



**IVE Platform**

# **J.E.D.I. Solution Guide**

*Release 5.0*

**Juniper Networks, Inc.**

1194 North Mathilda Avenue  
Sunnyvale, CA 94089

USA

408-745-2000

**[www.juniper.net](http://www.juniper.net)**

Part Number: 50A053105PB

Juniper Networks, the Juniper Networks logo, NetScreen, NetScreen Technologies, the NetScreen logo, NetScreen-Global Pro, ScreenOS, and GigaScreen are registered trademarks of Juniper Networks, Inc. in the United States and other countries.

Juniper Networks, the Juniper Networks logo, NetScreen, NetScreen Technologies, Neoteris-Secure Access, Neoteris-Secure Meeting, NetScreen-SA 1000, NetScreen-SA 3000, NetScreen-SA 5000, IVE, GigaScreen, and the NetScreen logo are registered trademarks of Juniper Networks, Inc. NetScreen-5GT, NetScreen-5XP, NetScreen-5XT, NetsukePart Number: 50A053105PBen-25, NetScreen-50, NetScreen-100, NetScreen-204, NetScreen-208, NetScreen-500, NetScreen-5200, NetScreen-5400, NetScreen-Global PRO, NetScreen-Global PRO Express, NetScreen-Remote Security Client, NetScreen-Remote VPN Client, NetScreen-IDP 10, NetScreen-IDP 100, NetScreen-IDP 500, GigaScreen ASIC, GigaScreen-II ASIC, and NetScreen ScreenOS are trademarks of Juniper Networks, Inc. All other trademarks and registered trademarks are the property of their respective companies.

Copyright © 2001 D. J. Bernstein. Copyright © 1985-2003 by the Massachusetts Institute of Technology. All rights reserved. Copyright © 2000 by Zero-Knowledge Systems, Inc. Copyright © 2001, Dr. Brian Gladman <brg@gladman.uk.net>, Worcester, UK. All rights reserved. Copyright © 1989, 1991 Free Software Foundation, Inc. Copyright © 1989, 1991, 1992 by Carnegie Mellon University. Derivative Work - 1996, 1998-2000. Copyright © 1996, 1998-2000 The Regents of the University of California. All Rights Reserved. Copyright © 1999-2001 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted. Copyright © 1995 Tatu Ylonen <ylo@cs.hut.fi>, Espoo, Finland. All rights reserved. Copyright © 1986 Gary S. Brown. Copyright © 1998 CORE SDI S.A., Buenos Aires, Argentina. Copyright © 1995, 1996 by David Mazieres <dm@lcs.mit.edu>. Copyright © 1998-2002. The OpenSSL Project. All rights reserved. Copyright © 1989-2001, Larry Wall. All rights reserved. Copyright © 1989, 1991 Free Software Foundation, Inc. Copyright © 1996-2002 Andy Wardley. All Rights Reserved. Copyright © 1998-2002. Canon Research Centre Europe Ltd. Copyright © 1995-1998. Jean-loup Gailly and Mark Adler.

Juniper Networks NetScreen-SA, NetScreen-SA FIPS, and NetScreen-SM Administration, Release 5.0

Copyright © 2005, Juniper Networks, Inc.  
All rights reserved. Printed in USA.

Writers: Claudette Hobbart, Paul Battaglia

Revision History

31 May 2005

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

# Table of Contents

<b>Chapter 1</b>	<b>J.E.D.I. overview</b>	<b>1</b>
<b>Chapter 2</b>	<b>Using the IVE Web console to implement endpoint solutions</b>	<b>3</b>
	Creating and implementing Host Checker policies .....	3
	Securing managed versus unmanaged endpoints using remediation .....	5
	Using Host Checker remediation .....	5
	Host Checker configuration .....	6
	Realm and role configuration .....	7
	Securing endpoints using generic host checks.....	8
	Process Checking.....	8
	File Version Checking .....	9
	Port Checking.....	10
	Registry Setting Checking.....	10
	Securing endpoints using specific checks.....	11
	Screen Saver Check .....	12
	Microsoft Hotfix Installation Check.....	12
	Windows Update Status Verification.....	13
	Network Bridge is Disabled.....	13
	IP Forwarding is Disabled.....	13
	Internet Connection Sharing Disabled .....	14
	Windows XP Firewall.....	14
	McAfee VirusScan Enterprise 7.1.0.....	14
	Securing endpoints using Sygate On-Demand.....	15
	Securing endpoints using Sygate Security Agent .....	15
	Sygate Management Server Configuration .....	15
	Sygate Security Agent Configuration.....	15
	Securing endpoints using TrendMicro OfficeScan .....	16
	Trend Micro Office Scan Version Check.....	16
	Trend Micro OfficeScan Realtime Scan.....	17
	Securing endpoints using Whole Security Confidence Online.....	17
	Securing endpoints that use the Google Desktop Search .....	18
	Host Checker configuration .....	18
<b>Chapter 3</b>	<b>Using J.E.D.I. interfaces to implement endpoint solutions</b>	<b>21</b>
	Choosing which interface to use.....	22
	Using the Host Check Client Interface .....	22
	Signing your custom DLL.....	23
	Deploying and maintaining your custom DLL.....	24
	NetScreen Host Checker (NHC) API .....	24
	Using the Host Check Server Integration Interface .....	25
	Deploying third party applications through Host Checker .....	25
	Creating your interface DLL.....	30



## Chapter 1

# J.E.D.I. overview

Juniper Networks developed the Juniper Endpoint Defense Initiative (J.E.D.I.) to provide a comprehensive solution to assess the trust worthiness of IVE endpoints. Using J.E.D.I. components, you can secure the systems of users inside and outside your network before allowing them to connect to your IVE appliance.

J.E.D.I. provides two major methods for implementing first-class endpoint defense solutions:

- **IVE Web console**—Using standard options in any IVE appliance’s Web console, you can configure Host Checker running on Windows clients to call third-party endpoint security products such as Sygate Security Agent, ZoneAlarm Pro, Zone Labs Integrity, McAfee Desktop Firewall, and InfoExpress CyberGatekeeper Agent and examine whether the products are running as configured. For clients running Windows, Macintosh, or Linux, you can also use options in the Web console to check for the application fingerprints such as processes, files, and ports and allow or deny access to the IVE appliance and its resources based on the presence or absence of those fingerprints. On Windows clients, you can also check registry entries. For more information, see “Using the IVE Web console to implement endpoint solutions” on page 3.
- **J.E.D.I. APIs (Windows only)**—Using the J.E.D.I. APIs (also called Host Checker Interfaces), you create custom solutions for Windows clients that integrate third-party products into the IVE products. You can use the Host Check Client Interface to run your own DLLs through Host Checker or use the Host Check Server Integration Interface to tightly integrate your own DLLs and corresponding files into IVE appliances. For more information, see “Using J.E.D.I. interfaces to implement endpoint solutions” on page 21.

For descriptions of the Host Checker and Cache Cleaner components included in J.E.D.I., see the *NetScreen Instant Virtual Extranet Platform Administration Guide*.

The J.E.D.I. features provide the following benefits:

**Table 1: J.E.D.I Features and Benefits**

Feature	Benefit
Port Checks	Requires or restricts the specified network ports to control which network connections a client can generate during a session
Process Checks	Requires or restricts the specified processes and validates the process image using an MD5 checksum to control the software that is active during a session

**Table 1: J.E.D.I Features and Benefits (Continued)**

<b>Feature</b>	<b>Benefit</b>
File Checks	Requires or restricts the specified files and validates the file binary using an MD5 checksum to identify corporate PC images and disallow access to untrusted or potentially malicious machines
Registry Setting Checks	Requires or restricts the specified registry settings to identify corporate PC images, system configurations, and installed software and application settings to disallow access to untrusted or potentially malicious machines.
Age and Version Checks	Configurable options for the File Checks and Registry Setting Checks features allow you to verify the age of files such as virus clients and signatures, or the version number of applications using registry settings
Host Check Client Interface	Enables enterprises to enforce an endpoint trust policy for managed PCs that have personal firewalls, antivirus client, or other installed security clients and quarantine non-compliant hosts
Host Check Server Integration Interface	Enables enterprises to deliver and update third party security agents from the IVE, reducing the amount of public facing infrastructure, providing consolidated, standards-based reporting of endpoint security events, and enabling policy-based remediation of non-compliant clients

## Chapter 2

# Using the IVE Web console to implement endpoint solutions

This section includes the following information:

- “Creating and implementing Host Checker policies” on page 3
- “Securing managed versus unmanaged endpoints using remediation” on page 5
- “Securing endpoints using generic host checks” on page 8
- “Securing endpoints using specific checks” on page 11
- “Securing endpoints using Sygate On-Demand” on page 15
- “Securing endpoints using Sygate Security Agent” on page 15
- “Securing endpoints using TrendMicro OfficeScan” on page 16
- “Securing endpoints using Whole Security Confidence Online” on page 17
- “Securing endpoints that use the Google Desktop Search” on page 18

---

## Creating and implementing Host Checker policies

This section describes the high-level steps required to configure Host Checker through the IVE Web console. For more detailed implementation instructions, see the *NetScreen Instant Virtual Extranet Platform Administration Guide*.

To configure Host Checker, you must perform these tasks:

1. **Create global policies specifying the methods or rules that you want to enforce**—*Methods* check for third-party endpoint security products such as Sygate Security Agent, ZoneAlarm Pro, Zone Labs Integrity, McAfee Desktop Firewall, and InfoExpress CyberGatekeeper Agent. *Rules* specify requirements that clients must meet, such as the existence or absence of certain processes, registry values, or files on their systems. Use settings in the **System > Configuration > Security > Host Checker** page of the Web console to create global policies.

2. Determine at which levels within the IVE framework you want to enforce the policies:
  - To enforce Host Checker policies when the user first accesses the IVE, implement the policies at the realm level by using the **Users|Administrators > Authentication > Select Realm > Authentication Policy > Host Checker** page of the Web console.
  - To allow or deny users access to roles based on their compliance with Host Checker policies, implement the policies at the role level by using the **Users|Administrators > Roles|Delegation > Select Role > General > Restrictions > Host Checker** page of the Web console.
  - To map users to roles based on their compliance with Host Checker policies, use custom expressions in the **Users|Administrators > Authentication > Select Realm > Role Mapping** page of the Web console.
  - To allow or deny users access to individual resources based on their compliance with Host Checker policies, use conditions in the **Resource Policies > Select Resource > Select Policy > Detailed Rules > Select|CreateRule** page of the Web console.
3. For Windows clients, determine whether you need to use a pre-authentication access tunnel between the clients and policy server(s) or resources. For instructions, see “Specify Host Checker pre-authentication access tunnel definitions” on page 28.
4. Specify how users can access the Host Checker client. Host Checker enforces the policies you define using a client-side agent.
  - To enable automatic installation of the Host Checker client on all platforms, use the **Users|Administrators > Authentication > Select Realm > Host Checker** page of the Web console.
  - To download the Host Checker installer and manually install it on your Windows users’ systems, use the **Maintenance > System > Installers** page of the Web console.
5. Determine whether you want to create client-side logs. If you enable client-side logging through the **System > Configuration > Security > Client-side Logs** page of the Web console, the IVE appliance creates log files on your users’ systems and writes to the file whenever Host Checker runs.



**NOTE:** To provision client access accurately, use multiple policies to verify compliance at a detailed level. For example, you might use one policy to verify virus checking, another policy to verify spyware checking, and a third policy to verify adware checking.

---

## Securing managed versus unmanaged endpoints using remediation

You can use Host Checker policies to distinguish between a managed and unmanaged computer, and grant access accordingly. When defining a Host Checker policy, you can also specify remediation actions that you want Host Checker to take if a user's computer does not meet the requirements of the policy. For example, you can configure Host Checker to display a remediation page to the user. This page can contain your specific instructions and links to resources to help the user bring his computer into compliance with the Host Checker policy requirements.

For example, you can grant access levels based on whether the user's computer passes all of your Host Checker policies, some of them, or none of them:

**Table 2: Using Host Checker to set access for managed and unmanaged hosts**

Type of Endpoint	Host Checker Status	Access Level
Managed	Passed all Host Checker policies	Full access
Managed	Passed Host Checker policies that determine whether this is a managed host, but failed other Host Checker policies	Remediation actions applied or limited access
Unmanaged	Failed all Host Checker policies that determine whether this is a managed host	Force access through the Virtual Desktop

### Using Host Checker remediation

When defining a Host Checker policy, you can specify remediation actions that you want Host Checker to take if a user's computer does not meet the requirements of the policy.

For example, suppose a corporate policy requires that all users must have the latest signature files for a certain antivirus program before allowing access, and that a user tries to connect but his files are not up to date. You can configure Host Checker to display a remediation page to the user which contains your specific instructions and links to resources to help the user bring his computer into compliance with the Host Checker policy requirements. After the user updates his antivirus signature files and signs in again, the user has full access. Otherwise, you can grant access only via a Virtual Desktop.

The user may see a remediation page that contains instructions and a link such as:

#### **Your computer's security is unsatisfactory**

Your computer does not meet the following security requirements. Please follow the instructions below to fix these problems. When you are done click **Try Again**. If you choose to **Continue** without fixing these problems, you may not have access to all of your intranet servers.

#### **1. TrendMicro**

Instructions: You do not have the latest signature files. **Click here to download the latest signature files.**

## 2. ManagedPC

Instructions: Please sign into the IVE again from the Virtual Desktop.

For each Host Checker policy, you can configure two types of remediation actions:

- **User-driven**—Using custom instructions, you can inform the user about the failed policy and how to make his computer conform. The user must take action to successfully re-evaluate the failed policy.
- **Automatic (system-driven)**—You can configure Host Checker to automatically remediate the user’s computer. For example, you can kill processes, delete files, or launch another policy. On Windows, you can also call the `HCIF_Module.remediate()` function. (See “`HCIF_Module.remediate()` function (version 2 only)” on page 33.) Host Checker does not inform users when performing automatic actions. You could, however, include information in your custom instructions about the automatic actions.

For each policy, you can enable any or all of the following remediation actions to occur on Windows, Macintosh, or Linux client computers:

- **Enable Custom Instructions (user-driven)**—You can display instructions on the Host Checker remediation page to the user, and include links to resources such as policy servers or Web sites.
- **Evaluate other policies (automatic)**—You can specify one or more alternate policies that you want Host Checker to evaluate if the current policy requirements are not met. For example, if a user attempts to access the IVE from an outside client computer such as a kiosk, you can enable this option to evaluate an alternate policy that requires the user to access the IVE in a Sygate Virtual Desktop environment.
- **Kill Process (automatic)**—You can kill processes if the client computer does not meet the current policy requirements.
- **Delete Files (automatic)**—You can delete files if the client computer does not meet the current policy requirements.

### **Host Checker configuration**

To configure a Host Checker policy with remediation actions:

1. In the **System > Configuration > Security > Host Checker** page of the Web console, create the Host Checker policy.
2. Under **Remediation** on the **Host Checker Policy** page, select the remediation actions for the policy.

## Realm and role configuration

In the **Users > Authentication > General** page of the Web console, configure the IVE with a realm that allows the user to select a role. In the **Users > Roles** page, configure the following roles on the realm with Host Checker policies:

- **Full access role**—On this role, enforce all Host Checker policies including one or more that determine whether the endpoint is a managed host, such as requiring that all users must have the latest signature files for a certain antivirus program. You can also enforce other Host Checker policies you want to require for full access.
- **Partial access role**—On this role, enforce a Host Checker policy that determines whether the endpoint is a managed host. You can also enforce a subset of the other Host Checker policies that evaluate the endpoint for partial access in case the endpoint does not pass the Host Checker policies for full access.
- **Unmanaged access role**—On this role, enforce a Host Checker policy that determines whether the endpoint is a managed host. If the endpoint doesn't pass this policy, you can display a remediation page and instruct the user to run a Virtual Desktop and then sign in again.

In the role mappings for the realm (**Users > Authentication > Role Mapping** page), add the roles you want to be available for users and select the **User must select from among assigned roles** option.

In each role's Host Checker configuration on the **Users > Roles > SelectRole > General > Restrictions > Host Checker** page, select the **Allow users whose workstations meet the requirements specified by these Host Checker policies** option. Select the policies you want to enforce for the role.

## Securing endpoints using generic host checks

This section contains information about using Host Checker to perform generic host checks:

- “Process Checking” on page 8
- “File Version Checking” on page 9
- “Port Checking” on page 10
- “Registry Setting Checking” on page 10

### Process Checking

You can use a Process Check rule to check whether certain processes are running on a host. You can require a process to run, or you can deny access if a process is running. You can also specify a MD5 checksum value to verify that the process is the correct version.

To configure a Host Checker policy with a Process Check rule:

1. In the Web console, choose **System > Configuration > Security > Host Checker**.
2. Under **Policies**, click **New**. Enter a name in the **Policy Name** field and then click **Continue**.
3. Click the tab that corresponds to the operating system for which you want to specify Host Checker options—**Windows**, **Mac**, or **Linux**. In the same policy, you can specify different processes on each operating system.
4. Under **Rule Settings**, select **Attribute Check: Process**
5. Enter a rule name and then specify settings for the Process Check rule:

**Table 3: Attribute Check: Process**

For this setting:	Specify:
<b>Process Name:</b>	< name of a process (executable file) > <sup>a</sup>
<b>Access Setting:</b>	Required or Deny
<b>MD5 checksum:</b>	< calculate the appropriate MD5 checksum value (optional) >

a. You can use a wildcard character to specify the process name.



**NOTE:** Specify the MD5 checksum value of each executable file to which you want the policy to apply. For example, an executable may have different MD5 checksum values on a desktop, laptop, or different operating systems.

6. Click **Save Changes**.

## File Version Checking

You can use a File Check rule to verify that an application is the correct version by checking the application's MD5 checksum value. For example, you can verify that the user's antivirus software is up to date, or that a required application is the correct version.

To configure a Host Checker policy with a File Check rule:

1. In the Web console, choose **System > Configuration > Security > Host Checker**.
2. Under **Policies**, click **New**. Enter a name in the **Policy Name** field and then click **Continue**.
3. Click the tab that corresponds to the operating system for which you want to specify Host Checker options—**Windows**, **Mac**, or **Linux**. In the same policy, you can specify different files on each operating system.
4. Under **Rule Settings**, select **Attribute Check: File**
5. Enter a rule name and then specify settings for the File Check rule:

**Table 4: Attribute Check: Files**

For this setting:	Specify:
<b>File Name:</b>	< name of a file (any file type) > <sup>a</sup>
<b>Access Setting:</b>	Required or Deny
<b>File modified less than</b>	< maximum age (in days) for a file >
<b>MD5 checksum:</b>	< calculate the appropriate MD5 checksum value >

a. You can use a wildcard character to specify the file name. You can also use an environment variable to specify the directory path to the file. (You cannot use a wildcard character in the directory path.) Enclose the variable between the < % and % > characters.



**NOTE:** To check the age of virus signatures, specify the path to a file in the **File Name** field whose timestamp indicates when the antivirus software last updated the virus signatures, such as a virus signature database or log file that updates each time the database updates. For example, if you use TrendMicro, you may specify:

C:\Program Files\Trend Micro\OfficeScan Client\TmUpdate.ini.

6. Click **Save Changes**.

## Port Checking

Ports check rules control the network connections that a client can generate during a session. You can use a Ports Check rule to require a client machine to have certain ports open or closed in order for the user to access the IVE. For example, you can use a Ports Check rule to verify that there are no listening ports on any interface on the client.

To configure a Host Checker policy with a Ports Check rule:

1. In the Web console, choose **System > Configuration > Security > Host Checker**.
2. Under **Policies**, click **New**. Enter a name in the **Policy Name** field and then click **Continue**.
3. Click the tab that corresponds to the operating system for which you want to specify Host Checker options—**Windows**, **Mac**, or **Linux**. In the same policy, you can specify different processes on each operating system.
4. Under **Rule Settings**, select **Attribute Check: Ports**
5. Enter a rule name and then specify settings for the Ports Check rule:

**Table 5: Attribute Check: Ports**

For this setting:	Specify:
<b>Port List:</b>	< a comma delimited list (without spaces) of ports or port ranges (1 to 65534) >
<b>Access Setting:</b>	Required or Deny

6. Click **Save Changes**.

## Registry Setting Checking

You can use a Registry Setting Check rule to require a client machine to have certain registry settings in order for the user to access the IVE. For example, you can require a client to have certain corporate PC images, system configurations, and software settings in order to access the IVE. You can also use Registry Setting Check rules to evaluate the age of required files and allow or deny access accordingly.

To configure a Host Checker policy with a Registry Setting Check rule:

1. In the Web console, choose **System > Configuration > Security > Host Checker**.
2. Under **Policies**, click **New**. Enter a name in the **Policy Name** field and then click **Continue**.
3. Click the tab that corresponds to the operating system for which you want to specify Host Checker options—**Windows**, **Mac**, or **Linux**. In the same policy, you can specify different processes on each operating system.
4. Under **Rule Settings**, select **Attribute Check: Registry Setting**

5. Enter a rule name and then specify settings for the Registry Setting rule:

**Table 6: Attribute Check: Registry Setting**

For this setting:	Specify:
<b>Registry Root key:</b>	< a root key value such as HKEY_LOCAL_MACHINE >
<b>Registry Subkey:</b>	< the path to the application folder for the registry subkey >
<b>Name:</b>	< name of the key's value that you want to require (optional) >
<b>Type:</b>	String, Binary, or dword
<b>Value:</b>	< required registry key value (optional) >

6. If the key value represents an application version, select **Minimum version** to allow the specified version or newer versions of the application. The IVE uses lexical sorting to determine if the client contains the specified version or higher. For example:

3.3.3 is newer than 3.3

4.0 is newer than 3.3

4.0a is newer than 4.0b

4.1 is newer than 3.3.1



**NOTE:** You can use this option to specify version information for an antivirus application to make sure that the client antivirus software is current.

7. Click **Save Changes**.

## Securing endpoints using specific checks

This section contains the following examples of using Host Checker to perform specific host checks:

- “Screen Saver Check” on page 12
- “Microsoft Hotfix Installation Check” on page 12
- “Windows Update Status Verification” on page 13
- “Network Bridge is Disabled” on page 13
- “IP Forwarding is Disabled” on page 13
- “Internet Connection Sharing Disabled” on page 14
- “Windows XP Firewall” on page 14
- “McAfee VirusScan Enterprise 7.1.0” on page 14

## Screen Saver Check

You can use the following Registry Setting Check rules in a Host Checker policy to automatically activate a screen saver and lock a host with a password after a period of inactivity. If the user leaves his computer unattended, the screen saver activates and prevents another user from using it.

This screen saver policy checks the registry to see if the screen saver is password protected and sets it to display a blank screen after 900 seconds (15 minutes) of inactivity.

**Table 7: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_CURRENT_USER
<b>Registry Subkey:</b>	Control Panel\Desktop\ScreenSaveTimeOut
<b>Type:</b>	string
<b>Value:</b>	900

**Table 8: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_CURRENT_USER
<b>Registry Subkey:</b>	Control Panel\Desktop\ScreenSaveActive
<b>Type:</b>	string
<b>Value:</b>	1

For configuration instructions, see “Registry Setting Checking” on page 10.

## Microsoft Hotfix Installation Check

You can use Registry Setting Check rules to check a Windows client’s registry for installed KB patches at the required level. The following rules are examples of checking for two specific patches.

**Table 9: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\Microsoft\Updates\Windows XP\SP2\KB810217\
<b>Type:</b>	string

**Table 10: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\Microsoft\Updates\Windows XP\SP2\KB821557\
<b>Type:</b>	string

For configuration instructions, see “Registry Setting Checking” on page 10.

## Windows Update Status Verification

You can use the following Registry Setting Check rule to check the Active Update configuration setting in a Windows client's registry to verify that the Windows update configuration is correct.

**Table 11: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate\Auto Update\AUState
<b>Type:</b>	dword
<b>Value:</b>	2

For configuration instructions, see “Registry Setting Checking” on page 10.

## Network Bridge is Disabled

You can use the following Registry Setting Check rule to verify that network bridging is disabled in Windows XP to ensure that a client cannot bridge traffic to another network while connected to the IVE.

**Table 12: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SYSTEM\CurrentControlSet\Services\BridgeMP\DisableForwarding
<b>Type:</b>	dword
<b>Value:</b>	1

For configuration instructions, see “Registry Setting Checking” on page 10.

## IP Forwarding is Disabled

You can use the following Registry Setting Check rule to verify that IP forwarding is disabled in Windows XP to ensure that a host is not allowed to forward (route) IP traffic.

**Table 13: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\IPEnableRouter
<b>Type:</b>	dword
<b>Value:</b>	0

For configuration instructions, see “Registry Setting Checking” on page 10.

## Internet Connection Sharing Disabled

You can use the following Registry Setting Check rule to verify that Internet Connection Sharing is disabled in Windows XP to ensure that a host is unable to share an Internet Connection, including a Network Connect connection.

**Table 14: Attribute Check: Registry Setting**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SYSTEM\CurrentControlSet\Services\SharedAccess\Start
<b>Type:</b>	dword
<b>Value:</b>	3

For configuration instructions, see “Registry Setting Checking” on page 10.

## Windows XP Firewall

You can use the following Process Check rule with a MD5 checksum value to verify that the correct version of the Windows XP personal firewall is enabled on a client.

**Table 15: Attribute Check: Process**

For this setting:	Enter this value:
<b>Process Name:</b>	alg.exe
<b>Access Setting:</b>	Required
<b>MD5 checksum:</b>	< calculate the appropriate MD5 checksum value >

For configuration instructions, see “Process Checking” on page 8.

## McAfee VirusScan Enterprise 7.1.0

You can combine the following Process Check, File Check, and Registry Setting Check rules to verify that McAfee VirusScan is running on a host and it is the correct version.

**Table 16: Attribute Check: Process**

For this setting:	Enter this value:
<b>Process Name:</b>	McShield.exe
<b>Access Setting:</b>	Required
<b>MD5 checksum:</b>	< calculate the appropriate MD5 checksum value >

For configuration instructions, see “Process Checking” on page 8.

**Table 17: Attribute Check: Files**

For this setting:	Enter this value:
<b>File Name:</b>	c:\Program Files\Network Associates\VirusScan\Shcfg32.exe
<b>Access Setting:</b>	Required
<b>MD5 checksum:</b>	< calculate the appropriate MD5 checksum value >

For configuration instructions, see “File Version Checking” on page 9.

**Table 18: Attribute Check: Registry Setting**

<b>For this setting:</b>	<b>Enter this value:</b>
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\Network Associates\TVD\VirusScan Enterprise\CurrentVersion\szProductVer
<b>Type:</b>	string
<b>Value:</b>	7.1.0.187

For configuration instructions, see “Registry Setting Checking” on page 10.

---

## Securing endpoints using Sygate On-Demand

For information on securing endpoints using Sygate On-Demand, see the *Sygate On-Demand/Juniper IVE SSL Integration Guide* available from the Juniper Support site.

---

## Securing endpoints using Sygate Security Agent

This section describes the recommended configuration of the IVE and the Sygate Security Agent for interoperability and optimal security.

### **Sygate Management Server Configuration**

The only required configuration you must perform on the Sygate Management Server is to create Host Integrity rule(s) and enable the Host Integrity checks for specific groups.

### **Sygate Security Agent Configuration**

You must configure the Sygate Security Agent to look up the Sygate Management Server(s) using a fully-qualified DNS domain name only. If you use a private IP address (such as 10.0.0.1) in the `sylink.xml` file, then the Sygate Security Agent cannot connect to the Sygate Management Server(s) via the IVE’s SSL VPN or Secure Application Manager (SAM).

Clients connected to the IVE use the policy defined in the Remote location.

When configuring the `sylink.xml` file, set the `ConfigControl FreezeSMSList` value to `1`. Otherwise, the Sygate Management Server overwrites the null IP address in the defined Sygate Management Server IP/name pairs. In subsequent communications, the client uses the IP and not the name to contact the server, thereby breaking the connection.

**Figure 1: Sample client sylink.xml file**

```
<?xml version="1.0" encoding="UTF-8" ?>
<ServerSettings>
  <ProfileControl ByServer="1" />
  <ConfigControl FreezeSmsList="1" FreezeSmsListOrder="0" />
  <RefreshSeconds>600</RefreshSeconds>
  <TimeoutSeconds>600</TimeoutSeconds>
  <RegisterClient BasedOnUserLogon="0" />
  <Server Ip="0" Name="smsserver1.corp.infopeople.ca" HttpPort="80" HttpsPort="443" />
  <Server Ip="0" Name="smsserver2.corp.infopeople.ca" HttpPort="80" HttpsPort="443" />
</ServerSettings>
```

As an alternative, you can also use an Internet-accessible IP address of the Sygate Management Server.



**NOTE:** You must configure the heartbeat interval in the `sylink.xml` file since the Sygate Management Server blocks it from updates.

## Securing endpoints using TrendMicro OfficeScan

This section describes a solution using Host Checker to require that Trend Micro OfficeScan is running and up to date before allowing a user to connect to the IVE. This solution verifies:

- The Trend Micro OfficeScan software and signatures are up to date.
- The real time scan option is enabled and the Trend Micro OfficeScan process is running.

### Trend Micro Office Scan Version Check

You can use the following Registry Setting Check rule to verify that OfficeScan is up to date with the specified version.

**Table 19: Attribute Check: Registry Setting for Pattern Version Check**

For this setting:	Enter this value:
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\TrendMicro\PC-cillinNTCorp\CurrentVersion\Misc.\PatternVer
<b>Type:</b>	dword
<b>Value:</b>	951

You can use the following Registry Setting Check rule to check the virus pattern version and the scanning engine version.

**Table 20: Attribute Check: Registry Setting for Engine Version Check**

<b>For this setting:</b>	<b>Enter this value:</b>
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\TrendMicro\PC-cillinNTCorp\CurrentVersion\Misc.\EngineZipVer
<b>Type:</b>	string
<b>Value:</b>	7.100

For configuration instructions, see “Registry Setting Checking” on page 10.

### **Trend Micro OfficeScan Realtime Scan**

You can use the following Process Check rule to check that OfficeScan is currently running on the workstation. You can also verify that it is the correct version by checking the OfficeScan’s MD5 checksum value.

**Table 21: Attribute Check: Process**

<b>For this setting:</b>	<b>Enter this value:</b>
<b>Process Name:</b>	NTRTScan.exe
<b>Access Setting:</b>	Required
<b>MD5 checksum:</b>	< calculate the appropriate MD5 checksum value >

For configuration instructions, see “Process Checking” on page 8.

You can use the following Registry Setting Check rule to check that Real Time scan is enabled.

**Table 22: Attribute Check: Registry Setting**

<b>For this setting:</b>	<b>Enter this value:</b>
<b>Registry Root key:</b>	HKEY_LOCAL_MACHINE
<b>Registry Subkey:</b>	SOFTWARE\TrendMicro\PC-cillinNTCorp\CurrentVersion\Real Time Scan Configuration\Enable
<b>Type:</b>	dword
<b>Value:</b>	1

For configuration instructions, see “Registry Setting Checking” on page 10.

---

## **Securing endpoints using Whole Security Confidence Online**

For information on securing endpoints using Whole Security Confidence Online, see the *Whole Security Confidence Online Integration Guide* available from the Juniper Support site.

## Securing endpoints that use the Google Desktop Search

Google Desktop Search creates a security concern for companies providing remote access to their network. In particular, this applies to SSL VPN sites as remote access is more readily available from most endpoints in a client-less fashion.

Google Desktop Search is an application composed of several processes that index content that a user viewed or saved. This enables a user to search for content from a web page hosted on the local computer. Google Desktop Search can search the full text of email, files, viewed web pages, and chats. Specifically, users can:

- Search email from Outlook 2000 + and Outlook Express 5 +
- Search files in TXT, HTML, DOC, XLS, and PPT formats (Office 2000 +)
- Search chats from AOL 7 + and AOL Instant Messenger 5 +
- Search web pages viewed in Internet Explorer 5 +

You can create a Host Checker policy that checks for the Google Desktop Search processes and displays a HTML remediation page. This HTML page can display instructions to the user to either launch a virtual desktop, and/or that you are killing the Google Desktop Search processes.

### Host Checker configuration

You can use the following Process Check rules to check whether Google Desktop Search is currently running on the workstation. If it is, you can deny access and configure remediation actions.

**Table 23: Attribute Check: Process**

For this setting:	Enter this value:
<b>Process Name:</b>	googledesktop.exe
<b>Access Setting:</b>	Deny

**Table 24: Attribute Check: Process**

For this setting:	Enter this value:
<b>Process Name:</b>	googledesktopindex.exe
<b>Access Setting:</b>	Deny

**Table 25: Attribute Check: Process**

For this setting:	Enter this value:
<b>Process Name:</b>	googledesktopcrawl.exe
<b>Access Setting:</b>	Deny

To configure a Host Checker policy with remediation actions for the Google Desktop Search Process Check rules:

1. In the **System > Configuration > Security > Host Checker** page of the Web console, create a Host Checker policy with the three Process Check rules to check whether Google Desktop Search is running. For configuration instructions, see “Process Checking” on page 8.
2. Under **Remediation** on the **Host Checker Policy** page, select **Enable Custom Instructions** and enter the instructions you want to display to the user on the Host Checker remediation page. For example, you can instruct the user that Google Desktop Search is running and to click a link to launch a virtual desktop.
3. If you want to kill the Google Desktop Search processes, select **Kill Process** and then enter the names of the three Google Desktop Search processes on each line. Note that the user must have the required privileges to kill the processes.



## Chapter 3

# Using J.E.D.I. interfaces to implement endpoint solutions

If you want to implement your own third-party endpoint solution and cannot use the methods described in the previous chapter, you can use the J.E.D.I. APIs provided with the product.



**NOTE:** The information in this chapter applies to Host Checker running on Windows clients only.

---

An IVE appliance provides two different APIs that you can use to integrate third-party functionality:

- **Host Check Client Interface (Windows only)**—The Host Check Client Interface is an API that allows you to run your own DLLs using Host Checker. Through the interface, you can prompt Host Checker to run a DLL that you already installed on the user’s system or distributed as part of a corporate OS image, including programs that check compliance with corporate images, antivirus software, and personal firewall clients. Host Checker runs the specified DLL when a user signs into the IVE, and then bases its subsequent actions on the success or failure result that the DLL returns. For example, you may deny a user access to the IVE if the client check software fails. For more information, see “Using the Host Check Client Interface” on page 22.
- **Host Check Server Integration Interface (Windows only)**—The Host Check Server Integration Interface is an API that allows you to tightly integrate a J.E.D.I. compliant system with the IVE. Like the Host Check Client Interface, you can use the Host Check Server Integration Interface to prompt Host Checker to run third-party software on the client, including host integrity scans, malware detectors, and virtual environments. With this interface, you may also specify with granularity what Host Checker should do based on the result of the diverse policy checks that the third-party applications conduct. You can invoke these policies to dynamically map users to realms, roles, and resources based on the results of individual policies contained in your software package. For information, see “Using the Host Check Server Integration Interface” on page 25.

For more information on the Host Check interfaces, see:

- “Choosing which interface to use” on page 22
- “Using the Host Check Client Interface” on page 22
- “Using the Host Check Server Integration Interface” on page 25

## Choosing which interface to use

The following table outlines when to use the Host Check Client Interface vs. the Host Check Server Integration Interface.

**Table 26: Host Check client interface vs. server integration interface**

Use the client interface to:	Use the server interface to:
<ul style="list-style-type: none"> <li>■ Enforce simple Host Checker policies. For example, you can use the client interface to return a yes/no value for a question such as “Is the proper software installed, running, or compliant?”</li> </ul>	<ul style="list-style-type: none"> <li>■ Enforce multiple Host Checker policies and/or hierarchical Host Checker policies.</li> </ul>
<ul style="list-style-type: none"> <li>■ Reference DLLs that are already installed on the user’s system</li> </ul>	<ul style="list-style-type: none"> <li>■ Deploy third-party applications to the user’s system</li> <li>■ Integrate custom DLLs or software onto the IVE appliance</li> </ul>
<ul style="list-style-type: none"> <li>■ Enforce policies that do not require logging</li> </ul>	<ul style="list-style-type: none"> <li>■ Log results from your policy checks</li> </ul>

## Using the Host Check Client Interface

The Host Check Client Interface involves communicating with a third-party endpoint security application through its API and examining the return values to verify the trustworthiness of the client machine. Through the Host Check Client Interface, the IVE Host Checker feature currently supports tight integration with Sygate Enforcement API, Sygate Security Agent, Zone Labs ZoneAlarm Pro, Zone Labs Integrity, McAfee Desktop Firewall 8.0, and InfoExpress CyberGatekeeper Agent. To support other endpoint security applications or those that do not have an API, the IVE platform provides a generic API library in the C programming language. This Windows API is called the Host Check Client Interface API and contains the `NHC_EndpointSecure()` function, which checks the endpoint configuration.

Host Check Client Interface integration typically includes these steps:

1. An IVE administrator enables Host Checker for the desired realm, role, or resource. For this realm, role, or resource, the administrator specifies a 3rd party NHC check rule on the Web Console's Host Checker page. This rule specifies the location of your custom DLL on a client machine.
2. The IVE appliance downloads an ActiveX installer with the Host Check Client Interface package to the client machine of an authenticated user who is trying to access the realm, role, or resource.

3. The Host Check Client Interface package loads your custom DLL from the DLL location on the client. Before calling the `NHC_EndpointSecure()` function, the package calls the Windows `WinVerifyTrust()` function to validate the DLL's digital signature. (See "Signing your custom DLL" on page 23 for information about the user experience.)
4. The Host Check Client Interface package calls the `NHC_EndpointSecure()` function. If the function returns `NHC_STATUS_SECURE`, the third-party product endpoint security check succeeds and the IVE appliance maps the user to the realm, role, or resource. If the endpoint security check fails for a realm-level policy, the user sees an error stating that the computer does not comply with the endpoint security policy, and the user is redirected to the sign-in page. If you specify the URL to a failure page, this page opens in another browser window. If the endpoint security check fails for a role or resource level policy, the IVE appliance simply does not map the user to the role or resource.

### **Signing your custom DLL**

We strongly recommend that you digitally sign your custom DLL to ensure content integrity. Prior to calling your DLL, Host Checker calls the Windows `WinVerifyTrust()` function to attempt to verify the trustworthiness of the DLL.

- If your DLL is not digitally signed and the user's browser security settings are Medium-to-High, a message informs the user that the DLL is not signed and the Windows `WinVerifyTrust()` function cannot verify it as trustworthy. The user may choose to proceed, in which case Host Checker calls the DLL. If the `NHC_EndpointSecure()` function returns `NHC_STATUS_SECURE`, the IVE maps the user to the realm, role, or resource. If the user chooses to cancel the operation, the IVE does not map the user to the realm, role, or resource.
- If the DLL is digitally signed but `WinVerifyTrust()` cannot verify the trustworthiness of the DLL, a message informs the user. The user may choose to proceed, in which case the Host Checker calls the `NHC_EndpointSecure()` function. If this function returns `NHC_STATUS_SECURE`, the IVE maps the user to the realm, role, or resource. If the user chooses to cancel the operation, the IVE appliance does not map the user to the realm, role, or resource.
- If the DLL is digitally signed and `WinVerifyTrust()` verifies the trustworthiness of the DLL, a message informs the user that the content provider and its integrity have been verified and asked if he wishes to proceed. If the user proceeds, Host Checker calls `NHC_EndpointSecure()`. If this function returns `NHC_STATUS_SECURE`, the IVE maps the user to the realm, role, or resource.

## Deploying and maintaining your custom DLL

To deploy your custom DLL, you need to:

- At the system level, configure the Host Checker feature to call your custom DLL and specify the path (including the file name) to where the DLL is stored on client machines.
- Install the DLL on the appropriate client machines or distribute the DLL as part of the corporate PC image.
- Create a low-privilege realm or role that users can access if the endpoint security check fails and make the DLL available on a page configured as an IVE bookmark for those users.
- Check the DLL timestamp to ensure that the version is current. If the endpoint security check fails, redirect users to a "safety page" that enables them to authenticate and download the current DLL.

## NetScreen Host Checker (NHC) API

This section contains function definitions for the NHC API.

### NHC API Definitions

```
#define NHC_STATUS_SECURE      1
#define NHC_STATUS_FAILURE    -1
#define NHC_STATUS_UNSECURE   -2
#define NHC_STATUS_BADPARAMETER -3
```

### NHC\_EndpointSecure() Required

The NHC\_EndpointSecure function verifies the security of the endpoint based on the criteria established by the implementer.

### Syntax

```
NHC_API int WINAPI NHC_EndpointSecure(void);
```

### Parameters

None

### Return Values

- NHC\_STATUS\_SECURE—The endpoint client is secure
- NHC\_STATUS\_UNSECURE—The endpoint client is not secure
- NHC\_STATUS\_FAILURE—The endpoint application is not running
- NHC\_STATUS\_BADPARAMETER—An internal error has occurred; the endpoint client should be judged insecure

**C Header file: NHC.h**

Include this header file in your DLL:

```
#ifndef NHC_EXPORTS
#define NHC_API __declspec(dllexport)
#else
#define NHC_API __declspec(dllimport)
#endif

#define NHC_STATUS_SECURE      1
#define NHC_STATUS_FAILURE    -1
#define NHC_STATUS_UNSECURE   -2
#define NHC_STATUS_BADPARAMETER -3

#ifdef __cplusplus
extern "C" {
#endif
NHC_API int NHC_EndpointSecure(void);
#ifdef __cplusplus
}
#endif
```

---

## Using the Host Check Server Integration Interface

The Host Check Server Integration Interface contains a generic API library in the C++ programming language. This section describes the API and how to implement it for use with Host Checker.

### ***Deploying third party applications through Host Checker***

To deploy third party applications through Host Checker, you must complete the steps defined in the following sections:

1. “Create an endpoint security package” on page 26
2. “Upload the package to the IVE” on page 29
3. “Configure Host Checker to use the package” on page 30

## Create an endpoint security package

An endpoint security package is a collection of files that comprise your third party functionality. When creating a package, you must include the following types of files:

- **Interface DLL**—An interface DLL is a program that carries out your specialized functions on the client. When creating your DLL, you must provide an interface between your modules and Host Checker using functions defined in the Host Check Server Integration Interface.
- **Package definition file**—A package definition file, which must be named `MANIFEST.HCIF`, defines the name of your interface DLL, the Host Checker policies defined in your DLL, and any optional pre-authentication access tunnel definitions. For more information, see “Creating the `MANIFEST.HCIF` package definition file” on page 26. Note that if you do not include policies in your package, Host Checker simply enforces that the package has run on the client. If you do declare policies through this file, they become available through the IVE Web console where you can implement them at the realm, role, and resource policy levels.
- **Host Check Server Integration Interface header file**—The Host Check Server Integration Interface header file (`hcif.h`) defines the Host Check Server Integration Interface functions that you must use in your DLL. You must add this file to your package’s `include` folder in order to use the functions that are provided with the Host Check Server Integration Interface. You can find a copy of this file in the SDK provided with the product.

In addition to these required files, you may also include your own data files in the package.

### Creating the `MANIFEST.HCIF` package definition file

Within the `MANIFEST.HCIF` package definition file, you must include one definition per line, with a blank line between each definition, using the following format:

```
HCIF-Main : <DLLName>
```

```
HCIF-Policy : <PolicyName>
```

```
HCIF-IVE-Tunnel: <client-loopback>:port <policy-server>:port
```

where:

`<DLLName>` is the name of the interface DLL, such as `myPestPatrol.dll`.

`<PolicyName>` is the name of a policy defined in the DLL, such as `myFileCheck`. You can define multiple policies by using the `HCIF-Policy` statement for each policy.

`HCIF-IVE-Tunnel: <client-loopback>:port <policy-server>:port` is an optional pre-authentication access tunnel definition. For information on the syntax of a tunnel definition, see “Specify Host Checker pre-authentication access tunnel definitions” on page 28.

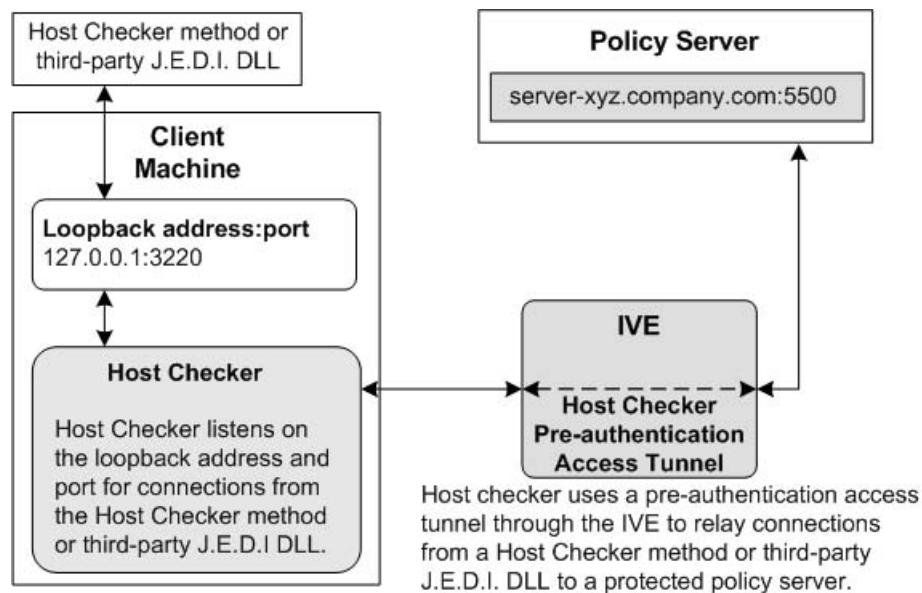
### Using access tunnels to make policy servers available to Host Checker clients

If your policies require Host Checker methods or third-party J.E.D.I. DLLs to access a policy server (or other resource) to check compliance before users are authenticated, you can use one of the following methods to make the resource available to the Host Checker Windows clients:

- Deploy the policy server in a DMZ where Host Checker methods or third-party J.E.D.I. DLLs can access the server directly instead of going through the IVE**—This is the simplest deployment because you do not have to define a Host Checker pre-authentication access tunnel through the IVE between clients and the policy server.
- Deploy the policy server in a protected zone behind the IVE (Windows only)**—This deployment requires you to define a *pre-authentication access tunnel*. A pre-authentication access tunnel enables Host Checker methods or third-party J.E.D.I. DLLs to access the IVE-protected policy server or resource before the IVE authenticates users. To define a pre-authentication access tunnel, you associate a loopback address (or host name) and port on the client with an IP address and port on the policy server. You add one or more tunnel definitions to the **MANIFEST.HCIF** file, which you then upload to the IVE. You can upload multiple **MANIFEST.HCIF** files to the IVE. For all third-party policies enabled on a realm, Host Checker creates tunnels for all of the tunnel definitions in all of the **MANIFEST.HCIF** files, assuming the definitions are unique. For configuration instructions, see “Upload a Host Checker policy package to the IVE” on page 121.

While running on a Windows client, Host Checker listens for a connection on each loopback address and port you specify in the tunnel definitions. The connections can originate from the integrated Host Checker methods and from client-side or server-side J.E.D.I. DLLs. Host Checker uses the pre-authentication access tunnel(s) to forward the connections through the IVE to the policy server(s) or other resource.

**Figure 2: Host Checker creates a tunnel from a client to a policy server behind the IVE**





**NOTE:** Host Checker pre-authentication access tunnels are supported on Windows only.

### **Specify Host Checker pre-authentication access tunnel definitions**

A definition for a Host Checker pre-authentication access tunnel configures access to one policy server or other resource. Each tunnel definition consists of a pair of IP addresses and ports: one loopback IP address and port on the client, and one IP address and port on the policy server. You specify one or more tunnel definition(s) in the MANIFEST.HCIF package definition file.

The syntax of a pre-authentication access tunnel definition is:

```
HCIF-IVE-Tunnel: <client-loopback>:port <policy-server>:port
```

where:

<client-loopback> is a loopback address that begins with 127. and takes any of the following forms:

- An IP address and port that takes the form of 127.\*.\*:port. To avoid conflicts with JSAM, do not use 127.0.0.1 with port 80, but you can use 127.0.0.1 with other ports. For example: 127.0.0.1:3220
- A host name that resolves to a loopback address that begins with 127. You can use a local hosts file on each client computer or a DNS server to resolve the loopback address.
- A host name that does not resolve to a loopback address, or it resolves to a non-loopback address. In these cases, Host Checker allocates a loopback address and updates the local hosts file on the client with the mapping. Note that the user must have administrator privileges in order for Host Checker to modify the local hosts file. If the user does not have administrator privileges, Host Checker cannot update the hosts file and cannot open the pre-authentication access tunnel. In that case, Host Checker logs an error.

<policy-server> is the IP address or host name of the back-end policy server. The IVE resolves the host name you specify.

For example, in the following tunnel definition, 127.0.0.1:3220 is the client loopback address and port, and mysygate.company.com:5500 is the policy server host name and port:

```
HCIF-IVE-Tunnel: 127.0.0.1:3220 mysygate.company.com:5500
```

Or you can use a host name for the client, as in this example:

```
HCIF-IVE-Tunnel: mysygate.company.com:3220 mysygate.company.com:5500
```

Keep the following in mind when specifying tunnel definitions:

- You must add a blank line between each line in the **MANIFEST.HCIF** file, and you can use a semi-colon at the beginning of a line to indicate a comment. For example:

```
HCIF-Main : myPestPatrol.dll
```

```
HCIF-Policy : myFileCheck
```

```
HCIF-Policy : myPortCheck
```

```
; Tunnel definitions
```

```
HCIF-IVE-Tunnel: 127.0.0.1:3220 mysygate.company.com:5500
```

```
HCIF-IVE-Tunnel: 127.1.1.1:3220 mysygate2.company.com:5500
```

```
HCIF-IVE-Tunnel: mysygate.company.com:3220 mysygate3.company.com:5500
```

- If `<client-loopback>` is a non-loopback address, then Host Checker cannot open the pre-authentication access tunnel and logs an error instead.
- If you use a loopback address other than **127.0.0.1** (such as **127.0.0.2** and above), clients who are using Windows XP Service Pack 2 must install the XP SP2 Hot Fix. See:

```
http://support.microsoft.com/default.aspx?scid=kb;en-us;884020
```

- Your DLL must use the same `<client-loopback>` address and port or host name that you specify in the **MANIFEST.HCIF** file.
- Because a pre-authentication access tunnel is open only while Host Checker is running, your DLL can access its IVE-protected policy server only while Host Checker is running.
- If your DLL uses **HTTPS** to connect to its policy server via a host name that resolves properly to the loopback address, no server certificate warnings appear. However, if the your DLL connects explicitly via a loopback address, then server certificate warnings do appear because the host name in the certificate does not match the loopback address. (Your DLL can configure the DLL to ignore these warnings.)

### Upload the package to the IVE

In order for the IVE to recognize a package definition file, you must name it **MANIFEST.HCIF** and include it a folder named **META-INF**. You must then archive the the **META-INF** folder containing the **MANIFEST.HCIF** file along with the interface DLL and any initialization files in one zip file for the policy package. For example, the contents of the zip file for a policy package can be:

```
META-INF/MANIFEST.HCIF
hcif-myPestPatrol.dll
hcif-myPestPatrol.ini
```

You can upload multiple policy packages each containing a different `MANIFEST.HCIF` file to the IVE. Host Checker creates tunnels for all of the tunnel definitions in all of the `MANIFEST.HCIF` files, assuming the definitions are unique.

Then you upload the package zip file to the IVE through the **System > Configuration > Security > Host Checker** page of the Web console. Note that after you upload a Host Checker policy package to the IVE, you cannot modify the package contents on the server. Instead, you must modify the package on your local system and then upload the modified version to the IVE.

### Configure Host Checker to use the package

After you upload a valid package, the IVE automatically exposes the policies contained in the package in the realm, role, and resource policy configuration pages of the Web console. You may then use these pages to implement the policies. When a user tries to access a realm, role, or resource protected by one of these policies, the IVE runs the program that you have uploaded to the server.

## Creating your interface DLL



**NOTE:** Version 5 of the IVE supports a new version 2 of the Host Checker API that contains the following changes:

- The `HCIF_Module.checkPolicy ()` function is a new function that replaces the `HCIF_Check()` function in version 1.
- The `HCIF_Module.remediate()` function is new.

Version 5 of the IVE also supports version 1 of the Host Checker API. For information on specifying the version you want to use, see “`HCIF_Module.version()` function” on page 32.

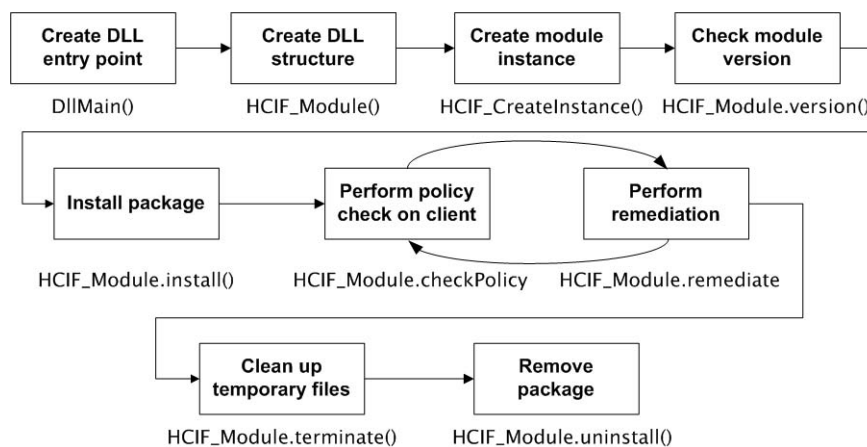
When you create an interface DLL to carry out your specialized functions on user machines, you must use the functions provided in the Host Check Server Integration Interface to:

1. Provide an entry point for the DLL, as explained in “`DllMain()` function” on page 31.
2. Create a module structure for your DLL, as explained in “`HCIF_Module()` function” on page 31
3. Create an instance for your module, as explained in “`HCIF_CreateInstance()` function” on page 32.
4. Specify the API version you want to use with your module, as explained in “`HCIF_Module.version()` function” on page 32.
5. Install your endpoint security package on the user’s machine, as explained in “`HCIF_Module.install()` function” on page 32.
6. Perform endpoint security checks on individual policies, as explained in “`HCIF_Module.checkPolicy()` function (version 2 only)” on page 32. This function loops with the `HCIF_Module.remediate()` function.

7. (Optional) Perform remediation, as explained in “HCIF\_Module.remediate() function (version 2 only)” on page 33. This function loops with the HCIF\_Module.checkPolicy() function.
8. Stop the DLL and clean up temporary files, as explained in “HCIF\_Module.terminate() function” on page 33.
9. Remove your package from the user’s machine, as explained in “HCIF\_Module.uninstall() function” on page 33.

The following diagram illustrates the steps and corresponding functions:

**Figure 3: Creating your interface DLL (Host Checker API version 2)**



**NOTE:**

- The hcif.cpp sample file in the SDK includes all of the functions described here.
- You can test whether your package properly conforms to the standards described here using the hciftool.exe tool provided with the SDK.

### DllMain() function

Use the DllMain() function to define an entry point for your DLL within the main Host Checker framework. This function alerts Host Checker that it needs to execute your third party DLL. When adding the DllMain() function to your project, copy and use the exact version that is in the hcif.cpp sample that is included with the SDK.

### HCIF\_Module() function

Use the HCIF\_Module() function to create a structure for your DLL. You should call all of the other functions described in this section from within the HCIF\_Module() function.

**HCIF\_CreateInstance() function**

Use the `HCIF_CreateInstance()` function to create an individual instance of your service and to create a DLL entry point. The Host Checker framework calls the `HCIF_CreateInstance()` function to obtain a pointer to the `HCIF_Module` structure.

**HCIF\_Module.version() function**

Use the `HCIF_Module.version()` function to return the Host Checker API version that was used to compile the module. The Host Checker framework uses the return value to determine whether to call the `HCIF_Module.checkPolicy()` function or the `HCIF_Check()` function. If you specify the version return value is 2 (which is the default value defined in the `HCIF_API_Version` constant), the Host Checker framework calls the `HCIF_Module.checkPolicy()` function. If you specify the version return value is 1, the Host Checker framework calls the `HCIF_Check()` function. For backwards compatibility, the framework only calls the `HCIF_Check()` function for a version 1 interface DLL.

**HCIF\_Module.install() function**

Use the `HCIF_Module.install()` function to deliver required files from the IVE to the user's system. Within the `HCIF_Module.install()` function, you may use the standard C `getFile` function to retrieve files that you have uploaded to the IVE and copy them to Host Checker's default directory on the user's system:

```
C:\Documents and Settings\\Application Data\Juniper Networks\Host Checker
```

**HCIF\_Check() function (version 1 only)**

For an interface DLL that uses version 1 of the Host Checker API, use the `HCIF_Check()` function to perform your client-side checks. You may use the `HCIF_Check()` function to run the core execution modules in your package. These modules should check the compliance of the endpoint with the configured policies and return a value of `TRUE` if the check was successful or `FALSE` if the check failed.

**HCIF\_Module.checkPolicy() function (version 2 only)**

For an interface DLL that uses version 2 of the Host Checker API, use the `HCIF_Module.checkPolicy()` function to perform client-side checks using individual policies. For example, suppose there are two policies named `Policy-1` and `Policy-2`. The framework calls this function for `Policy-1`, and then it calls this function again for `Policy-2` individually. That is, you can call an individual policy and run a specific set of code for that policy only.

This function should check the compliance of the endpoint with the configured individual policies and call `HCIF_Service report()` to report the status of each individual policies. The framework calls this function periodically during a session. This function returns a value of `TRUE` if the check was successful or `FALSE` if the check failed.

**NOTE:**

- Only those policies enabled on a realm are passed to the `HCIF_Module.checkPolicy()` function.
  - The framework uses the return value of the `HCIF_Module.checkPolicy()` function as the status of the overall Host Checker policy that contains the individual J.E.D.I. third-party policies.
- 

**HCIF\_Module.remediate() function (version 2 only)**

For an interface DLL that uses version 2 of the Host Checker API, Host Checker passes a list of failed policies to the `HCIF_Module.remediate()` function. Use the list of failed policies in the `HCIF_Module.remediate()` function to perform automatic or user-driven remediation actions to fix the policy failures. The framework only calls this function if the **Remediate** option is enabled on the associated Host Checker policy. (This option is on the **System > Configuration > Security > Host Checker** page of the Web console). This function returns a value of `TRUE` if the remediation was successful, or `FALSE` if the remediation failed.



**NOTE:** The return value of the `HCIF_Module.remediate()` function does *not* affect the status of the overall Host Checker policy that contains the individual J.E.D.I. third-party policies.

---

**HCIF\_Module.terminate() function**

Use the `HCIF_Module.terminate()` function to perform any final clean up at the end of a session, including freeing resources, removing temporary files, or reverting registry entries. The Host Checker framework calls the `HCIF_Module.terminate()` function at the end of the user's IVE session.

**HCIF\_Module.uninstall() function**

Use the `HCIF_Module.uninstall()` function to remove all traces of your module from the user's system. The Host Checker framework calls the `HCIF_Module.uninstall()` function after the `HCIF_Module.terminate()` function.

