Nessus Credential Checks for Unix and Windows

June 15, 2011

(Revision 25)

Table of Contents

INTRODUCTION

This paper describes how to perform authenticated network scans with Tenable Network Security's **Nessus** vulnerability scanner. Authenticated network scans allow a remote network audit to obtain "host-based" data such as missing patches and operating system settings. Please share your comments and suggestions with us by emailing them to support@tenable.com.

Nessus leverages the ability to log into remote Unix hosts via Secure Shell (SSH). For Windows hosts, Nessus leverages a variety of Microsoft authentication technologies.

Note that Nessus also uses the Simple Network Management Protocol (SNMP) to make version and information queries to routers and switches. Although this is a form of "local checks", it is not covered in this document.

This document also makes extensive references to "Nessus", but the basic concepts are also true for Tenable's SecurityCenter.

STANDARDS AND CONVENTIONS

Throughout the documentation, filenames, daemons and executables are indicated with a **`courier bold`** font such as **`setup.exe`**.

Command line options and keywords are also printed with the **`courier bold`** font. Command line options may or may not include the command line prompt and output text from the results of the command. Often, the executed command will be **boldfaced** to indicate what the user typed. Below is an example running of the Unix **`pwd`** command:

```
# pwd
/home/test/
#
```

Important notes and considerations are highlighted with this symbol and grey text boxes.

Tips, examples and best practices are highlighted with this symbol and white on blue text.

OVERVIEW OF NESSUS CREDENTIAL CHECKS

Tenable's Nessus scanner is a very effective network vulnerability scanner with a comprehensive database of plugins that check for a large variety of vulnerabilities that could be remotely exploited. In addition to remote scanning, the Nessus scanner can also be used to scan for local exposures.

Purpose

External network vulnerability scanning is useful to obtain a snapshot in time of the network services offered and the vulnerabilities they may contain. However, it is only an external perspective. It is important to determine what local services are running and to identify security exposures from local attacks or configuration settings that could expose the system to external attacks that may not be detected from an external scan.

In a typical network vulnerability assessment, a remote scan is performed against the external points of presence and an onsite scan is performed from within the network. Neither of these scans can determine local exposures on the target system. Some of the information gained relies on the banner information displayed, which may be inconclusive or incorrect. By using secured credentials, the Nessus scanner can be granted local access to scan the target system without requiring an agent. This can facilitate scanning of a very large network to determine local exposures or compliance violations.

The most common security problem in an organization is that security patches are not applied in a timely manner. A Nessus credentialed scan can quickly determine which systems are out of date on patch installation. This is especially important when a new vulnerability is made public and executive management wants a quick answer regarding the impact to the organization.

Another major concern for organizations is to determine compliance with site policy, industry standards (such as the Center for Internet Security (CIS) benchmarks) or legislation (such as Sarbanes-Oxley (SOX), Gramm-Leach-Bliley (GLBA) or HIPAA). Organizations that accept credit card information must demonstrate compliance with the Payment Card Industry Data Security Standards (PCI DSS). There have been quite a few well-publicized cases where the credit card information for millions of customers was breached. This represents a significant financial loss to the banks responsible for covering the payments and heavy fines or loss of credit card acceptance capabilities by the breached merchant or processor.

Access Level

Credentialed scans can perform any operation that a local user can perform. The level of scanning is dependant on the privileges granted to the user account that Nessus is configured to use.

Non-privileged users with local access on Unix systems can determine basic security issues, such as patch levels or entries in the **`/etc/passwd`** file. For more comprehensive information, such as system configuration data or file permissions across the entire system, an account with "root" privileges is required.

Credentialed scans on Windows systems require that an administrator level account be used. Several bulletins and software updates by Microsoft have made reading the registry to determine software patch level unreliable without administrator privileges. Administrative access is required to perform direct reading of the file system. This allows Nessus to attach to a computer and perform direct file analysis to determine the true patch level of the systems being evaluated. On Windows XP Pro, this file access will only work with a local administrator account if the "Network access: Sharing and security model for local accounts" policy is changed to "Classic – local users authenticate as themselves".

Technologies Used

The challenge in running a credentialed scan is to automatically provide the privileged credentials to the scanner in a secure manner. It would certainly defeat the purpose of scanning for security exposures if doing so would open an even greater exposure! Nessus supports the use of several secure methods to solve this problem on both Unix and Windows platforms.

Unix Systems

On Unix systems, Nessus uses Secure Shell (SSH) protocol version 2 based programs (e.g., OpenSSH, Solaris SSH, etc.) for host-based checks. This mechanism encrypts the data in transit to protect it from being viewed by sniffer programs. Nessus supports three types of authentication methods for use with SSH: username and password, public/private keys and Kerberos.

Username and Password

Although supported, Tenable does not recommend using a username and password for authentication with SSH. Static passwords are subject to "man in the middle" and brute force attacks when they have been in use over a long period of time.

Public/Private Keys

Public Key Encryption, also referred to as asymmetric key encryption, provides a more secure authentication mechanism by the use of a public and private key pair. In asymmetric cryptography, the public key is used to encrypt data and the private key is used to decrypt it. The use of public and private keys is a more secure and flexible method for SSH authentication. Nessus supports both DSA and RSA key formats.

Kerberos

Kerberos, developed by MIT's Project Athena, is a client/server application that uses a symmetric key encryption protocol. In symmetric encryption, the key used to encrypt the data is the same as the key used to decrypt the data. Organizations deploy a KDC (Key Distribution Center) that contains all users and services that require Kerberos authentication. Users authenticate to Kerberos by requesting a TGT (Ticket Granting Ticket). Once a user is granted a TGT, it can be used to request service tickets from the KDC to be able to utilize other Kerberos based services. Kerberos uses the CBC (Cipher Block Chain) DES encryption protocol to encrypt all communications.

The Nessus implementation of Kerberos authentication for SSH supports the "aes-cbc" and "aes-ctr" encryption algorithms. An overview of how Nessus interacts with Kerberos is as follows:

- End-user gives the IP of the KDC
- **`nessusd`** asks **`sshd`** if it supports Kerberos authentication
- **`sshd`** says yes
- **`nessusd`** requests a Kerberos TGT, along with login and password
- Kerberos sends a ticket back to **`nessusd`**
- **`nessusd`** gives the ticket to **`sshd`**
- **`nessusd`** is logged in

Windows Systems

Nessus supports several different types of authentication methods for Windows-based systems. Each of these methods takes a username, password and domain name (sometimes optional for authentication).

LANMAN

The Lanman authentication method was prevalent on Windows NT and early Windows 2000 server deployments. It is not really used on newer Windows deployments, but is retained for backwards compatibility.

NTLM and NTLMv2

The NTLM authentication method, introduced with Windows NT, provided improved security over Lanman authentication. However, the enhanced version, NTLMv2, is cryptographically more secure than NTLM and is the default authentication method chosen by Nessus when attempting to log into a Windows server.

SMB Signing

SMB signing is a cryptographic checksum applied to all SMB traffic to and from a Windows server. Many system administrators enable this feature on their servers to ensure that remote users are 100% authenticated and part of a domain. It is automatically used by Nessus if it is required by the remote Windows server.

SPNEGO

The SPNEGO (Simple and Protected Negotiate) protocol provides Single Sign On (SSO) capability from a Windows client to a variety of protected resources via the users' Windows login credentials. Nessus supports use of SPNEGO with either NTLMSSP with LMv2 authentication or Kerberos and RC4 encryption.

Kerberos

Nessus also supports the use of Kerberos authentication in a Windows domain. To configure this, the IP address of the Kerberos Domain Controller (actually, the IP address of the Windows Active Directory Server) must be provided.

NTLMSSP (NT Lan Manager Security Support Provider) and LMv2

If an extended security scheme (such as Kerberos or SPNEGO) is not supported or fails, Nessus will attempt to log in via NTLMSSP/LMv2 authentication. If that fails, Nessus will then attempt to log in using NTLM authentication.

Windows Usernames, Passwords and Domains

The SMB domain field is optional and Nessus will be able to log on with domain credentials without this field. The username, password and optional domain refer to an account that the target machine is aware of. For example, given a username of "joesmith" and a password of "`my4x4mpl3`", a Windows server first looks for this username in the local system's list of users, and then determines if it is part of a domain in there.

The actual domain name is only required if an account name is different on the domain from that on the computer. It is entirely possible to have an "Administrator" account on a Windows server and within the domain. In this case, to log onto the local server, the username of "Administrator" is used with the password of that account. To log onto the domain, the "Administrator" username would also be used, but with the domain password and the name of the domain.

Regardless of credentials used, Nessus always attempts to log into a Windows server with the following combinations:

- "Administrator" without a password
- A random username and password to test Guest accounts
- No username or password to test null sessions

CREDENTIAL CHECKS ON UNIX PLATFORMS

The process described in this section enables you to perform local security checks on Unix based systems. The SSH daemon used in this example is OpenSSH. If you have a commercial variant of SSH, your procedure may be slightly different.

To enable local security checks, there are two basic methods that can be used:

1. Use of a SSH private/public key pair
2. User credentials and **sudo** access or credentials for **su** access

PREREQUISITES

Configuration Requirements for SSH

Nessus 4.x supports the blowfish-cbc, aesXXX-cbc (aes128, aes192 and aes256), 3des-cbc and aes-ctr algorithms.

Some commercial variants of SSH do not have support for the blowfish algorithm, possibly for export reasons. It is also possible to configure an SSH server to only accept certain types of encryption. Check your SSH server to ensure the correct algorithm is supported.

User Privileges

For maximum effectiveness, the SSH user must have the ability to run any command on the system. On Unix systems, this is known as "**root**" privileges. While it is possible to run some checks (such as patch levels) with non-privileged access, full compliance checks that audit system configuration and file permissions require root access. For this reason, it is strongly recommended that SSH keys be used instead of credentials when possible.

Configuration Requirements for Kerberos

If Kerberos is used, **sshd** must be configured with Kerberos support to verify the ticket with the KDC. Reverse DNS lookups must be properly configured for this to work. The Kerberos interaction method must be **gssapi-with-mic**.

ENABLING SSH LOCAL SECURITY CHECKS ON UNIX

This section is intended to provide a high-level procedure for enabling SSH between the systems involved in the Nessus credential checks. It is not intended to be an in-depth tutorial on SSH. It is assumed the reader has the prerequisite knowledge of Unix system commands.

Generating SSH Public and Private Keys

The first step is to generate a private/public key pair for the Nessus scanner to use. This key pair can be generated from any of your Unix systems, using any user account. However, it is important that the keys be owned by the defined Nessus user.

To generate the key pair, use **ssh-keygen** and save the key in a safe place. In the following example the keys are generated on a Red Hat ES 3 installation.

```
# ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/Users/test/.ssh/id_dsa):
       /home/test/Nessus/ssh_key
```

```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in
/home/test/Nessus/ssh_key.
Your public key has been saved in
/home/test/Nessus/ssh_key.pub.
The key fingerprint is:
06:4a:fd:76:ee:0f:d4:e6:4b:74:84:9a:99:e6:12:ea
#
```

Do not transfer the private key to any system other than the one running the Nessus server. When **ssh-keygen** asks you for a passphrase, enter a strong passphrase or hit the "Return" key twice (i.e., do not set any passphrase). If a passphrase is specified, it must be specified in the Policies -> Credentials -> SSH settings options in order for Nessus to use key-based authentication.

Nessus Windows users may wish to copy both keys to the main Nessus application directory on the system running Nessus (**C:\Program Files\Tenable\Nessus** by default), and then copy the public key to the target systems as needed. This makes it easier to manage the public and private key files.

Creating a User Account and Setting up the SSH Key

On every target system to be scanned using local security checks, create a new user account dedicated to Nessus. This user account must have exactly the same name on all systems. For this document, we will call the user "nessus", but you can use any name.

Once the account is created for the user, make sure that the account has no valid password set. On Linux systems, new user accounts are locked by default, unless an initial password was explicitly set. If you are using an account where a password had been set, use the "**passwd -l**" command to lock the account.

You must also create the directory under this new account's home directory to hold the public key. For this exercise, the directory will be **/home/nessus/.ssh**. An example for Linux systems is provided below:

```
# passwd -l nessus
# cd /home/nessus
# mkdir .ssh
#
```

For Solaris 10 systems, Sun has enhanced the "**passwd(1)**" command to distinguish between locked and non-login accounts. This is to ensure that a user account that has been locked may not be used to execute commands (e.g., cron jobs). Non-login accounts are used only to execute commands and do not support an interactive login session. These accounts have the "NP" token in the password field of **/etc/shadow**. To set a non-login account and create the SSH public key directory in Solaris 10, run the following commands:

```
# passwd -N nessus

# grep nessus /etc/shadow
```

```
nessus:NP:13579::::::
# cd /export/home/nessus
# mkdir .ssh
#
```

Now that the user account is created, you must transfer the key to the system, place it in the appropriate directory and set the correct permissions.

Example

From the system containing the keys, secure copy the public key to system that will be scanned for host checks as shown below. 192.1.1.44 is an example remote system that will be tested with the host-based checks.

```
# scp ssh_hey.pub root@192.1.1.44:/home/nessus/.ssh/authorized_keys
#
```

You can also copy the file from the system on which Nessus is installed using the secure FTP command, "**sftp**". Note that the file on the target system must be named "authorized_keys".

Return to the System Housing the Public Key

Set the permissions on both the **/home/nessus/.ssh** directory, as well as the **authorized_keys** file.

```
# chown -R nessus:nessus ~nessus/.ssh/
# chmod 0600 ~nessus/.ssh/authorized_keys
# chmod 0700 ~nessus/.ssh/
#
```

Repeat this process on all systems that will be tested for SSH checks (starting at "Creating a User Account and Setting up the SSH Key" above).

Test to make sure that the accounts and networks are configured correctly. Using the simple Unix command "**id**", from the Nessus scanner, run the following command:

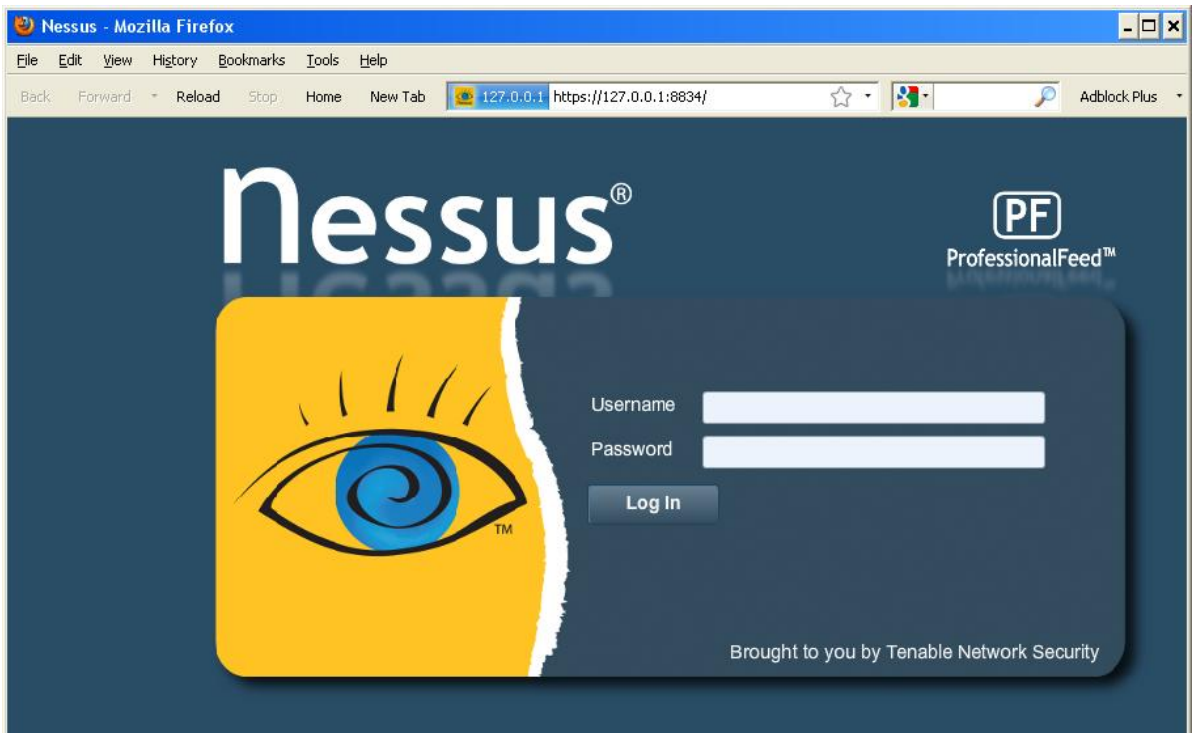
```
# ssh -i /home/test/nessus/ssh_key nessus@192.1.1.44 id
uid=252(nessus) gid=250(tns) groups=250(tns)
#
```

If it successfully returns information about the nessus user, the key exchange was successful.

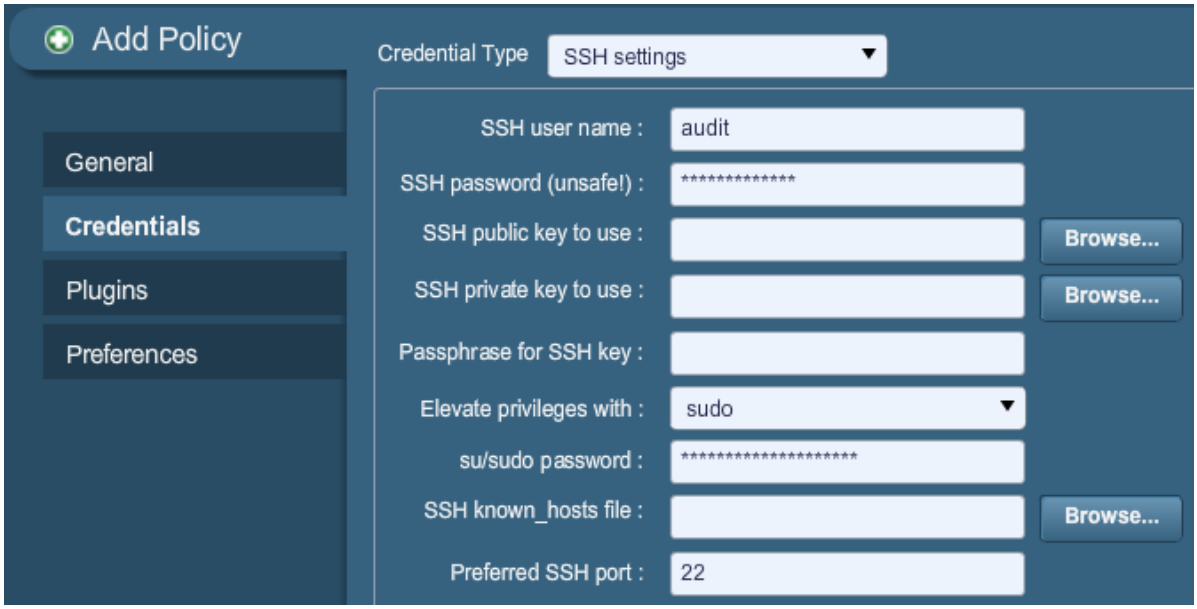
CONFIGURING NESSUS FOR SSH HOST-BASED CHECKS

Nessus User Interface

If you have not already done so, secure copy the private and public key files to the system that you will use to access the Nessus scanner.



Open a web browser and connect to the Nessus scanner user interface as seen above and click on the "Policies" tab. Create a new policy or edit an existing policy and select the "Credentials" tab on the left. Select "SSH settings" from the drop down menu at the top as shown below:



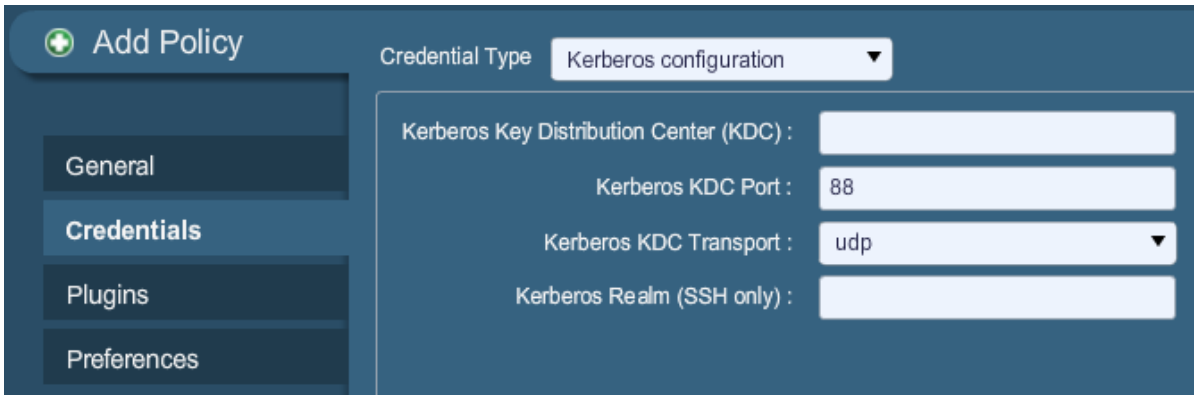
- For the item "SSH user name", enter the name of the account that is dedicated to Nessus on each of the scan target systems. It is set to "root" by default.
- If you are using a password for SSH, enter it in the "SSH password" box.

- If you are using SSH keys instead of a password (recommended), click on the "Select" button next to the box labeled "SSH public key to use" and locate the public key file on the local system.
- For the item "SSH private key to use" click on the "Select" button and locate the private key file (that is associated with the public key above) on the local system.
- If you are using a passphrase for the SSH key (optional), enter it in the box labeled "Passphrase for SSH key".
- Nessus and SecurityCenter users can additionally invoke "**su**" or "**sudo**" with the "Elevate privileges with" field and a separate password.
- If an SSH **known_hosts** file is available and provided as part of the scan policy in the "SSH known_hosts file" field, Nessus will only attempt to log into hosts in this file. This can ensure that the same username and password you are using to audit your known SSH servers is not used to attempt a log into a system that may not be under your control.

The most effective credentialed scans are those when the supplied credentials have "root" privileges. Since many sites do not permit a remote login as root, Nessus users can invoke "**su**" or "**sudo**" with a separate password for an account that has been set up to have "**su**" or "**sudo**" privileges.

An example screen capture of using "**sudo**" in conjunction with SSH keys follows. For this example, the user account is "audit", which has been added to the **/etc/sudoers** file on the system to be scanned. The password provided is the password for the "audit" account, not the root password. The SSH keys correspond with keys generated for the "audit" account:

If you are using Kerberos, you must configure a Nessus scanner to authenticate to a KDC. Select "Kerberos configuration" from the drop-down menu as shown below:



The default KDC port is "88" and the default transport protocol is "udp". The other value for transport is "tcp". Last, the Kerberos Realm name and IP address of the KDC are required.

Note that you must already have a Kerberos environment established to use this method of authentication.

At this point, click on "**Submit**" at the bottom of the window and the configuration will be complete. The new scan policy will be added to the list of managed scan policies.

Nessus Unix Command Line

Nessus support for host-based checks is available in Nessus 2.2.0 and later and requires that SSL support be compiled in. Run the "**`nessusd -d`**" command to verify that you have the correct version and SSL libraries as follows:

```
# nessusd -d
This is Nessus 4.4.1. [build M15078] for Linux 2.6.13-15-smp
compiled with gcc version 4.0.2 20050901 (prerelease) (SUSE Linux)
Current setup :
        flavor                          : suse10.0-x86
        nasl                            : 4.4.1
        libnessus                       : 4.4.1
        SSL support                     : enabled
        SSL is used for client / server communication
        Running as euid                 : 0
Magic hash: 49edd1433ffad7b87b446a4201faeedf  -
OpenSSL: OpenSSL 0.9.7g 11 Apr 2005
Include these infos in your bug reports
```

Using .nessus Files

Nessus has the ability to save configured scan policies, network targets and reports as a **`.nessus`** file. The above section "Nessus User Interface" describes creating a **`.nessus`** file that contains SSH credentials. For instructions on running a command line scan using the **`.nessus`** file, refer to the "Nessus User Guide" available at:

http://www.tenable.com/products/nessus/documentation

Using .nessusrc Files

If you are manually creating "**.nessusrc**" files, there are several parameters that can be configured to specify SSH authentication. An example of an unpopulated listing is shown below:

```
Use SSH to perform local security checks[entry]:SSH user name : =
Use SSH to perform local security checks[file]:SSH public key to use : =
Use SSH to perform local security checks[file]:SSH private key to use : =
Use SSH to perform local security checks[password]:Passphrase for SSH key
      : =
SSH settings[entry]:SSH user name : =
SSH settings[password]:SSH password (unsafe!) : =
SSH settings[file]:SSH public key to use : = no
SSH settings[file]:SSH private key to use : =
SSH settings[password]:Passphrase for SSH key : =
```

If you are using Kerberos, you must configure a Nessus scanner to authenticate to a KDC by entering the following information in the scanner **nessusrc** file:

```
Kerberos KDC port : 88
Kerberos KDC Transport : udp
Kerberos Realm (SSH Only) : myrealm
Kerberos Key Distribution Center (KDC): 192.168.20.66
```

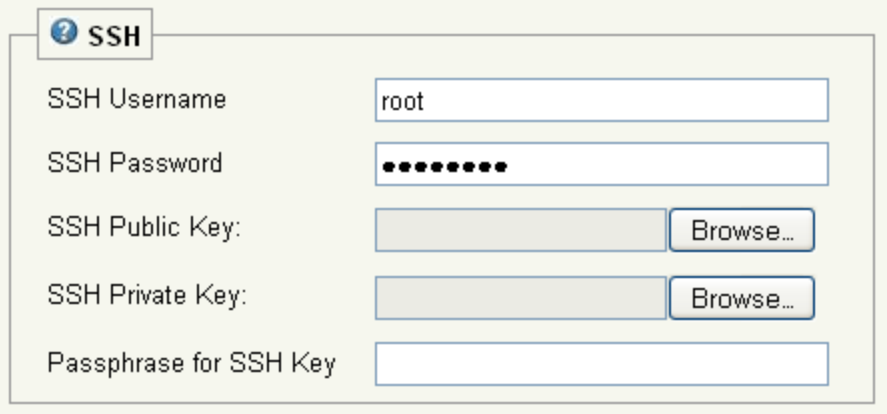
The default KDC port is "88" and the default transport protocol is "udp". The other value for transport is "tcp". Last, the Kerberos Realm name and IP address of the KDC are required.

Note that you must already have a Kerberos environment established to use this method of authentication.

Using SSH Credentials with the Tenable SecurityCenter

To use SSH credentials with SecurityCenter, upload the generated SSH public and private keys to the SecurityCenter console. Do not install them on the Nessus scanners directly since SecurityCenter downloads these credentials to the Nessus scanner when the scan is initiated.

The following is an example of a portion of the "Edit Scan Options" screen when editing the options of a policy. The last three fields are used to specify an account and specific public and private SSH keys to be used with testing. The SSH public key must be placed on every Unix host that will be tested for these "local checks".



SecurityCenter ships with several pre-defined vulnerability policies that have all of the "local checks" enabled for each individual OS. However, these policies must be copied and then have a specific SSH public/private key pair as well as a specific user account added so they can be used operationally.

The SSH public/private key pairs are managed by SecurityCenter and will be passed to each managed Nessus scanner.

Once these SSH public keys are installed on the desired Unix hosts and the private keys are installed under SecurityCenter, a trust relationship is created such that a user can log into each of the Unix hosts from the Nessus scanners. If the security of the Nessus scanners is compromised, new SSH public/private key pairs must be produced.

CREDENTIAL CHECKS ON WINDOWS PLATFORMS

PREREQUISITES

User Privileges

A very common mistake is to create a local account that does not have enough privileges to log on remotely and do anything useful. By default, Windows will assign new local accounts "Guest" privileges if they are logged into remotely. This prevents remote vulnerability audits from succeeding. Another common mistake is to increase the amount of access that the "Guest" users obtain. This reduces the security of your Windows server.

ENABLING WINDOWS LOGINS FOR LOCAL AND REMOTE AUDITS

The most important aspect about Windows credentials is that the account used to perform the checks should have privileges to access all required files and registry entries, and in many cases this means administrative privileges. If Nessus is not provided the credentials for an administrative account, at best it can be used to perform registry checks for the patches. While this is still a valid method to determine if a patch is installed, it is incompatible with some third party patch management tools that may neglect to set the key in the policy. If Nessus has administrative privileges, then it will actually check the version of the dynamic-link library (`.dll`) on the remote host, which is considerably more accurate.

Configuring a Local Account

To configure a stand-alone Windows server with credentials to be used that is not part of a domain, simply create a unique account as the administrator.

Make sure that the configuration of this account is not set with a typical default of "Guest only: local users authenticate as guest". Instead, switch this to "Classic: local users authenticate as themselves".

Configuring a Domain Account for Local Audits

To create a domain account for remote host-based auditing of a Windows server, the server must first be Windows Vista, Windows XP Pro, Windows 2003, Windows 2008 or Windows 7 and be part of a domain.

To configure the server to allow logins from a domain account, the "Classic" security model should be invoked. To do this, follow these steps:

1. Open "Group Policy" by clicking on "start", click "Run", type "gpedit.msc" and then click "OK".
2. Select Computer Configuration -> Windows Settings -> Security Settings -> Local Policies -> Security Options.
3. From the list of policies open "Network access: Sharing and security model for local accounts".
4. In this dialog, select "Classic – local users authenticate as themselves" and click "OK" to save this.

This will cause users local to the domain to authenticate as themselves, even though they are actually not really physically "local" on the particular server. Without doing this, all remote users, even real users in the domain, will actually authenticate as a "Guest" and will likely not have enough credentials to perform a remote audit.

Configuring Windows XP and 2003

When performing authenticated scans against Windows XP or 2003 systems, there are several configuration options that must be enabled:

1. The WMI service must be enabled on the target.
2. The Remote Registry service must be enabled on the target.
3. File & Printer Sharing must be enabled in the target's network configuration.
4. Ports 139 and 445 must be open between the Nessus scanner and the target.
5. An SMB account must be used that has local administrator rights on the target.

You may be required to change the Windows local security policies or they could block access or inherent permissions. A common policy that will affect credentialed scans is found under:

Administrative Tools -> Local Security Policy -> Security Settings -> Local Policies -> Security Options -> Network access: Sharing and security model for local accounts.

If this local security policy is set to something other than "Classic - local users authenticate as themselves", a compliance scan will not run successfully.

Configuring Windows 2008, Vista and 7

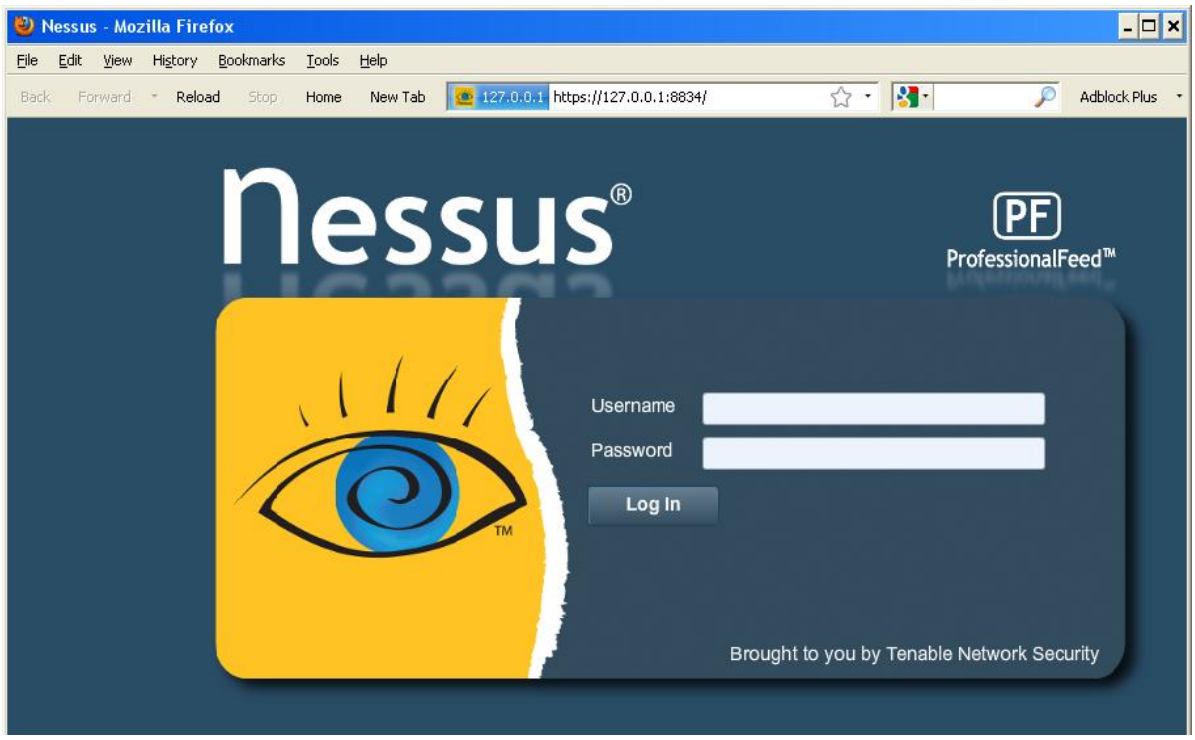
When performing authenticated scans against Windows 2008, Vista or 7 systems, there are several configuration options that must be enabled:

1. Under Windows Firewall -> Windows Firewall Settings, "File and Printer Sharing" must be enabled.
2. Using the **`gpedit.msc`** tool (via the "Run.." prompt), invoke the Group Policy Object Editor. Navigate to Local Computer Policy -> Administrative Templates -> Network -> Network Connections - > Windows Firewall -> Standard Profile -> Windows Firewall : Allow inbound file and printer exception, and enable it.
3. While in the Group Policy Object Editor, Local Computer Policy -> Administrative Templates -> Network -> Network Connections -> Prohibit use of Internet connection firewall on your DNS domain must be set to either "Disabled" or "Not Configured".
4. Windows User Account Control (UAC) must be disabled, or a specific registry setting must be changed to allow Nessus audits. To turn off UAC completely, open the Control Panel, select "User Accounts" and then set "Turn User Account Control" to off. Alternatively, you can add a new registry key named "LocalAccountTokenFilterPolicy" and set its value to "1". This key must be created in the registry at the following location: **`HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\system\LocalAccountTokenFilterPolicy`**. For more information on this registry setting, consult the MSDN 766945 KB.
5. The Remote Registry service must be enabled (it is disabled by default). It can be enabled for a one-time audit, or left enabled permanently if frequent audits are performed.

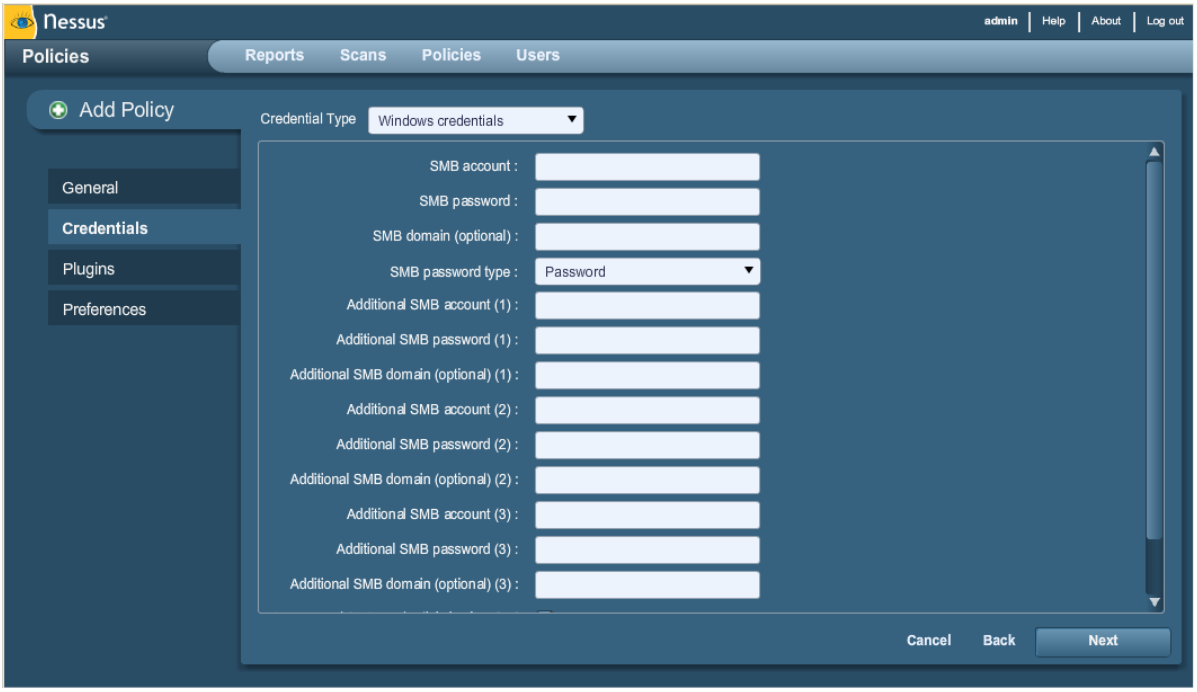
Nessus has the ability to enable and disable the Remote Registry service. For this to work, the target must have the Remote Registry service set to "Manual" and not "Disabled".

Configuring Nessus for Windows Logins

Nessus User Interface



Open a web browser and connect to the Nessus scanner user interface as seen above and click the "Policies" tab. Create a new policy or edit an existing policy and select the "Credentials" tab on the left. Select "Windows credentials" from the drop down menu at the top as shown below:



Specify the SMB account name, password and optional domain.

At this point, click on "**Submit**" at the bottom of the window and configuration will be complete. The new scan policy will be added to the list of managed scan policies.

Nessus Unix Command Line

Using .nessus Files

Nessus has the ability to save configured scan policies, network targets and reports as a **`.nessus`** file. The above section "Nessus User Interface" describes creating a **`.nessus`** file that contains Windows credentials. For instructions on running a command line scan using the **`.nessus`** file, refer to the "Nessus User Guide" available at:

http://www.tenable.com/products/nessus/documentation

Using .nessusrc Files

If you are manually building a "**`.nessusrc`**" file, there are three entries that allow for the configuration of the username, password and optional domain as shown below:

```
Login configurations[entry]:SMB account : =
Login configurations[password]:SMB password : =
Login configurations[entry]:SMB domain (optional) : =
```

DETECTING WHEN CREDENTIALS FAIL

If you are using Nessus to perform credentialed audits of Unix or Windows systems, analyzing the results to determine if you had the correct passwords and SSH keys can be difficult. Nessus users can now easily detect if their credentials are not working. Tenable has added Nessus plugin #21745 to the "Settings" plugins family.

This plugin detects if either SSH or Windows credentials did not allow the scan to log into the remote host. When a login is successful, this plugin does not produce a result. The following is an example report that was produced from trying to log into a remote machine with the incorrect username or password with Nessus:

TROUBLESHOOTING

Q. How do we know if the local scan is working?

A. Unless you have a 100% patched server, any local scan will likely return some sort of patch information. Depending on the operating system, it will also return a variety of information audits.

It may also be useful to take "Nessus" out of the equation and test to make sure that the accounts and networks are configured correctly. Using the simple Unix command **id**, from the Nessus scanner, run the following command:

```
# ssh -i /home/test/nessus/ssh_key nessus@192.1.1.44 id
#
```

Make sure to use the IP address of the system that the trust relationship is configured with as well as the user account (in this case user "nessus"). If the command succeeds, you will see the results of the **id** command as if it were run on your remote system.

On Unix audits the **ssh_get_info.nasl** script will report if the authentication was successful. If SSH logins are not working, you can increase the "report_verbosity" setting of your Nessus scan to "Verbose". This will show any error or diagnostic messages while this particular script is running.

For Windows audits, the **smb_login.nasl** and **smb_registry_access.nasl** scripts indicate if the login and password provided during the scan worked and if it was possible to read the remote registry. The **smb_registry_full_access.nasl** warns only if it was not possible to fully read the registry. Looking at the results of host-based checks for audits of a Windows server will show how the credentials worked.

In addition, the **hostlevel_check_failed.nasl** script detects if either SSH or Windows credentials did not allow the scan to log into the remote host.

Q. How do we know if the local scan is not working?

A. On Windows systems, login failure events will be generated at the server. If a domain controller is in use, the login failure events will be located there as well.

On Unix systems, login failures will be present in the system logs (such as **/var/log/messages**) unless a remote Kerberos controller is in use.

In addition, the **hostlevel_check_failed.nasl** script detects if either SSH or Windows credentials did not allow the scan to log into the remote host.

Q. What else can go wrong with my host checks?

A. There are many things that can block access. Some to consider include:

- Network firewalls that filter port 22 for SSH on Unix or port 445 for Windows
- Host-based firewalls that block connections to the mentioned ports
- On Unix systems, administrators that move SSH to ports other than 22

- Some host and network intrusion prevention systems prevent remote access
- The machine you are scanning is not a Unix or Windows server and could be a printer, router, fax machine or video display device

Q. I am testing SSH connections from the shell prompt of scan target hosts to the Nessus system to ensure proper connectivity. I find it experiences a delay as it connects, why?

A. This is most likely because the system is performing a DNS lookup when DNS is misconfigured. If your site uses DNS, contact your DNS administrator to address configuration issues. Some issues that could cause problems include missing reverse lookup zones. To test DNS lookups, perform the following:

```
# host IP_ADRR_OF_NESSUS_SERVER
```

If you have "dig" installed, you can also check with:

```
# dig -x IP_ADRR_OF_NESSUS_SERVER
```

If your site does not use DNS, the following steps will bypass the attempt to perform DNS lookups.

1. Edit the `/etc/nsswitch.conf` file so that the "hosts:" lines reads "hosts: files" Note: This may not be applicable to all OpenSSH releases.

2. Add the IP/name of the server running Nessus to the system's `/etc/hosts` file.

3. Configure the remote OpenSSH server to **not** perform DNS lookups on a host by setting both:

   - "UseDNS no" in the `sshd_config` file (for release 3.8), the default value is yes
   - "VerifyReverseMapping no"

SECURING YOUR SCANNER

WHY SHOULD I SECURE MY SCANNER?

If you configure a Nessus scanner to use credentials to log into a Unix or Windows server, your system will have credentials that could be leveraged by a malicious user. To prevent this, you must not only practice good security for the operating system your scanner is running on, but you must also be aware how an adversary can trick the scanner into disclosing security information.

WHAT DOES IT MEAN TO LOCK DOWN A SCANNER?

The ideal Nessus scanner would be driven entirely from a system console and not accept any network connections from any remote host. Such a system will be physically secured such that only authorized people are allowed access to it. This server could further be restricted with an external firewall or switch that only allows it to scan specific networks. Do not install personal firewall software directly on the Nessus scanner system. Remember that Nessus can be configured to only scan specific networks.

This type of scanner is not that useful. Consider allowing remote network access to the server. Nessus supports HTTP connections to port 8834 by default. A system firewall can be configured to only accept connections on port 8834 from valid Nessus clients.

If the box is to be administrated or operated remotely, secure remote access can also be used. On Unix, the Secure Shell (SSH) protocol can be used. Keep the SSH daemon up to date, use strong passwords and/or use stronger authentication techniques. On Windows servers, remote Terminal Services can be used to provide command and control over the services for Nessus Windows. In both cases, keep the system up to date and do not run unneeded network services. Please refer to the Center for Internet Security (CIS) benchmarks for guidance on hardening systems.

Secure Implementation of Unix SSH Audits

Never use SSH passwords to perform remote scans. If you are scanning a network, then all an adversary or malicious user would need to do is run a modified SSH daemon and record the attempted username and password. Even if you are using a unique username and password combination for each host, the use of static passwords is still vulnerable to exploitation.

If you are auditing one server with a known username and password for logging in over SSH, there is less chance that an adversary can use this against you. However, be sure to secure the configuration of the scan, because the username and password will be stored there in cleartext.

Secure Windows Audits

If the option "Only use NTLMv2" is disabled, then it is theoretically possible to trick Nessus into attempting to log into a Windows server with domain credentials via the NTLM version 1 protocol. This provides the remote attacker with the ability to use a "hash" obtained from Nessus. This "hash" can be potentially cracked to reveal a username or password. It may also be used to directly log into other servers. Force Nessus to use NTLMv2 by enabling the "Only use NTLMv2" setting at scan time. This prevents a hostile Windows server from using NTLM and receiving a "hash".

NTLMv2 can make use of "SMB Signing". Ensure that "SMB Signing" is enabled on all of your Windows servers to prevent any server that obtains a "hash" from a Nessus scan to reuse it. In addition, make sure you enforce a policy that mandates the use of strong passwords that cannot be easily broken via dictionary attacks from tools like John the Ripper and L0phtCrack.

Note that there have been many different types of attacks against Windows security to illicit "hashes" from computers for re-use in attacking servers. "SMB Signing" adds a layer of security to prevent these man-in-the-middle attacks.

FOR FURTHER INFORMATION

Tenable has produced a variety of other documents detailing Nessus' deployment, configuration, user operation and overall testing. These are listed here:

- **Nessus Installation Guide** – step by step walk through of installation
- **Nessus User Guide** – how to configure and operate the Nessus User Interface

- **Nessus Compliance Checks** – high-level guide to understanding and running compliance checks using Nessus and SecurityCenter
- **Nessus Compliance Checks Reference** – comprehensive guide to Nessus Compliance Check syntax
- **Nessus v2 File Format** – describes the structure for the `.nessus` file format, which was introduced with Nessus 3.2 and NessusClient 3.2
- **Nessus XML-RPC Protocol Specification** – describes the XML-RPC protocol and interface in Nessus
- **Real-Time Compliance Monitoring** – outlines how Tenable's solutions can be used to assist in meeting many different types of government and financial regulations

Please feel free to contact us at support@tenable.com, sales@tenable.com or visit our web site at http://www.tenable.com/.

ABOUT TENABLE NETWORK SECURITY

Tenable Network Security, the leader in Unified Security Monitoring, is the source of the Nessus vulnerability scanner and the creator of enterprise-class, agentless solutions for the continuous monitoring of vulnerabilities, configuration weaknesses, data leakage, log management and compromise detection to help ensure network security and FDCC, FISMA, SANS CAG and PCI compliance. Tenable's award-winning products are utilized by many Global 2000 organizations and Government agencies to proactively minimize network risk. For more information, please visit http://www.tenable.com.

Tenable Network Security, Inc.
7063 Columbia Gateway Drive
Suite 100
Columbia, MD 21046
410.872.0555
www.tenable.com