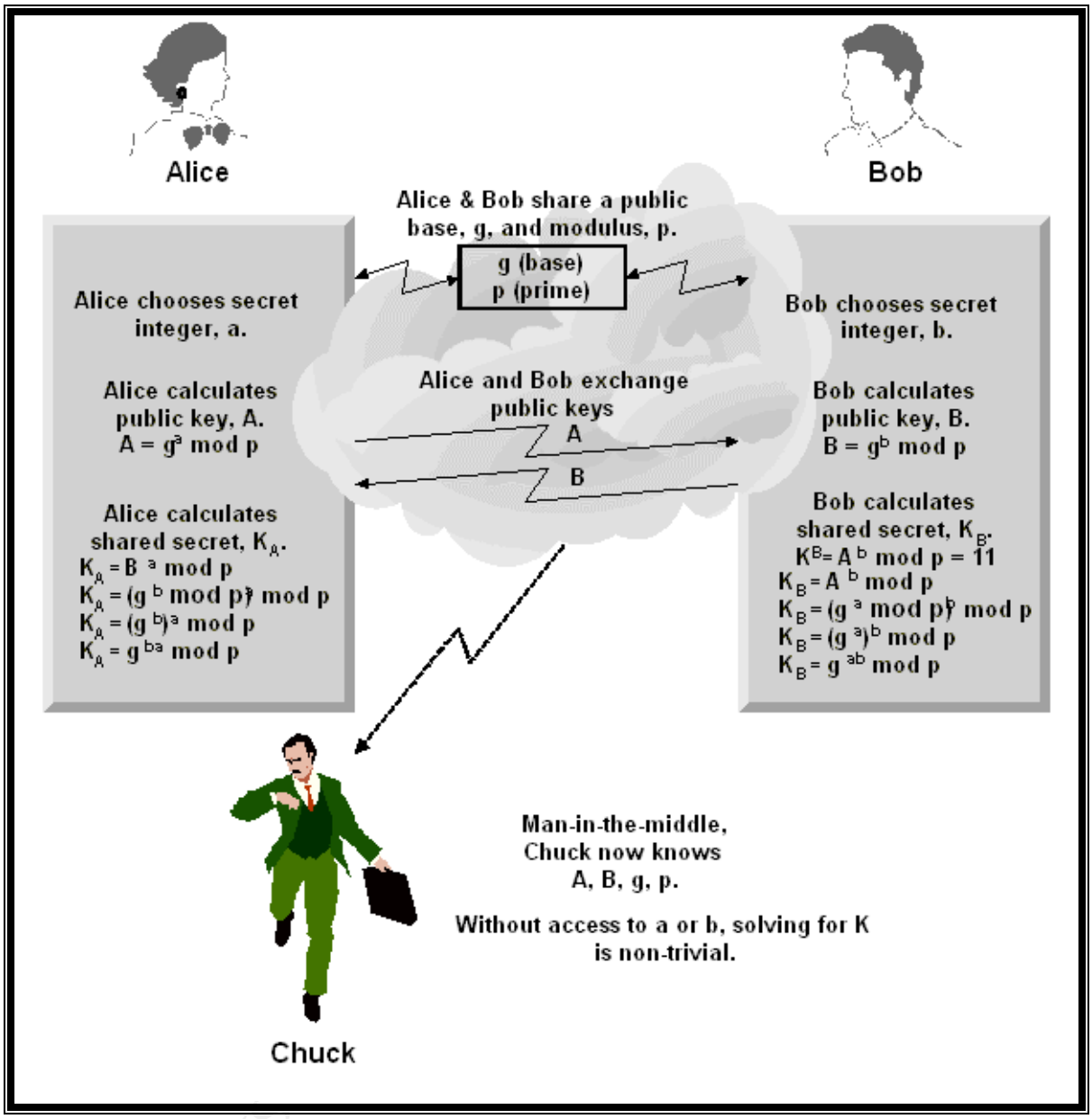
The protocol's use was aptly described by Alan Westrope in 1998, in an article entitled "Diffie-Hellman Key Exchange"[14]. In it he said:

> The protocol is based on modular exponentiation using a public base ***g***, a public modulus ***p***, and private exponents secretly chosen by each participant. For Alice and Bob to agree on a secret key, Alice first chooses a large integer ***a*** and Bob chooses a large integer ***b***. Alice then calculates ($g^a \bmod p$) = ***A*** and sends ***A*** to Bob, while Bob calculates ($g^b \bmod p$) = ***B*** and sends ***B*** to Alice. Alice computes her key as $B^a \bmod p$, and it's identical to Bob's key, $A^b \bmod p$, because both are equivalent to $g^{ab} \bmod p$. Alice and Bob can now use the shared key with a cipher of their choice for secure communication…

[12] Rescorla, E. "RFC 2631 - Diffie-Hellman Key Agreement Method." June, 1999. URL: http://www.ietf.org/rfc/rfc2631.txt (12 Nov. 2003).

[13] Diffie, Whitfield and Martin Hellman. "New Directions in Cryptography." June 3, 1976. URL: http://www.cs.jhu.edu/~rubin/courses/sp03/papers/diffie.hellman.pdf (12 Nov. 2003).

[14] Westrope, Alan. "Diffie-Hellman Key Exchange." 1998. URL: http://www.nyx.net/~awestrop/crypt/dh.htm (12 Nov. 2003).



Diffie-Hellman Key Exchange Algorithm

Firewall Configuration (Rule Sets)

The rule sets below were defined to demonstrate the minimum configuration requirement to protect the internal networks used in this simulation. Many variables, such as VPN, IPSec, Certificate Authorities and others will require the sets to be modified for real world application.

Define Source and Destination Objects. Define the BindView Information Server and each UNIX target to the firewall software as Objects (nodes). The definition should be similar to the following:

Host Node Definition for Firewall 2 (to provide NAT for Unix targets)						
General		Topology			NAT	
Name	IP Addr	Name	IP Addr	Netmask	Method	Net IP Addr
UnixTgtName	Actual IP	hme0	Actual IP	255.255..	Static	Public Addr
bv-IS	Actual IP					

Object Definition

Create a bvcontrol service. On each firewall between the bv-IS and UNIX target, define a bvcontrol service. Define it as being a TCP service addressed to port 1236. Be certain the firewall allows replies to be passed back to the originating host and port. Check Point™ does this as part of its stateful inspection feature in which a table of ongoing communications sessions is maintained based on information obtained from the packet. In that way, a response can be verified against the session table rather than having to examine the security rules.

Service Properties Definition			
Type	Name	Port (Dest)	Session Timeout
TCP	bvcontrol	1236	Default (3600s)

Service Definition

Create a firewall rule for queries. For queries, the firewall rule should be similar to the following:

Firewall Rule for Queries			
Source	Destination	Service	Action
bv-IS	Unix Targets	TCP bvcontrol	Allow

Query Rule

Create a firewall rule for setup. To run setup, the firewall rule should be similar to the following:

Firewall Rule for Setup			
Source	Destination	Service	Action
Unix Targets	bv-IS	TCP bvcontrol ICMP echo-request ICMP echo-reply	Allow

Setup Definition

Once setup has been completed, this rule can be deleted or otherwise disabled.

Network Address Translation

If NAT (Network Address Translation) is to be employed, it must be Static NAT or the firewall will change the addresses and ports when the session begins and authentication will fail.

Create firewall rules for NAT. The NAT rules (where only the UNIX targets are hidden) should be similar to the following:

Firewall Rule for Network Address Translation					
Original Packet			Translated Packet		
Source	Destination	Service	Source	Destination	Service
Unix Targets	Any	Any	Unix Targets (Valid Address)	Original	Original
Any	Unix Targets	Any	Original	Unix Targets	Original

Network Address Translation Definition

Conclusion

The results obtained through test observations and research support the premise that the security measures in place in the BindView product are more than adequate to protect the sensitive information being sent across an insecure medium while simultaneously prohibiting access to either the monitoring Console or the UNIX targets. The addition of the Check Point firewalls served to enhance the protection provided the private networks.

Significant security measures seen to be in effect included:

- BindView
    - private key authentication
    - triple DES (168 bit) encrypted SSL communications
    - Diffie-Hellman algorithm used to update private keys at each session
    - root privileges required to install and configure the UNIX daemon
    - only pre-defined query items are accepted by BindView UNIX daemon
    - only target registrations are accepted by the bv-Control for UNIX service on the bv-IS

- Check Point

    - role based administration and reporting
    - access to the protected network denied unless explicitly allowed by rule set
    - stateful inspection of communication packets
    - console/daemon communications limited by rule set to query or setup services
    - IP addresses of internal (UNIX) network protected by Network Address Translation

Appendix

BindView RMS Console and Information Server Installation

System Requirements

Minimum Requirements	BindView RMS Console	Information Server
CPU (MHz)	Pentium II 300+	Pentium III 600+
RAM (MB)	128+	196+
Disk Space (MB)	285	300
Windows Version	NT 4.0 (SP6a+) Or 2000	NT 4.0 (SP6) Or 2000
Internet Explorer	5.0	5.0
Other Software	Client for Microsoft Networks, Microsoft Outlook 2000	Client for Microsoft Networks, MSDE (v1.0 or 2000) or SQL Server (v7.0 or 2000)

MSDE Installation

BindView distributes an MSDE Install Wizard for installing Microsoft Data Engine (MSDE) v1.0 or 2000 on your machine. Invoke the MSDE Install Wizard from the Third-Party Components menu on the Install panel that appears when you insert your BindView RMS Console v7.20 SP 1 CD.

- Select Microsoft SQL Server Desktop Engine.
- MSDE Install Wizard. Select Next to continue.
- Computer Browser. Select the local machine.
- Install Directory. Select the desired installation location.
- Launch Install. Click Next to start the MSDE installation.
- Reboot after MSDE has been installed.

BindView RMS Console and Information Server Installation

- Autorun the BindView RMS Console and Information Server CD

- Select BindView RMS Console + Information Server from the Install menu.
- The Welcome panel of the Setup installation wizard appears.
    - Read the information on the panel and click Next.
- The Software License Agreement panel appears.
    - Read the license agreement and click Yes to accept the.
- The Choose Destination Location panel appears. Change the destination or click Next to continue.
- Select Program Folder display appears. Click Next to continue.
- Start Copying Files display. Click Next to continue.
- Security Alert. Reset or generate random password
- Security Alert. xp_cmdshell alert. Click OK to continue.
- Install RMS Console Service Pack 3
- Download and install RMS Console Hotfix CRB145914

bv-Control for UNIX Snap-in Installation

- Autorun the bv-Control for UNIX CD.
- Click the Install Products radio button .
- Click the Install button on the next screen.
- Click Next on the Welcome dialog screen.
- Read and click Yes to accept the terms of the End User License Agreement.
- Click Next on the Start Copying Files screen.
- Select the desired options and click Finish on the Setup Complete display.

Configure the BindView RMS Console

- Click Next on the BindView RMS Console Configuration wizard screen.
- Select bv-Control for UNIX on the Add/Remove Products page. Click Next.
- Add license codes. If this is a new installation, you will have to add license codes for each product.
    - To add license codes:
        - Enter the license code in the text box and click Add.
        - Click Have Disk and select a license file.
        - Drag a license file to the License Type list.
    - Repeat the process until you have entered all of your BindView product license codes. Click Next.
- Click Next on the License Summary panel.
- Click Next on the Add Licenses Completed panel.
- Click Next on the Add/Remove Products in Progress panel.
- The Add Users panel appears.
    - Type the desired user name or browse to select users.
    - Select properties in the User Properties box to define properties for each user. Click Next when you are finished.

- Review and then click Next on the Add Users Summary page.
- The Configuration Wizard page appears indicating configuration has been completed. Click Next.

Configure bv-Control for UNIX in the RMS Console

The BindView RMS Console opens. bv-Control for UNIX indicates Not Configured.

> **Note:** *The bv-Control for UNIX agent package must be installed on the UNIX target and setup.sh shell script must have been successfully completed prior to configuring the UNIX snap-in in the RMS Console*

- Right Click bv-Control for UNIX and choose Configure from the pulldown menu.
- Welcome dialog appears. Click Next.
- Information page appears. Read the information and then click Next.
- Click Next on the Configure UNIX Software dialog page.
- Add Credential Database is displayed.
  - Click and edit to add a new credential database
    - Type database name.
    - Type and Verify the database password.
  - Click Next to continue.
- Select Credentials panel appears.
  - Select the desired targets in the left window.
  - Select the desired credential database from the pulldown menu.
  - Click the right arrow to place the targets into the credential database.
  - Click Next.
- Assign a Credential Database to Each User. Click Next when finished.
- Credential Summary is displayed. Click Next.
- Completing the bv-Control for UNIX Configuration Wizard panel appears. Click Finish to return to the RMS Console.

Installing the bv-Control for UNIX Agent

bv-Control for UNIX Target Requirements

UltraSparc Station and Sun Solaris v2.6, 7, 8, 9
HP9000 servers or HP Visualize workstations and HP-UX v10.20, 11.0, 11.11 (11i)
RedHat Linux v6.2, 7.0, 7.1, 7.2, 7.3, 8.0, 9, Advanced Server 2.1
SuSE Linux v7.3, 8.0, 8.1, 8.2, Enterprise Server 8.1
AIX v4.3.3, 5.1, 5.2
20 MB disk space (in /opt for Solaris, HP-UX and SuSE Linux, in /usr/local for AIX and RedHat Linux)

TCP/IP networking hardware and software

Installation Procedure

- Logon to the target system as root user.
- Mount the CD-ROM drive using UNIX commands appropriate for your UNIX system.
- Install the package using commands appropriate for your UNIX system.

    - **AIX**
        - installp –acNQqwX –d /mnt/bv-Control_for_Unix/AIX/bv-Control.7.30.<build number> bvControl.rte. Press Enter.

    - **HP-UX**
        - swinstall –s /mnt/bv-Control_for_Unix/HP-UX/bvControl. Press Enter.

    - **Solaris**
        - pkgadd –d /cdrom/cdrom0/bv- Control_for_Unix/SunOS/ bv-Control.7.30.<build number> . Press Enter.

    - **RedHat Linux**
        - rpm –i /mnt/cdrom/ bv- Control_for_Unix/Linux/RedHat/ bv-Control.7.30.<build number>.i386.rpm . Press Enter.

    - **SuSE Linux**
        - rpm –i /media/cdrom/ bv- Control_for_Unix/Linux/SuSE/ bv-Control.7.30.<build number>.i386.rpm . Press Enter.

Register the bv-Control for UNIX Agent with the RMS Console

- Run the shell script (*setup.sh*)

    - For Solaris, SuSE Linux and HP-UX, type the following:
        - /opt/BindView/bvcontrol/setup.sh. Press Enter.

    - For AIX and RedHat Linux, type the following:
        - /usr/local/BindView/bvcontrol/setup.sh. Press Enter

- Type a or A to add configuration data. Press Enter.
- Type the system name or the IP address of the bv-IS. Press Enter.
- Enter a description to help identify the UNIX target. Press Enter.
- Type a Resource Name. Press Enter.
- Type a password. Press Enter.
- Confirm the password. Press Enter.

Note: *The following information regarding the setup.sh shell script session was captured using the Sun Microsystems Solaris 9 script*[15] *command. Once the command is invoked, subsequent commands and screen dialog are captured in a file in the local directory named "*typescript*". The command line to invoke scripts is:*

#script
.
#Run the commands to be captured here….
.
#^D (ends capture)

```
#cd /opt/BindView/bvcontrol
# ./setup.sh
Select <A>dd or <D>elete or <N>otify configuration data or
<Q>uit to exit: A

bv-Control for UNIX needs to exchange keys and
configuration data with the BindView Information Server.
IP address or system name can't be empty.
Enter the system name (or IP Address) of the BindView
Information Server:
bv-IS

Please enter a message to help identify this machine:
Sun80

bv-Control for UNIX needs to setup authentication via
passwords.
You will need to add a Resource Name and password for this
machine.
The password must be an Alphanumeric string containing no
spaces.
After you enter this information, you will need to
configure a credential database
in the BindView RMS Console with this information.
*** Warning: Please do not use a User Name and password in
the /etc/passwd file ***
Please enter a Resource Name to be used for authentication.
Resource Name: res_name

Please enter a Password to be used for authentication:
Please Verify the Password:
Contacting bv-IS, Please wait...
<Records><Cert>Success</Cert><Key>Success</Key></Records>
The daemon appears to be already running.
```

Setup.sh Script

[15] "script – make a record of a terminal session." URL: *http://docs.sun.com/db/doc/802-5747-01/6i9g1assk?a=view* (16 Nov. 2003).

Setup Session Capture

The following network session was captured during the registration of the UNIX daemon with the BindView Information Server using the setup.sh shell script provided as part of the bv-Control for UNIX package.

The session was captured using the Sun Microsystems Solaris v. 9.0 Snoop command. Portions of the session have been edited for security and readability. IP addresses were changed to hostnames and comments were added.

The command syntax[16] used to perform the capture was:

#snoop -PVrx 0 bv-IS Sun80 > setup.txt

Where:

P = non-promiscuous mode
V = intermediate level verbose
r = do not perform IP to Hostname resolution
x 0 = capture data beginning at offset 0 (capture the entire packet)
setup.txt = capture file

Setup pings the Information Server to validate the address.

_____

Sun80 -> bv-IS ETHER Type=0800 (IP), size = 98 bytes
Sun80 -> bv-IS IP D=bv-IS S=Sun80 LEN=84, ID=5773, TOS=0x0, TTL=255
Sun80 -> bv-IS **ICMP Echo request** (ID: 2728 Sequence number: 0)

```
 0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
16: 0054 168d 4000 ff01 3ff5 0adb 084c 0adb    .T..@...?....L..
32: 0725 0800 1ebe 0aa8 0000 3fac 2d86 000e    .%.......?.-...
48: 7656 0809 0a0b 0c0d 0e0f 1011 1213 1415    vV..............
64: 1617 1819 1a1b 1c1d 1e1f 2021 2223 2425    ..........  !"#$%
80: 2627 2829 2a2b 2c2d 2e2f 3031 3233 3435    &'()*+,-./012345
96: 3637                                       67
```

_____

bv-IS -> Sun80 ETHER Type=0800 (IP), size = 98 bytes
bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=84, ID=21635, TOS=0x0, TTL=128
bv-IS -> Sun80 **ICMP Echo reply** (ID: 2728 Sequence number: 0)

---

[16] "snoop(1M) – capture and inspect network packets." URL: *http://docs.sun.com/db/doc/816-0211/6m6nc677k?a=view* (14 Nov. 2003).

```
 0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
16: 0054 5483 4000 8001 80ff 0adb 0725 0adb    .TT.@.........%..
32: 084c 0000 26be 0aa8 0000 3fac 2d86 000e    .L..&.....?.-...
48: 7656 0809 0a0b 0c0d 0e0f 1011 1213 1415    vV..............
64: 1617 1819 1a1b 1c1d 1e1f 2021 2223 2425    ..........  !"#$%
80: 2627 2829 2a2b 2c2d 2e2f 3031 3233 3435    &'()*+,-./012345
96: 3637                                       67
```

TCP Session Setup

_____

Sun80 -> bv-IS TCP D=1236 S=32869 **Syn** Seq=3930255741 Len=0 Win=49640 Options=<mss 1460,nop,nop,sackOK>

bv-IS -> Sun80 TCP D=32869 S=1236 **Syn Ack**=3930255742 Seq=1226550716 Len=0 Win=64240 Options=<mss 1460,nop,nop,sackOK>

Sun80 -> bv-IS TCP D=1236 S=32869 **Ack**=1226550717 Seq=3930255742 Len=0 Win=49640

SSL Handshake

_____

Sun80 -> bv-IS ETHER Type=0800 (IP), size = 104 bytes
Sun80 -> bv-IS IP D=bv-IS S=Sun80 LEN=90, ID=5777, TOS=0x0, TTL=64
Sun80 -> bv-IS TCP D=1236 S=32869 Push Ack=1226550717 Seq=3930255742 Len=50 Win=49640

```
 0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
                                  ~
                                  ~
96: 8900 0002 0012 0100                       ........
```

_____

bv-IS -> Sun80 ETHER Type=0800 (IP), size = 60 bytes
bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=40, ID=21638, TOS=0x0, TTL=128
bv-IS -> Sun80 TCP D=32869 S=1236 Ack=3930255792 Seq=1226550717 Len=0 Win=64190

```
 0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
                                  ~
48: fabe 37e9 0000 0000 0000 0000             ú.7.........
```

SSL Authentication

_____

bv-IS -> Sun80 ETHER Type=0800 (IP), size = 818 bytes
bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=804, ID=21643, TOS=0x0, TTL=128

bv-IS -> Sun80 TCP D=32869 S=1236 Push Ack=3930255792 Seq=1226550717 Len=764 Win=64190

```
        0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
                                    ~
                                    ~
                        Transfer Certificate
                                    ~
                                    ~
      256: 5504 0b13 1362 762d 436f 6e74 726f 6c20    U....bv-Control
      272: 666f 7220 554e 4958 3111 300f 0603 5504    for UNIX1.0...U.
      288: 0a13 0842 696e 6456 6965 7730 81f1 3081    ...BindView0..0.
                                    ~
                                    ~
      816: 0000                                       ..
```

_____

Sun80 -> bv-IS ETHER Type=0800 (IP), size = 54 bytes
Sun80 -> bv-IS IP D=bv-IS S=Sun80 LEN=40, ID=5780, TOS=0x0, TTL=64
Sun80 -> bv-IS TCP D=1236 S=32869 Ack=1226551481 Seq=3930255792 Len=0 Win=48876

```
        0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
                                    ~
       48: beec 70bf 0000                             ..p...
```

_____

Sun80 -> bv-IS ETHER Type=0800 (IP), size = 211 bytes
Sun80 -> bv-IS IP D=bv-IS S=Sun80 LEN=197, ID=5781, TOS=0x0, TTL=64
Sun80 -> bv-IS TCP D=1236 S=32869 Push Ack=1226551481 Seq=3930255792 Len=157 Win=48876

```
        0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
                                    ~
                                    ~
      176: ae8b f847 ebf9 13d6 7425 5ad3 20fb c20f    ..øG.ù..t%Z. ...
      192: 1611 f78a 8ccf 89ce 05ca 4b93 3171 a1b9    ..........K.1q..
      208: 1327 a4                                    .'.
```

_____

bv-IS -> Sun80 ETHER Type=0800 (IP), size = 129 bytes
bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=115, ID=21644, TOS=0x0, TTL=128
bv-IS -> Sun80 TCP D=32869 S=1236 Push Ack=3930255949 Seq=1226551481 Len=75 Win=64033

```
        0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
```

```
 16: 0073 548c 4000 8006 80d2 0adb 0725 0adb    .sT.@.....%..
                                    ~
                                    ~
112: c655 312e d942 71a2 c9a0 3ba0 7d75 5360    .U1..Bq...;.}uS`
128: 63                                  c
```

_____

```
Sun80 -> bv-IS ETHER Type=0800 (IP), size = 54 bytes
Sun80 -> bv-IS IP  D=bv-IS S=Sun80 LEN=40, ID=5782, TOS=0x0, TTL=64
Sun80 -> bv-IS TCP D=1236 S=32869 Ack=1226551556 Seq=3930255949
Len=0 Win=48801

 0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
16: 0028 1696 4000 4006 ff13 0adb 084c 0adb    .(..@.@......L..
32: 0725 8065 04d4 ea42 f24d 491b b104 5010    .%.e...B.MI...P.
48: bea1 7022 0000                            ..p"..
```

Encrypted Data Transfer

_____

```
Sun80 -> bv-IS ETHER Type=0800 (IP), size = 339 bytes
Sun80 -> bv-IS IP  D=bv-IS S=Sun80 LEN=325, ID=5783, TOS=0x0, TTL=64
Sun80 -> bv-IS TCP D=1236 S=32869 Push Ack=1226551556
Seq=3930255949 Len=285 Win=48801

 0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
                                    ~
                                    ~
```

Transfer setup information to bv-IS

```
                                    ~
                                    ~
320: 067e 728b 9661 9237 1f93 3c74 16fb 12c2    .~r..a.7..<t....
336: c6a6 42                                  ..B
```

_____

```
bv-IS -> Sun80  ETHER Type=0800 (IP), size = 60 bytes
bv-IS -> Sun80  IP  D=Sun80 S=bv-IS LEN=40, ID=21645, TOS=0x0, TTL=128
bv-IS -> Sun80  TCP D=32869 S=1236 Ack=3930256234 Seq=1226551556
Len=0 Win=63748

 0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
16: 0028 548d 4000 8006 811c 0adb 0725 0adb    .(T.@.....%..
32: 084c 04d4 8065 491b b104 ea42 f36a 5010    .L...eI....B.jP.
48: f904 34a2 0000 0000 0000 0000              ù.4.........
```

_____

```
bv-IS -> Sun80  ETHER Type=0800 (IP), size = 1514 bytes
```

bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=1500, ID=21646, TOS=0x0, TTL=128
bv-IS -> Sun80 TCP D=32869 S=1236 Ack=3930256234 Seq=1226551556 Len=1460 Win=63748

```
 0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
16: 05dc 548e 4000 8006 7b67 0adb 0725 0adb    ..T.@...{g...%..
                              ~
```

Certificate and Key Returned to UNIX Target

```
                              ~
1488: 5695 9a9c d6a6 d2e3 1875 dc97 de98 a18e    V........u......
1504: 79d9 eac0 e944 e626 749d                    y....D.&t.
```

---

bv-IS -> Sun80 ETHER Type=0800 (IP), size = 359 bytes
bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=345, ID=21647, TOS=0x0, TTL=128
bv-IS -> Sun80 TCP D=32869 S=1236 Push Ack=3930256234 Seq=1226553016 Len=305 Win=63748

```
 0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
16: 0159 548f 4000 8006 7fe9 0adb 0725 0adb    .YT.@........%..
                              ~
                              ~
                              ~
336: d288 885d f4e7 7622 bfb2 3388 526f 8051    ...]..v"..3.Ro.Q
352: 9bcf 8e24 6cbb 07                          ...$l..
```

Sun80 -> bv-IS ETHER Type=0800 (IP), size = 54 bytes
Sun80 -> bv-IS IP D=bv-IS S=Sun80 LEN=40, ID=5784, TOS=0x0, TTL=64
Sun80 -> bv-IS TCP D=1236 S=32869 Ack=1226553321 Seq=3930256234 Len=0 Win=47036

```
 0: 0006 5baf e670 0800 20d1 c5bb 0800 4500    ..[..p.. .....E.
16: 0028 1698 4000 4006 ff11 0adb 084c 0adb    .(..@.@......L..
32: 0725 8065 04d4 ea42 f36a 491b b7e9 5010    .%.e...B.jI...P.
48: b7bc 6f05 0000                             ..o...
```

---

bv-IS -> Sun80 ETHER Type=0800 (IP), size = 83 bytes
bv-IS -> Sun80 IP D=Sun80 S=bv-IS LEN=69, ID=21648, TOS=0x0, TTL=128
bv-IS -> Sun80 TCP D=32869 S=1236 Push Ack=3930256234 Seq=1226553321 Len=29 Win=63748

```
 0: 0800 20d1 c5bb 0006 5baf e670 0800 4500    .. .....[..p..E.
16: 0045 5490 4000 8006 80fc 0adb 0725 0adb    .ET.@........%..
```

```
32: 084c 04d4 8065 491b b7e9 ea42 f36a 5018    .L...eI....B.jP.
48: f904 c873 0000 1503 0000 1863 b6ba b831    ù..s.......c...1
64: e15c 0450 1c77 1bb9 3f6d 5927 fc5b 0cce    .\.P.w..?mY'.[..
80: fe35 0b                                    þ5.
```

Terminate the TCP Session

_____

bv-IS -> Sun80 TCP D=32869 S=1236 **Fin** Ack=3930256234 Seq=1226553350 Len=0 Win=63748

Sun80 -> bv-IS TCP D=1236 S=32869 **Ack**=1226553351 Seq=3930256234 Len=0 Win=49640

bv-IS -> Sun80 TCP D=32869 S=1236 **Rst** Seq=1226553351 Len=0 Win=0

Tasks

The following table lists the pre-defined bv-Control for UNIX queries that composed the tasks run during testing. The queries were chosen at random.

Task1 for firewall test	Task2 for firewall test
Targets	Machine Configuration
Detailed List of Filesystems	Machine is WINS Server
Detailed List of Installed Packages	Machines with Minimum Amount of RAM
Detailed List of Internet Services	Network Configuration
Detailed List of Local Groups	Wrong Default Router - HP and SunOS only
Detailed List of Local Users	All Filesystems Disk Space
Detailed List of Machines	All Filesystems Documentation
Detailed List of Processes	File CRC
Machine Configuration	Local Filesystems Disk Space
Network Configuration	Local Filesystems Documentation
Advanced Machine Documentation	Unused Filesystems
Machine Hardware Documentation	Account Lockout Documentation
Machine Network Documentation	Accounts by User ID
Syslog	Accounts by User Name
Quick List of Filesystems	Basic User Documentation
Quick List of Installed Packages	Detailed User Documentation
Quick List of Internet Services	Group Membership
Quick List of Local Users	Groups
Quick List of Processes	Login Accounts
30k files	
Task 3 for firewall test	**Task 4 for firewall test**

Users Without a Home Directory	List of Internet Services
Advanced Process Information	Machines Allowing FTP
Basic Process Information	Machines Allowing Remote Login
Process CPU and Memory Utilization	Machines Allowing Remote Shell
Internet Services Information	Machines Allowing Telnet
Detailed List of Installed Packages	Machines Allowing TFTP
HP-UX Systems	Machines with Old Package Installed
Contents of syslog Configuration File	Solaris Systems
Search for All su Attempts Recorded in a syslog File	Specific Operating System Machines
	Contents of the syslog Configuration File
Contents of Login Defaults File	Contents of inetd Configuration File
Contents of nsswitch Configuration File	Contents of inittab File
Contents of PAM Files	Contents of Local Services File
Contents of syslog Configuration File	Contents of System-wide profile File
Contents of system File - Solaris Only	Passwd File Specifies NIS
Machine Hardware Documentation	Advanced Machine Documentation
Current Login Terminals	Failed Logon Attempts
Machines User is Logged Into	Find Files with SUID or SGID Bits Set
Accounts Never Logged In	Users with Invalid Shells
Administrator Equivalent Accounts	Users Without Passwords

Query Session Capture

The following session was captured using Snort™ in the packet capture mode. Snort™ was running on a RedHat Linux 9 machine located between Firewall 1 and Firewall 2.

> **Note**: *Snort™ is an open source intrusion detection system and is available for download at* http://www.snort.org . *The command line used to capture the session below is*[17]*:*
>
> ./snort -dev -l ./log -h 10.10.10.0/24
>
> Where: "dev –l" tells snort to print TCP/IP, data link and application data into the directory "log."
> Captured packet information is displayed and printed into directories named the same as the IP address of the host that is not on the 10.10.10.0 net under "log".

[17] "Chapter 1 – Snort Overview." URL: *http://www.snort.org/docs/writing_rules/chap1.html#tth_sEc1.2* (16 Nov. 2003).

Source-bv-IS
(dynamic port assignment)

Destination-Target daemon
(port 1236)
Public Address
(actual address hidden)

TCP Connect

```
08/18-14:56:03.535397 0:4:76:8C:F5:4A -> 0:50:4:D1:81:A1 type:0x800
len:0x3E
10.24.4.190:1241 -> 100.10.10.76:1236 TCP TTL:127 TOS:0x0 ID:4760
IpLen:20 DgmLen:48 DF
******S* Seq: 0x4DC5D6B0  Ack: 0x0  Win: 0xFAF0  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/18-14:56:03.536506 0:50:4:D1:81:A1 -> 0:4:76:8C:F5:4A typ
len:0x3E
100.10.10.76:1236 -> 10.24.4.190:1241 TCP TTL:63 TOS:0x0 ID:25370
IpLen:20 DgmLen:48 DF
***A**S* Seq: 0xB40059F6  Ack: 0x4DC5D6B1  Win: 0xC1E8  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/18-14:56:03.536894 0:4:76:8C:F5:4A -> 0:50:4:D1:81:A1 type:0x800
len:0x3C
10.24.4.190:1241 -> 100.10.10.76:1236 TCP TTL:127 TOS:0x0 ID:4761
IpLen:20 DgmLen:40 DF
***A**** Seq: 0x4DC5D6B1  Ack: 0xB40059F7  Win: 0xFAF0  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/18-14:56:03.537479 0:4:76:8C:F5:4A -> 0:50:4:D1:81:A1 type:0x800
len:0x3D
10.24.4.190:1241 -> 100.10.10.76:1236 TCP TTL:127 TOS:0x0 ID:4762
IpLen:20 DgmLen:47 DF
***AP*** Seq: 0x4DC5D6B1  Ack: 0xB40059F7  Win: 0xFAF0  TcpLen: 20
48 65 6C 6C 6F 0D 0A                                  Hello..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/18-14:56:03.537672 0:50:4:D1:81:A1 -> 0:4:76:8C:F5:4A type:0x800
len:0x3C
100.10.10.76:1236 -> 10.24.4.190:1241 TCP TTL:63 TOS:0
IpLen:20 DgmLen:40 DF
***A**** Seq: 0xB40059F7  Ack: 0x4DC5D6B8  Win: 0xC1E8

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/18-14:56:03.798716 0:50:4:D1:81:A1 -> 0:4:76:8C:F5:4A type:0x800
len:0x52
100.10.10.76:1236 -> 10.24.4.190:1241 TCP TTL:63 TOS:0x0 ID:25372
IpLen:20 DgmLen:68 DF
***AP*** Seq: 0xB40059F7  Ack: 0x4DC5D6B8  Win: 0xC1E8  TcpLen: 20
```

Server/Daemon (SSL) Handshake

```
48 65 6C 6C 6F 7E 39 38 34 7E 37 7E 33 30 7E 33  Hello~984~7~30~3
33 35 7E 37 33 30 7E 37 33 30 0D 0A              35~730~730..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Hello   ProcessID   Major/Minor Version   Build No.   RapidFire Level

Hello   984   7   30   335   730   730

Security Patch Level

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
08/18-14:56:03.813349 0:4:76:8C:F5:4A -> 0:50:4:D1:81:A1 type:0x800
len:0x68
10.24.4.190:1241 -> 100.10.10.76:1236 TCP TTL:127 TOS:0x0 ID:4763
IpLen:20 DgmLen:90 DF
***AP*** Seq: 0x4DC5D6B8  Ack: 0xB4005A13  Win: 0xFAD4  TcpLen: 20
16 03 00 00 2D 01 00 00 29 03 00 3F 41 2F 60 91  ....-...)..?A/`.
1A 8C EE C6 75 3C A0 E3 D3 A4 83 94 37 8A D5 5B  ....u<......7..[
10 8D 87 D4 FF 32 CD 02 00 4D 25 00 00 02 00
01 00
```

Encrypted Comms Begins

SSL Setup

Authenticate

Diffie-Hellman Key Exchange

Query/Data

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

08/18-14:56:03.813703 0:50:4:D1:81:A1 -> 0:4:
len:0x3C
100.10.10.76:1236 -> 10.24.4.190:1241 TCP TTL:63 TOS:0x0 ID:25373
IpLen:20 DgmLen:40 DF
***A**** Seq: 0xB4005A13  Ack: 0x4DC5D6EA  Win: 0xC1E8  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

08/18-14:56:03.839115 0:50:4:D1:81:A1 -> 0:4:76:8C:F5:4A type:0x800
len:0x334
100.10.10.76:1236 -> 10.24.4.190:1241 TCP TTL:63 TOS:0x0 ID:25374
IpLen:20 DgmLen:806 DF
***AP*** Seq: 0xB4005A13  Ack: 0x4DC5D6EA  Win: 0xC1E8  TcpLen: 20
16 03 00 00 4A 02 00 00 46 03 00 3F 41 2E A7 02  ....J...F..?A...
6F CF B7 CF 02 D7 DD 29 DB B5 9E 47 51 C0 C0 7D  o......)...GQ..}
96 1E CA 72 39 16 C3 74 EF E9 4A 20 72 EA A9 4E  ...r9..t..J r..N
32 F1 34 0B F4 0C 4B
```

.

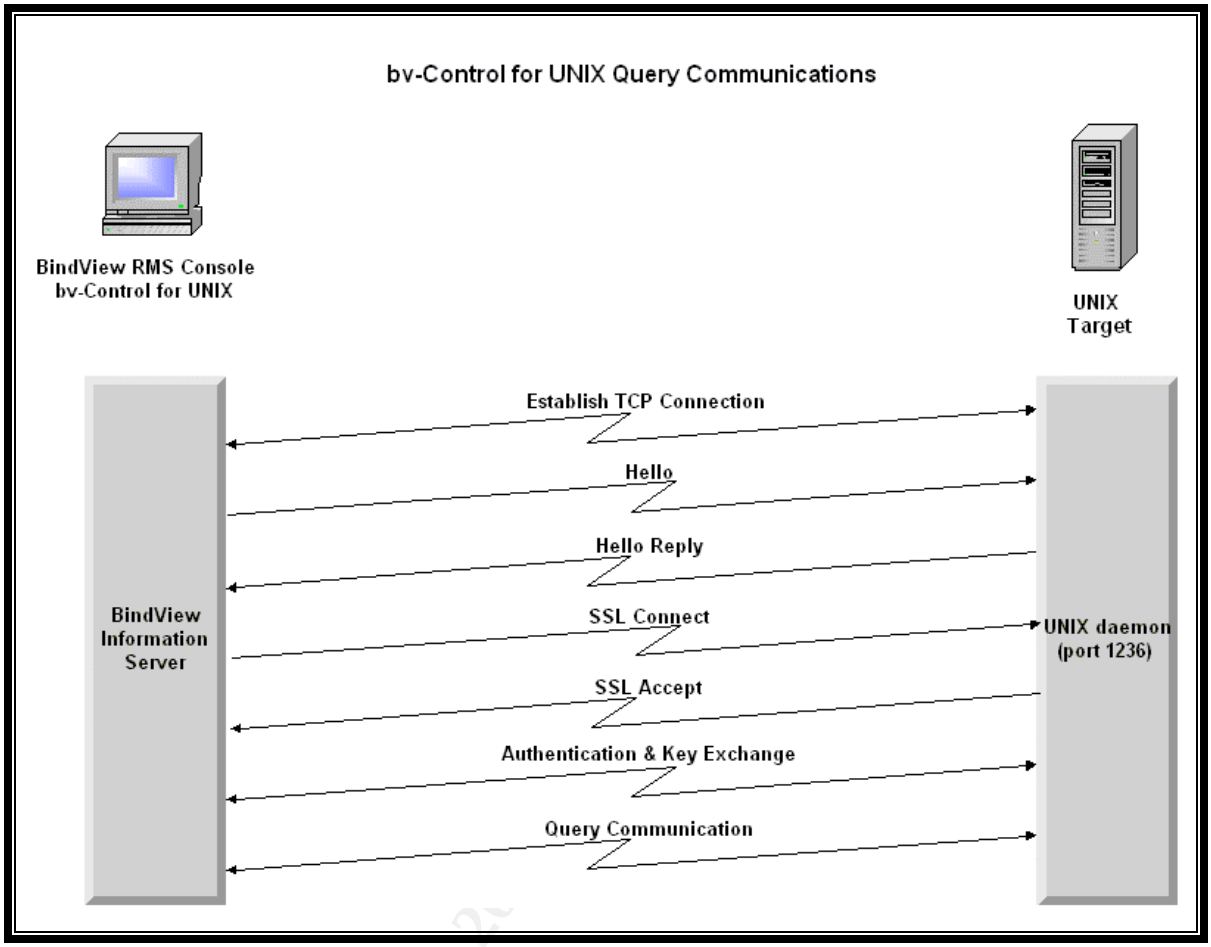
Encrypted data not shown for security purposes

.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Communication Session Ends

```
08/18-14:56:05.572008 0:50:4:D1:81:A1 -> 0:4:76:8C:F5:4A type:0x800
len:0x3C
100.10.10.76:1236 -> 10.24.4.190:1241 TCP TTL:63 TOS:0x0 ID:25392
IpLen:20 DgmLen:40 DF
***** ** Seq: 0xB4008070  Ack: 0x0  Win: 0xBD5C  TcpLen: 20
```



bv-Control for UNIX Query Communications[18]

[18] Walsh, Dan. "bv-Control for Unix Tech Training." 4 Dec. 2000. pg. 32.

References

"3.6.1 What is Diffie-Hellman?" http://www.rsasecurity.com/rsalabs/faq/3-6-1.html

"bv-Control® for UNIX®." URL: http://www.bindview.com/Products/VulnMgmt/AssesmentandSecurity/bv-Control_Unix.cfm

"bv-Control® for UNIX®." URL: http://www.bindview.com/resources/Datasheets/bvControl_UNIX_DS.pdf

"Chapter 1 – Snort™ Overview." URL: http://www.snort.org/docs/writing_rules/chap1.html#tth_sEc1.2

"Introduction to SSL." URL: http://developer.netscape.com/docs/manuals/security/sslin/contents.htm#1046261

"stateful inspection." URL: http://www.webopedia.com/TERM/S/stateful_inspection.html

"Stateful Inspection™ Firewall Technology." URL: http://www.bindview.com/resources/Datasheets/bvControl_UNIX_DS.pdf

Diffie, Whitfield and Martin Hellman. "New Directions in Cryptography." 3 June, 1976. URL: http://www.cs.jhu.edu/~rubin/courses/sp03/papers/diffie.hellman.pdf

Egevang, Kjeld and Paul Francis. "RFC 1631-The IP Network Address Translator (NAT)." May, 1994. URL: http://www.faqs.org/rfcs/rfc1631.html

"FireWall-1, The industry's only integrated solution for both network and application security." URL: http://www.checkpoint.com/products/protect/firewall-1.html

"FireWall-1, Network Address Translation." URL: http://www.checkpoint.com/products/protect/firewall-1_netaddress.html

"FireWall-1, What is a FireWall?" URL: http://www.checkpoint.com/products/protect/firewall-1_primer.html

Gokhale, Ravi. From a bv-Control for UNIX lecture. 31 Jul 2003.

Greenfield, David. "SSL and TLS." Network Magazine.com. 04 Dec 2002. URL: http://www.networkmagazine.com/shared/article/showArticle.jhtml;jsessionid=GSSF0IBPHZ1AQQSNDBCCKHQ?articleId=8703479&pgno=2

Onyszko, Tomasz. "Secure Socket Layer." Window Security Magazine. 19 Jul 2002. URL: *http://www.windowsecurity.com/articles/Secure_Socket_Layer.html*

Palmgren, Keith. "Diffie-Hellman Key Exchange – A Non-Mathematician's Explanation." 2001. *http://www.netip.com/articles/keith/diffie-helman.htm*

Rekhter, Yakov, Moskowitz, Robert G., Karrenberg, Daniel, Jan de Groot, Geert. "RFC 1597-Address Allocation for Private Internets." March, 1994. URL: *http://www.faqs.org/rfcs/rfc1597.html*

Rescorla, E. "RFC 2631 - Diffie-Hellman Key Agreement Method." June, 1999. URL: *http://www.ietf.org/rfc/rfc2631.txt*

Walsh, Dan. "bv-Control for Unix Tech Training." 4 Dec 2000.

Westrope, Alan. "Diffie-Hellman Key Exchange." 1998. URL: *http://www.nyx.net/~awestrop/crypt/dh.htm*