

AhnLab HackShield

Programming Guide

Version 5.1

AhnLab HackShield Pro for Online Games

Preface

Copyright (C) AhnLab, Inc. 1988-2008. All rights reserved.

Contents of this document and the related software programs are protected by the Copyrights Act and the Computer Program Protection Act.

Document Overview

This document describes the overall structure and API functions of AhnLab HackShield for Online Games and AhnLab HackShield Pro for Online Games (hereinafter referred to as HackShield).

Customer Support

AhnLab Customer Center	
Address	AhnLab, 6 th Fl., CCMM Building, Yeouido-dong 12, Yeongdeungpo-gu, Seoul (150-869)
Home Page	http://global.ahnlab.com
Email	hs@ahnlab.com
Telephone	82-2-2186-3082 (Corporate Hotline) 82-2-2186-3000 (Customer Support)
Fax	82-2-2186-6100 (Representative) 82-2-2186-3001 (Customer Support)

Table of Contents

1.	Introduction to HackShield	8
1.1.	Functions.....	8
1.2.	System Environment.....	9
2.	Basic Functions	10
2.1.	Overview	10
2.1.1.	<i>Functions</i>	10
2.1.2.	<i>Features</i>	12
2.1.3.	<i>System Architecture</i>	13
2.2.	Application Programming	15
2.2.1.	<i>Programming Procedure</i>	16
2.2.2.	<i>Preparation</i>	18
2.2.3.	<i>Calling Initialization Function</i>	26
2.2.4.	<i>Calling Service Starting Function</i>	29
2.2.5.	<i>Writing Callback Function</i>	29
2.2.6.	<i>Calling Service Stopping Function</i>	30
2.2.7.	<i>Calling Service Uninitializing Function</i>	30
2.3.	Application Programming Interface	31
3.	Server Side Detection (For HackShield Pro).....	50
3.1.	Overview	50
3.1.1.	<i>Functions</i>	50
3.1.2.	<i>Features</i>	50
3.1.3.	<i>System Architecture</i>	51
3.2.	Application Programming	54
3.2.1.	<i>Programming Application</i>	54
3.3.	Application Programming Interface	56
4.	Extended Server Side Detection (For HackShield Pro, 4.2 and higher versions)	78
4.1.	Overview	78
4.1.1.	<i>Functions</i>	78
4.1.2.	<i>Features</i>	79
4.1.3.	<i>System Architecture</i>	79
4.2.	Application Programming	82
4.2.1.	<i>Programming Application</i>	82
4.3.	Application Programming Interface	84
5.	Real-time Decryption of Executable file (For HackShield Pro).....	101
5.1.	Overview	101
5.1.1.	<i>Functions</i>	101
5.1.2.	<i>Features</i>	102
5.1.3.	<i>System Architecture</i>	102
5.2.	Application Programming	104
5.2.1.	<i>Programming Procedure</i>	104
5.2.2.	<i>Preparation</i>	104
5.2.3.	<i>HspLauncherLib-related Files</i>	105
5.2.4.	<i>_AhnHsp_StartLauncher</i>	105
5.2.5.	<i>_AhnHsp_StartGame</i>	106
5.2.6.	<i>_AhnHsp_CloseHandle</i>	107
5.3.	Application Programming Interface	108
6.	Data File/Message Encryption	114

6.1.	Overview	114
6.1.1.	Function	114
6.1.2.	Features.....	114
6.1.3.	System Architecture.....	115
6.2.	Application Programming	116
6.2.1.	Programming Procedure.....	116
6.2.2.	Preparation	116
6.2.3.	HsCryptLib File	117
6.2.4.	_HsCrypt_InitCrypt.....	117
6.2.5.	_HsCrypt_GetEncMsg.....	118
6.2.6.	_HsCrypt_GetDecMsg.....	118
6.2.7.	_HsCrypt_FRead.....	119
6.3.	Application Programming Interface	120
7.	User Authority Support	127
7.1.	Overview	127
7.1.1.	Functions	127
7.1.2.	Features.....	127
7.1.3.	System Architecture.....	128
7.2.	Application Programming	129
7.2.1.	Programming Procedure.....	129
7.2.2.	Preparation	130
7.2.3.	_AhnHsUserUtil_CreateUser.....	130
7.2.4.	_AhnHsUserUtil_SetFolderPermission.....	131
7.3.	Application Programming Interface	132
8.	HackShield Update (For HackShield Pro)	137
8.1.	Overview	137
8.1.1.	Functions	137
8.1.2.	Features.....	137
8.1.3.	System Architecture.....	138
8.2.	Application Programming	139
8.2.1.	Programming Application.....	139
8.3.	Application Programming Interface	143
9.	Monitoring Service (For HackShield Pro)	146
9.1.	Overview	146
9.1.1.	Functions	146
9.1.2.	Features.....	146
9.1.3.	System Architecture.....	147
9.2.	Application Programming	148
9.2.1.	Programming Application.....	148
9.3.	Application Programming Interface	150
10.	LMP (For HackShield Pro).....	153
10.1.	Overview.....	153
10.1.1.	Functions	153
10.1.2.	Features.....	154
10.1.3.	System Architecture.....	154
10.2.	Application Programming	156
10.2.1.	Programming Application.....	156
11.	Tools.....	160
11.1.	HackShield Cryptor Tool (For HackShield Pro)	160
11.1.1.	What Is HackShield Cryptor?.....	160
11.1.2.	Functions	160
11.2.	Using HackShield Cryptor Tool.....	161

11.2.1.	<i>Screen Description</i>	161
11.2.2.	<i>Menu Description.....</i>	162
11.2.3.	<i>Using Tree View</i>	167
11.2.4.	<i>Using File List</i>	169
11.2.5.	<i>Checking Log View</i>	171
11.2.6.	<i>Encrypting Executable files Using HackShield Cryptor Tool.....</i>	172
11.2.7.	<i>Executing HCryptor.exe</i>	177
11.3.	<i>AntiCpSvr Tool(For HackShield Pro)</i>	178
11.3.1.	<i>Functions</i>	178
11.4.	<i>AntiCpSvr Tool</i>	179
11.4.1.	<i>Creating CRC Information File Based on Manual UI Setting</i>	179
11.4.2.	<i>Automatically Creating CRC Information File</i>	180
11.5.	<i>HSBGen Tool (For HackShield Pro 4.2 or higher versions)</i>	181
11.5.1.	<i>Functions</i>	181
11.6.	<i>Using HSBGen Tool.....</i>	182
11.6.1.	<i>Creating HSB Information File Based on Manual UI Settings.....</i>	182
11.6.2.	<i>Automatically Creating HSB Information File</i>	183
11.7.	<i>HSUpSetEnv Tool (For HackShield Pro 5.1 or higher versions)</i>	185
11.7.1.	<i>Functions</i>	185
11.8.	<i>Using HSUpSetEnv Tool.....</i>	186
11.8.1.	<i>Creating HSUpdate.env File</i>	186
11.9.	<i>CSInspector Tool (For HackShield Pro 5.1 or higher versions).....</i>	188
11.9.1.	<i>Functions</i>	188
11.10.	<i>CSInspector Tool</i>	189
11.10.1.	<i>Setting the File to Protect</i>	189
12.	<i>Appendix</i>	190
12.1.	<i>FAQ</i>	190
12.2.	<i>Index.....</i>	191
12.3.	<i>Revisions.....</i>	194

Tables

Table 2-1 HackShield-related Files	18
Table 2-2 Files in Smart Update Module	19
Table 2-3 Automatic Smart Update Creation Files.....	20
Table 2-4 Files to be Installed in Update Server	21
Table 2-5 Files to be Installed in Smart Update Server.....	22
Table 2-6 Update Server Configuration.....	23
Table 3-1 AntiCPSvr-related Files	54
Table 4-1 AntiCpXSvr-related Files	82
Table 5-1 HsCryptLib-related Files.....	105
Table 6-1 HsCryptLib Files	117
Table 7-1 HsUserUtil Files.....	130
Table 8-1 update-related Files.....	139
Table 8-2 Files to be Installed in the Update Server	140
Table 9-1 Monitoring-related Files.....	148
Table 10-1 LMP-related Files	156
Table 10-2 Packets Supported by LMP.....	159

Figures

Figure 2-1 Structure of HackShield Pro.....	13
Figure 2-2 Application Programming Procedure	16
Figure 2-3 Update Server Configuration.....	23
Figure 3-1 Operating Principles of AntiCpSvr.....	52
Figure 4-1 Operating Principles of AntiCpX	80
Figure 5-1 General Architecture and Operating Principles of Real-time Decryption	102
Figure 6-1 General Architecture and Operating Principles of HsCryptLib.....	115
Figure 7-1 General Architecture and Operating Principles of HsUserUtil	128
Figure 7-2 HsUserUtil Programming Procedure.....	129
Figure 8-1 HackShield update Operating Principles of	138
Figure 9-1 Operating Principles of the Monitoring Service.....	147
Figure 10-1 Operating Principles of Local Memory Protection.....	155
Figure 11-1 HackShield Cryptor	161
Figure 11-2 Recent Project File List	162
Figure 11-3 Ehsvc.dll File Error	164
Figure 11-4 HackShield Cryptor Version Management.....	165
Figure 11-5 HackShield Cryptor Version	166
Figure 11-6 Renaming Game Folder.....	167
Figure 11-7 Creating Folder.....	167
Figure 11-8 Deleting Folder.....	168
Figure 11-9 File List View	169
Figure 11-10 Adding File	169
Figure 11-11 Deleting File	170
Figure 11-12 Deleting All Files	170
Figure 11-13 Viewing File Properties	170
Figure 11-14 Log View.....	171
Figure 11-15 HackShield CRC Information Creation Tool	179
Figure 11-16 Command-line Type AntiCpSvrTool.exe.....	180
Figure 11-17 HSB File Generator.....	182
Figure 11-18 Command-line Type HSBGen.exe (for General Files).....	183
Figure 11-19 Command-line Type HSBGen.exe (for Packed Files)	184
Figure 11-20 HSUpSetEnv.exe	186
Figure 11-21 CSInspector.exe.....	189

1. Introduction to HackShield

AhnLab HackShield is a security solution for online games. It provides the functions; 1) detect and block hack attacks, 2) prevent cracks, 3) protect executable files, and 4) provide data encryption.

1.1. Functions

Anti-Hacking

Functions include signature-based hacking tool detection, anti-memory hack, anti-speed hack, debugging block, message hooking block, file manipulation block, and auto mouse block.

Server-side Solution for Cracking Prevention (For HackShield Pro)

Interfaces with servers to detect manipulation of executable files and memory, and check the operational status of HackShield.

Real-time Decryption of Executable files (For HackShield Pro)

Encrypts important files using library and executable files provided by HackShield, and decrypts encrypted executable files in real time. Protects executable files from malicious users by decrypting them in real time.

Data File/Message Encryption

Encrypts important data files and messages exchanged between the server and client to protect them from unauthorized users.

User Account Execution

Provides the function of running programs and executing files under User or Guest account, not Administrator account, in NT series.

1.2. System Environment

The following system environment is required for installing and operating HackShield.

Category	Recommended	Minimum Requirements
OS	Windows 98/ME/2000 Professional/XP Home /XP Professional/Server 2003/Vista	Windows 98 or higher
CPU	Intel Pentium 500Mhz or higher	Intel Pentium 133MHz or higher IBM-PC compatible
RAM	128MB or higher	32MB
HDD	2MB or higher	2MB

Caution

When operating HackShield in a Windows NT series server such as a web server or a DB server, performance may decline unexpectedly.

Note

Only Intel x86 series CPUs (including x86 compatible AMD) are supported. Alpha chip machines running on Windows NT or NEC pc 8xxx machines running on Windows are not supported.

2. Basic Functions

This chapter describes the system structure and programming methods needed to implement HackShield's anti-hacking and hacking tool detection functions.

Note

The sample codes contained in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

2.1. Overview

2.1.1. Functions

Signature-based Hacking Tool Detection

AhnLab signatures detect hacking tools registered in the engine. If a hacking tool is registered in the engine, the corresponding process will be forcibly terminated (depending on the option, the process may not be forcibly terminated) and the corresponding file name will be sent to the game client by the callback function. If a hacking tool not registered in the engine is detected, the signature will be updated to block the hacking tool.

Anti-Memory Hack

HackShield blocks memory access through Windows API (OpenProcess, Read/WriteProcessMemory...) used in the game program. HackShield directly traces the memory at the kernel level and blocks the hack attack which may manipulate the result data.

Caution

If a program that is not a hacking tool directly accesses the memory, the program may not properly operate.

Anti-Speed Hack

Speed hacks usually manipulate the timer installed in the system (hardware method) or time-related APIs in the OS (software method). To prevent such speed hack attacks, HackShield regularly compares the system time of the microprocessor level and the logical time created by the OS. If there is a difference between system time and OS time, the game client will be notified by the callback function.

Depending on your system, the OS, and the characteristics of the game, the extent of speed hack detection may differ. The developer can set the speed hack detection rates in five levels using parameters.

Note

Speed hack refers to a program that manipulates the timer or time-related functions provided by the Windows system.

Debugging Block

HackShield analyzes the game program using the debugger and blocks all debugger tracing to prevent hack attacks. If a debugger tracing attempt is detected, the game client will be notified by the callback function. Even when HackShield is being initialized, a check will be done whether a debugging program such as SoftICE is running. If a debugging program is running, an error will be returned.

Message Hooking Block

HackShield prevents hackers from getting events needed for hacking using the message hooking function.

File Manipulation Block

When HackShield is being initialized and when the game is being executed, the system will check whether the HackShield files are the same as the initially distributed files. If files were manipulated or file names were changed, the game client will be notified.

Caution

The HackShield module does not detect whether the game client program file was manipulated. The developer can check whether the game client program module was manipulated using the Server Side Detection library (AntiCpSvr). If the game client program file was manipulated, the anti-hacking function of HackShield may not work properly.

Auto Mouse Block

HackShield blocks auto mouse to prevent manipulation of the game program and overload in the game server.

2.1.2. Features

Interface Function (API)

HackShield provides interface DLL so the developer can use HackShield functions and view the result data. Using the provided interface DLL, developers can select only the functions that they need.

Sample Program

A test game client program using the APIs offered by HackShield is provided. The game developer can check HackShield's functions and performance and easily develop client programs by referring to the sample program.

Packer

A packer is provided for the developer to encrypt and compress all HackShield executable files and related DLL files and modify HackShield codes. When HackShield is executed, decryption and decompression will occur in the memory.

Smart Update

Smart updates provide up-to-date anti-hacking functions and hacking tool detection engines. The game client can download the latest anti-hacking functions and the hacking tool detection engines through the update server. The update server supports both FTP and HTTP. The FTP can be accessed through anonymous login or user login. When an update is being performed, the developer can check the progress of the update by selecting "Options".

2.1.3. System Architecture

HackShield is provided in a library form using SDK, not as an executable file. HackShield's general architecture and operating principles are as follows:

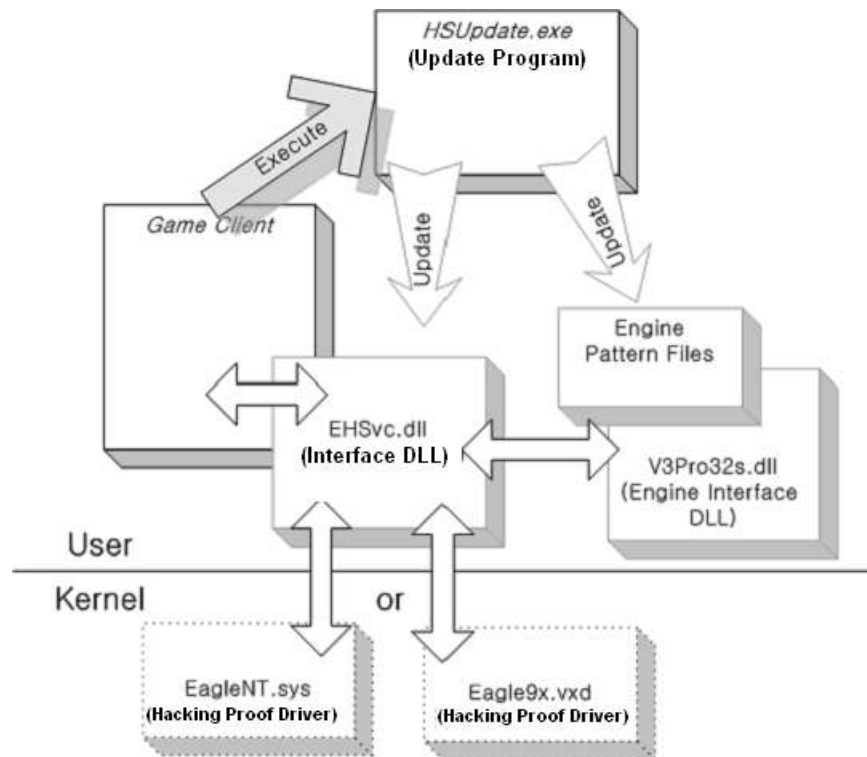


Figure 2-1 Structure of HackShield Pro

HSUpdate.exe (Update Program)

HackShield update program that receives anti-hacking modules and hacking tool detection modules from the update server. If desired, the client may use it only for engine file updates.

EHSvc.dll (Interface DLL)

Initiates the hacking prevention module and hacking tool detection engine. Hacking tool detection and hack blocking results are sent to the game client through the callback function.

V3Pro32s.dll (Engine Interface DLL)

Starts the hacking tool detection engine that uses AhnLab's anti-virus technology. When a hacking tool registered in the engine signature file is executed, this will immediately be detected and the game process will be notified. If a hacking tool, not registered in the engine signature file, is detected, the hacking tool will be immediately registered in the engine signature file and blocked, and the engine signature file will be updated.

Eagle9x.vxd or EagleNT.sys (Anti-hacking Driver)

Driver file operating in the kernel mode and performing anti-hacking function. Included in the EHSvc.dll (interface DLL) file. Depending on the OS, Eagle9x.vxd

or EagleNT.sys will be loaded onto the system.

2.2. Application Programming

This chapter describes how to implement anti-hacking and hacking tool detection functions using the APIs provided by HackShield.

Note

The sample codes in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

2.2.1. Programming Procedure

The game client developer can implement hacking tool detection and anti-hacking functions as follows.

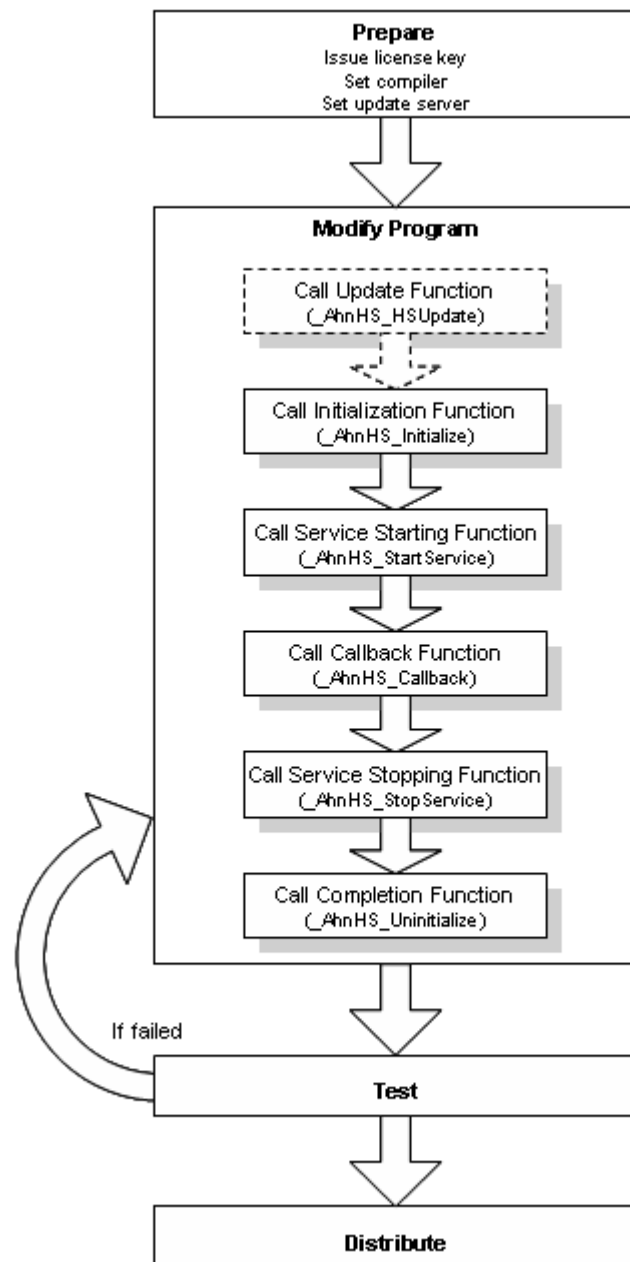


Figure 2-2 Application Programming Procedure

1. Prepare: Check the list of the provided HackShield files and copy the necessary files. To start programming, the developer must have a license key.
2. Check Engine Update Result: Check whether the engine was successfully updated.

Note

When updating the engine using the HackShield update program, the developer must call the update function using the provided update library (HSUpChk.lib).

3. Call Initialization Function: Write a code to call the HackShield initialization function to check file manipulation status and initialize data.
4. Call Service Starting Function: Write a code to call the service starting function to block hack attacks and detect hacking tools.
5. Call Callback Function: Write a callback function that blocks hack attacks and detects hacking tools.
6. Call Service Stopping Function: Add calling of the service stopping function in the part dealing with program exit to stop anti-hacking and hacking tool detection functions.
7. Call Completion Function: Write a code to call the service stopping function and the uninitialization function.
8. Test whether the written source code works properly.
9. Distribute to the game client users.

2.2.2. Preparation

This chapter provides the list of HackShield files to be checked under the installation directory before the programming process, and describes how to set the compiler, how to use the update module, how to set the update server, and how to issue license keys.

2.2.2.1. HackShield-related Files

Installation Directory

The game developer must create the HackShield folder in the game client folder and store HackShield files in the HackShield folder.

<Game Directory>\HShield\

Note

When the initialization function is called, the path of the HackShield file will be returned as a parameter. A function is also provided to check whether HackShield files are being forged/modified during the run-time. To use this function, the developer must create the HackShield folder within the game client folder.

HackShield-related Files

Table 2-1 HackShield-related Files

File Name	Installation Folder	Description
3N.mhe	[HackShield folder]	Heuristic engine File
EhSvc.dll	[HackShield folder]	Interface DLL
HShield.dat	[HackShield folder]	HackShield-related dat file
psapi.dll	[HackShield folder]	Process Status Helper DLL
v3warpds.v3d	[HackShield folder]	Anti-hacking engine Pattern File
v3warpns.v3d	[HackShield folder]	Anti-hacking engine Pattern File
v3pro32s.dll	[HackShield folder]	Hacking Tool detection engine Interface DLL
HShield.h	[game source folder]	Header File
HShield.lib	[game source folder]	Library File

Compiler Setting

The project file of the game client program where HackShield is installed must include the HShield.lib file in the library or source code list. HShield.lib checks the DLL file version to check whether HackShield files are being forged/modified. Version.lib must be added to the compiler link option.

2.2.2.2. Update Module

To regularly update the HackShield engine file, the game developer must use its own patch module or AhnLab's Smart update module.

[Case 1] Game Developer's Patch Module

If HackShield-related files are updated using the game developer's patch module, a separate update module may not be distributed. Only the existing game patch module and HackShield-related files may be distributed.

[Case 2] Smart Update Module

If the HackShield engine file is updated using AhnLab's Smart Update module, the game patch module and the following Smart Update-related modules must be distributed.

Table 2-2 Files in Smart Update Module

File Name	Installation Folder	Description
AhnUpCtl.dll	[HackShield folder]	Update DLL
AhnUpGS.dll	[HackShield folder]	Update DLL
AspINet.dll	[HackShield folder]	Update DLL
Bz32Ex.dll	[HackShield folder]	Update DLL
HSInst.dll	[HackShield folder]	Update DLL
HSUpdate.env	[HackShield folder]	Update configuration file
HSUpdate.exe	[HackShield folder]	Update executable file
mspatcha.dll	[HackShield folder]	Delta update-related file
V3hunt.dll	[HackShield folder]	Update DLL
V3InetGS.dll	[HackShield folder]	Update DLL
HSUpChk.h	[Game source folder]	Header file
HSUpChk.lib	[Game source folder]	Library file

When updating the HackShield engine file using the Smart Update, the installation directory is as follows:

Example - Installation Directory for Smart update Module

<HackShield Directory>\

```
3N.mhe
AhnUpCtl.dll
AhnUpGS.dll
AspINet.dll
Bz32Ex.dll
EhSvc.dll
HShield.dat
HSInst.dll
HSUpdate.env
HSUpdate.exe
mspatcha.dll
psapi.dll
V3Hunt.dll
V3InetGS.dll
```

```

v3pro32s.dll
v3warpds.v3d
v3warpns.v3d

<HackShield Directory>\Update\
ahn.ui
ahn.uic
ahni2.dll
ahnupctl.dll
autoup.exe
v3bz32.dll

```

If the directory of the Smart Update module is updated by AhnLab's Smart Update module, HackShield engine files will also be automatically updated. Depending on the client's requirements, this function may only be used for updating the HackShield engine file.

Table 2-3 Automatic Smart Update Creation Files

File Name	Installation Folder	Description
ahn.ui	[HackShield folder] /Update/	Update Information File
ahn.uic	[HackShield folder] /Update/	Update Information File
ahni2.dll	[HackShield folder] /Update/	Update DLL
ahnupctl.dll	[HackShield folder] /Update/	Update DLL
autoup.exe	[HackShield folder] /Update/	Update Executable file (File Patch)
v3bz32.dll	[HackShield folder] /Update/	Update DLL (File Compression)

2.2.2.3. Update Server Setting

The game developer may use its own patch function for AhnLab's Smart Update module to regularly update the HackShield engine file.

Note

If the game developer has its own patch function, it is recommended that the developer use its game patch to update the HackShield engine.

Note

The update server supports both FTP and HTTP. FTP supports both anonymous login and user login. Passive mode is supported if FTP is used. During an update, the game developer can check the update progress by selecting "options".

[Case 1] Game Developer's Patch Module

When updating the HackShield engine file using the game developer's patch, the following files must be installed in the update server.

Table 2-4 Files to be Installed in Update Server

File Name	Description
3N.mhe	Heuristic engine file
Ehsvc.dll	HackShield interface DLL
HShield.dat	HackShield-related dat file
psapi.dll	Process status helper DLL
V3pro32s.dll	Hacking tool detection engine interface DLL
V3warpds.v3d	Hacking tool pattern engine file
V3warpns.v3d	Hacking Tool pattern engine file

[Case 2] Smart update Module

When updating HackShield using AhnLab's Smart Update module, the game developer must create HackShield Update folder in the update server and install the following files under the folder.

Table 2-5 Files to be Installed in Smart Update Server

File Name	Description
ahn.ui	Update Information File (Engine File Version)
ahn.uic	Update Information File (Engine File Version)
ahni2.dll	Update File
ahnupctl.dll	Update File
autoup.exe	Update File
v3bz32.dll	Update File
win\eb\b\v3_echo_sl\v3pro32s.dl-	Hacking Tool detection engine Interface DLL (Compressed File)
win\eb\b\b_sign_sl\v3warpds.v3-	Hacking Tool Pattern engine File (Compressed File)
win\eb\b\b_sign_sl\v3warpns.v3-	Hacking Tool Pattern engine File (Compressed File)
patch\39\3n.mh-	Heuristic engine File (Compressed File)
patch\39\ahn.ui	Update Information File (Patch File Version)
patch\39\ahn.uic	Update Information File (Patch File Version)
patch\39\ahnupctl.dl-	Update File (Compressed File)
patch\39\ahnupgs.dl-	Update File (Compressed File)
patch\39\aspinet.dl-	Update File (Compressed File)
patch\39\bz32ex.dl-	Update File (Compressed File)
patch\39\ehsvc.dl-	HackShield Interface DLL (Compressed File)
patch\39\hshield.da-	HackShield-related dat file (Compressed File)
patch\39\hsinst.dl-	Update File (Compressed File)
patch\39\hsupdate.ex-	Update Executable file (Compressed File)
patch\39\mspatcha.dl-	Update File (Compressed File)
patch\39\psapi.dl-	Process Status Helper DLL (Compressed File)
patch\39\v3hunt.dl-	Update File (Compressed File)
patch\39\v3inetgs.dl-	Update File (Compressed File)

When updating the HackShield engine file using AhnLab's Smart Update module, the game developer must call the update library function before EhSvc.dll (a HackShield interface DLL) is called and start Smart Update.

When the update of HSUpdate.exe, the executable file of the Smart Update, is completed, the result must be notified as the return value of the update library function. The developer must create a Smart Update file (HSUpdate.env) as described below.

Caution

When using AhnLab's Smart Update module, the developer must execute the game client after Smart Update is completed.

Configuring Update Server

1. Run HSUpSetEnv.exe, the Smart Update module's configuration file. The following window will appear:

The screenshot shows the 'HackShield Update Setup' window. It contains a 'Server List' table with columns: Name, Proto..., IP/URL, Port, ID. Below the table is a 'Delete' button. Below the 'Server List' section is a 'Server Name' text box. Underneath, there are two radio buttons: 'HTTP' and 'FTP'. The 'FTP' radio button is selected. Below the radio buttons are four text boxes: 'Update Server' (for HTTP), 'Update Server' (for FTP), 'User ID', and 'Password'. There is an 'Add' button to the right of the 'User ID' and 'Password' boxes. At the bottom of the window are 'Save' and 'Close' buttons.

Figure 2-3 Update Server Configuration

2. Enter the data in each field by referring to the following table.

Table 2-6 Update Server Configuration

Item	Description
Server Name	Name of update server
Update Method	
Web (HTTP)	Select when using a web server for update. Update Server: Enter the update server address. If the update files are in a certain subdirectory, enter the entire directory path. Enter the port number of the HTTP server in the next field.

	The developer may register multiple update servers by clicking the Add button.
FTP	<p>Select when using FTP server for update.</p> <p>Update Server: Enter the address of the update server. If the update files are in a certain subdirectory, not a root directory, enter the entire directory path. Enter the port number of the FTP server in the next field.</p> <p>User ID: Enter the user account to access the update directory.</p> <p>Password: Enter the password to access the update directory.</p> <p>(For anonymous access, you do not have to enter the user ID and password.)</p> <p>The developer may register multiple update servers by clicking the Add button.</p>

Checking Update Results

Call the update function using the provided update library, and check the update results.

Call the update function as follows:

Example

```
nResult = AhnHS_HSUpdate(szHSDir, 600000);
```

The update function must be called before the HackShield Initialization function to prevent the HackShield module from being updated.

If a launcher or update program is running before the game client with HackShield installed is started, call the update function in the program.

Caution

EhSvc.dll file is released with the HShield.dat file. Therefore, these two files must always use the same version.

Note

For more information about HackShield update, refer to [8. HackShield update functions](#).

2.2.2.4. Receiving License Key

To use the APIs provided by HackShield, the game developer must have a license key issued by AhnLab.

The game developer may receive a license key as follows:

1. Send the name of the executable file that uses EhSvc.dll, the game developer's name, and the name of the game program to AhnLab and request a license key.
2. A unique (four-digit) game code and a 24-digit license key will be issued.
3. Use the issued game code and the license key as the parameters of the Initialization function `_AhnHS_Initialize`.

Caution

If incorrect data is sent as the parameters for the initialization function `_AhnHS_Initialize`, the initialization function will not be properly called and an error (`HS_ERR_INVALID_LICENSE`) will be returned.

2.2.3. Calling Initialization Function

After preparation for programming is completed, call the initialization function `_AhnHS_Initialize`. The initialization function must be called before the anti-hacking and hacking tool detection functions can be executed.

The initialization function is called by the routine that initializes the game client program instance or the main window.

The following is an example of the initialization function. In the following example, the last parameter of the initialization function is `AHNHS_CHKOPT_ALL` so that all options are used.

(There is a core option¹ that is automatically applied even if it isn't selected by the game developer. The core option is automatically applied even if it isn't selected by the user.)

Example

```
nResult = _AhnHS_Initialize(
    szFullFileName,
    AhnHS_Callback,
    1000, "B228F2916A48AC24",
    AHNHS_CHKOPT_ALL,
    AHNHS_SPEEDHACK_SENSING_RATIO_NORMAL
);
```

Caution

Initialization function `_AhnHS_Initialize` may be called only once for each process. Calling the initialization function more than once may cause an error.

After the initialization function is called, the CRC data must be checked to verify the validity of HackShield interface DLL file (`EhSvc.dll`) and to see whether HackShield files are being forged/modified. A function that checks the size and version of the DLL file must be called to check whether the files are being forged/modified.

Note

HackShield provides encryption and compression functions for the interface DLL file (`EhSvc.dll`) using the packer to prevent file manipulation. If a change is made during the game, it will be detected. If a HackShield DLL file is changed, the game process will be notified through the Callback function.

Calling the initialization function when the `EhSvc.dll` file is manipulated or the file version is not correct will cause an error (`HS_ERR_INVALID_FILES`).

Note

The part that calls the HackShield function in the game client program (not the

¹ Core option: `AHNHS_CKOPT_SPPEDHACK` | `AHNHS_CHKOPT_READWRITEPROCESSMEMORY` | `AHNHS_CHKOPT_KDTRACER` | `AHNHS_CHKOPT_OPENPROCESS` | `AHNHS_CHKOPT_AUTOMOUSE` | `AHNHS_CHKOPT_PROCESSSCAN`

HackShield interface DLL file EhSvc.dll) could get forged/modified. Therefore, it is recommended that the game client files are encrypted, compressed, and distributed by the provided packer program to prevent manipulation of game client program files.

However, HackShield Pro provides a function that uses the server interface client crack to prevent the game from being started by a cracked executable file.

Some users and hackers may attack the program by running a program in the lower compatibility mode in Windows XP. If a program runs on the lower compatibility mode, Windows XP may be perceived by the system as Windows 98, ME, or Windows 2000, bringing unexpected results. If the program is running on a lower compatibility mode and the initialization function of the HackShield module is called, an error (HS_ERR_COMPATIBILITY_MODE_RUNNING) will be returned.

When the initialization function is called, the following six options in addition to the inspection option may be selected.

AHNHS_USE_LOG_FILE

Stores log files to deal with potential errors.

AHNHS_ALLOW_SVCHOST_OPENPROCESS

Allows svchost.exe to properly operate when OpenProcess() prevents svchost.exe, a system process of Windows NT, from calling OpenProcess() for the game process.

AHNHS_ALLOW_LSASS_OPENPROCESS

Allows lsass.exe to properly operate when OpenProcess() prevents lsass.exe, a system process of Windows NT, from calling OpenProcess() for the game process.

AHNHS_ALLOW_CSRSS_OPENPROCESS

Allows csrss.exe to properly operate when OpenProcess() prevents csrss.exe, a system process of Windows NT, from calling OpenProcess() for the game process.

AHNHS_DONOT_TERMINATE_PROCESS

Prevents system crashes due to bugs in the hacking tool when the hacking tool is terminated by the real-time hacking tool inspection function, and notifies only the hacking tool detected events to the game process. The game programmer must decide which measures to take once the game process receives detected events. If the hacking tool is forcibly terminated, system problems may occur due to abnormal termination of the hacking tool. It is recommended that the AHNHS_CHKOPT_PROCESSCAN option is also used.

AHNHS_DISPLAY HACKSHIELD LOGO

Displays the HackShield logo during HackShield initialization.

AHNHS_CHKOPT_PROTECTSCREEN

This option is recommended only for games that support the full screen and Alt + Tab prevention functions. Setting this option can help prevent hack attacks that steal screen information from the game.

Caution

Using this option may damage the user interfaces of other programs. Before

using this option, contact the AhnLab Technical Support Team.

2.2.4. Calling Service Starting Function

To start the HackShield anti-hacking and hacking tool detection functions, the game developer must call the service starting function `_AhnHS_StartService` as shown below:

Example

```
nResult = _AhnHS_StartService( );
```

The service starting function `_AhnHS_StartService` must be called after the initialization function `_AhnHS_Initialize` has been properly called.

Caution

To ensure that the game client program is secure, it is recommended to call the service starting function as soon as the initialization function is called. The later the service starting function is called, the more likely that a hacking tool will attack the game client.

2.2.5. Writing Callback Function

Calling the service starting function will report the current service state as an event. Add each event handling method to the callback function as follows:

Example

```
int __stdcall _AhnHS_Callback( long lCode,
                              long lParamSize,
                              void* pParam )
{
    switch(lCode)
    {
        case _AHNHS_ENGINE_DETECT_GAME_HACK:
            break;
        case _AhnHS_ACTAPC_DETECT_KDTRACE:
            break;
    }
}
```

Caution

Do not write a code such as that calling the callback function is taken as the cause of a hack attack. The callback code may be applicable only to certain games or may return incorrect detection resulting in error codes. See the callback functions in the sample code in the *HackShield SDK Package or contact the AhnLab Technical Support Team.

2.2.6. Calling Service Stopping Function

Before stopping the game client program, call the service stopping function `_AhnHS_StopService` to stop the anti-hacking and hacking tool detection functions.

When pausing the HackShield service without stopping the game client program, the developer must call the service stopping function `_AhnHS_StopService` and call the service starting function `_AhnHS_StartService`.

Example

```
nResult = _AhnHS_StopService ( );
```

2.2.7. Calling Service Uninitializing Function

To completely uninitialized the game Client program, call the service uninitializing function `_AhnHS_Uninitialize` as shown in the following example and free the memory allocated for the program.

Example

```
nResult = _AhnHS_Uninitialize( );
```

Once the service uninitializing function `_AhnHS_Uninitialize` is called, the game client can no longer be protected by HackShield. Therefore, the game developer must call the uninitializing function at the last point when the game client is uninitialized.

Caution

Improper uninitialization of the game client program may prevent the HackShield anti-hacking driver from unloading. In this case, calling the initialization function `_AhnHS_Initialize` will return an error (`HS_ERR_INIT_DRV_FAILED`) because the anti-hacking driver of HackShield is already loaded in the user's system. In this case, the developer must reboot the system and restart the game client program.

2.3. Application Programming Interface

AhnHS_Initialize

DESCRIPTION

Initializes HackShield and configures options. Can be called only once when the program is initialized. An error may be returned if another game program uses HackShield or if the service is abnormally stopped.

SYNTAX

```
EHSVC_API
int __stdcall
_AhnHS_Initialize(
    const char* szFileName
    PFN_AhnHS_Callback pfn_Callback,
    int nGameCode,
    const char* szLicenseKey,
    DWORD dwOption,
    UINT unSHackSensingRatio
);
```

PARAMETERS

Parameter	Value	Description
szFileName		Full path of EHSvc.dll
Pfn_Callback		Callback function pointer
nGameCode	4-digit numeric code	Unique code of each game
szLicenseKe	24-digit character string	License key for each game
dwOption		Initialization option setting
unSHackSensingRatio		Speed hack detection rate level

OPTIONS

Caution

The following shows core options that are automatically applied.

```
AHNHS_CHKOPT_SPEEDHACK
AHNHS_CHKOPT_READWRITEPROCESSMEMORY
AHNHS_CHKOPT_KDTRACER, AHNHS_CHKOPT_OPENPROCESS
AHNHS_CHKOPT_AUTOMOUSE, AHNHS_CHKOPT_PROCESSSCAN
```

AHNHS_CHKOPT_SPEEDHACK (Core Option)

Uses the speed hack blocking function. Reports a speed hack through the callback function if a hardware control or software-type speed hack is detected. The sensitivity of the detection depends on the detection rate level. Refer to the speed hack sensing ratio.

AHNHS_CHKOPT_READWRITEPROCESSMEMORY (Core Option)

Prevents the memory of the current process using HackShield DLL from being used by other processes.

AHNHS_CHKOPT_KDTRACER (Core Option)

Detects a kernel mode debugger, and informs it to the game process.

AHNHS_CHKOPT_OPENPROCESS (Core Option)

Blocks calling of the OpenProcess API function that provides information on the process currently using HackShield DLL.

AHNHS_CHKOPT_AUTOMOUSE (Core Option)

Prevents auto mouse from affecting currently running processes.

AHNHS_CHKOPT_PROCESSSCAN (Core Option)

Regularly checks the process list and detects hacking tools.

AHNHS_CHKOPT_MESSAGEHOOK

Blocks message hooking.

AHNHS_CHKOPT_ALL

Includes all options described above.

AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION

Protects the memory of the protection target file.

AHNHS_USE_LOG_FILE

Stores logs related to HackShield operation. Log files are created as hshield.log in the same directory that contains EhSvc.dll. Log files are used for error analysis. Because log files are encrypted, the contents can be viewed only by a separate program.

AHNHS_ALLOW_SVCHOST_OPENPROCESS

Makes it possible to open the game process in SvcHost, a service program running on the NT-series OS. In Windows XP, the newly added Windows audio service is used for sound output. The Windows audio service opens the client process, and selecting the option to block OpenProcess will prevent proper sound output. Add this option when sound output by DirectMusic or DirectSound does not work properly on Windows XP.

AHNHS_ALLOW_LSASS_OPENPROCESS

Allows LSASS.exe, a service program running on the NT-series OS, to open the game process. If the game client includes credit card payment control, LSASS.exe will directly access the game process. The game developer may open the game process using this option.

AHNHS_ALLOW_CSRSS_OPENPROCESS

Allows CSRSS.exe, a service program running on the NT-series OS, to open the game process. If the game client includes credit card payment control, CSRSS.exe will directly access the game process. The game developer may open the game process using this option.

AHNHS_DONOT_TERMINATE_PROCESS

When the process list is inspected to detect hacking tools, the default setting is to forcibly stop the hacking tool process. However, the game developer may choose not to forcibly stop the hacking tool process using this option. It is recommended that this option is used together with the AHNHS_CHKOPT_PROCESSSCAN option.

AHNHS_DISPLAY_HACKSHIELD_LOGO

Displays the HackShield logo during initialization. Operates as a separate thread, and the logo will automatically disappear in two weeks. Used to show that HackShield is running.

SPEEDHACK SENSING RATIO

AHNHS_SPEEDHACK_SENSING_RATIO_HIGHEST

The most sensitive level with reference data of 14.5% and -12.5%. When the speed is greater than 1.1487 times or less than 0.8706 times that of A Speeder Speed Hack program, it will be diagnosed as a speed hack.

AHNHS_SPEEDHACK_SENSING_RATIO_HIGH

The second most sensitive level with reference data of 17.5% and -15.5%. When the speed is greater than 1.1892 times or less than 0.8409 times that of A Speeder Speed Hack program, it will be diagnosed as a speed hack.

AHNHS_SPEEDHACK_SENSING_RATIO_NORMAL

The second most sensitive level with reference data of 17.5% and -21%. When the speed is greater than 1.2746 times or less than 1.2746 times that of A Speeder Speed Hack program, it will be diagnosed as a speed hack.

AHNHS_SPEEDHACK_SENSING_RATIO_LOW

The second most sensitive level with reference data of 31.5% and -23.5%. When the speed is greater than 1.3195 times or less than 0.7579 times that of A Speeder Speed Hack program, it will be diagnosed as a speed hack.

AHNHS_SPEEDHACK_SENSING_RATIO_LOWEST

The second most sensitive level with reference data of 39.0% and -28.5%. When the speed is greater than 1.4142 times or less than 1.4142 times that of A Speeder Speed Hack program, it will be diagnosed as a speed hack.

RETURN VALUE

HS_ERR_OK (Value = 0x0000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

HS_ERR_INVALID_PARAM(Value = 0x002)

- Description: Incorrect parameter.
- Cause: Incorrect setting of callback function pointer, or null license key value.
- Solution: Only occurs during the development process. No special handling is required.

HS_ERR_COMPATIBILITY_MODE_RUNNING (Value = 0x004)

- Description: The game was executed in the compatibility mode.
- Cause: Occurs when the game client runs in compatibility mode in the Windows XP series.
- Solution: Users running the game in compatibility mode will be considered malicious users. The program must be forcibly stopped program and restarted.

HS_ERR_INVALID_LICENSE (Value = 0x100)

- Description: Incorrect license.
- Cause: The game code set as the parameter for the initialization function and the license key do not match the actual values.
- Solution: Check whether the license key was properly entered.

HS_ERR_INVALID_FILES (Value = 0x101)

- Description: Incorrect HackShield file.
- Cause: Occurs when the interface DLL file (EhSvc.dll) is manipulated or when the version is different.
- Solution: Check if the HackShield file is a previous version or if the file exists in the HackShield folder.

HS_ERR_INIT_DRV_FAILED (Value = 0x102)

- Description: The HackShield driver was not started.
- Cause: Occurs when the anti-hacking driver was not initialized. In this case hack attacks are not properly blocked. The developer must restart the program.
- Solution:
 - ① Check if the authority of EagleNT.sys file was changed in the System folder, making it inaccessible.
 - ② Check that the EagleNT.sys file is not unloaded and is running.

If one of the above situations arises, contact the AhnLab HackShield Technical Support Team.

HS_ERR_ALREADY_INITIALIZED (Value = 0x104)

- Description: HackShield has been initialized.
- Cause: Occurs when _AhnHS_Initialize was called and the system is initialized.
- Solution: _AhnHS_Initialize can be called only once when the program is initialized. This function cannot be called several times. Check that the above function was not called several times.

HS_ERR_DEBUGGER_DETECT (Value = 0x105)

- Description: A debugger was detected.
- Cause: Occurs when a debugger is running in the system.
- Solution:
 - ① Perform debugging using the Dev Version, not a release version, from the development process. (Refer to “Remarks”.)
 - ② If this error occurs, a hack attack may occur. Terminate the game and stop the debugger. Then, restart the game.

If the error occurs when there is no debugger, contact the AhnLab Technical Support Team.

HS_ERR_NEED_ADMIN_RIGHTS (Value = 0x107)

- Description: Admin account is needed.
- Cause: Occurs if the game client is executed by a general user account, not an administrator account, on a Windows NT-series system.
- Solution: If the game client was executed by a user account, check whether a shadow account has been created. To use the HackShield functions with a general user account, a HackShield shadow account must be created first. For more information, see 'User Authority Support. **User Authority Support**'.

HS_ERR_UNKNOWN (Value = 0x001)

- Description: Unknown error has occurred.
- Cause: Occurs when an unknown error arises.
- Solution: If the trouble persists, contact the AhnLab Technical Support Team.

REMARKS

To debug the game client during the development or test process, use HShield.lib, Ehvsc.dll, and HShield.dat designed for developers.

(When Ehsvc.dll and HShield.dat files of different release versions are used together, server interfaces may not function properly.)

- Path: \Developer\

If the developer uses a developer module, the following logo will be displayed when HackShield is executed. (The developer can hide the logo by clicking on it.)



AhnHS_StartService

DESCRIPTION

Runs the hacking tool detection and anti-hacking functions. Must be called after the `_AhnHS_Initialize` function is called, and cannot be called more than once. If the service was stopped by the `_AhnHS_StopService` function, the function must be called again to restart the service.

SYNTAX

```
EHSVC_API  
int __stdcall  
_AhnHS_StartService( );
```

PARAMETERS

None.

RETURN VALUE

HS_ERR_OK (Value = 0x000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- Description: HackShield was not initialized.
- Cause: `_AhnHS_Initialize` function was not called. Or the function was called while HackShield was not initialized.
- Solution: This error occurs only in the development process, so a special solution to this problem is not needed.

HS_ERR_START_ENGINE_FAILED (Value = 0x200).

- Description: Engine loading failed.
- Cause: Occurs when initialization of the hacking tool pattern engine fails although the hacking tool process detection option (`AHNHS_CHKOPT_PROCESSCAN`) was set when the initialization function was called. Also occurs when the hacking tool pattern engine-related files were not properly installed or the pattern engine-related files (`v3pro32s.dll`, `v3warps.v3d`, and `v3warps.v3d`) were not installed in the HackShield folder.

- Solution: Forcibly stop the program, and restart or reinstall the program.

HS_ERR_ALREADY_SERVICE_RUNNING (Value = 0x201)

- Description: HackShield is already running.
- Cause: Occurs when the `_AhnHS_StartService` function is called while the function is already running.
- Solution: To call this function again, the developer must stop the service first by calling `_AhnHS_StopService`. This error only occurs in the development process so a special solution to this problem is not needed.

HS_ERR_DRV_FILE_CREATE_FAILED (Value = 0x202)

- Description: Creation of HackShield driver file failed.
- Cause: Occurs when the driver file for anti-hacking cannot be created. The HackShield program includes an anti-hacking driver in the DLL file that is loaded when the game is started. This error occurs when the driver file is not properly created when the game starts.
- Solution:
 - ① Check if the session has writing authority for the System folder.
 - ② Check if the HackShield driver file (EagleNT.sys) in the System folder is read-only or inaccessible. In this case, delete the file and restart HackShield.
(If the trouble persists, contact the AhnLab Technical Support Team.)

HS_ERR_REG_DRV_FILE_FAILED (Value = 0x203)

- Description: Registration of the HackShield drive file failed.
- Cause: Occurs when registration of the driver file for anti-hacking fails.
- Solution: If this error occurs, HackShield will not properly function. In this case, the developer must forcibly terminate the program, and restart or reinstall the program.

HS_ERR_START_DRV_FAILED (Value = 0x204)

- Description: Running of the HackShield drive failed.
- Cause: Occurs when the game client shuts down improperly and is restarted without calling the `_AhnHS_StopService` or the `_AhnHS_Uninitialize` functions, causing the originally loaded driver to stop or an incorrect driver to be loaded. If this error occurs, the system may be unstable, so the developer must terminate the program and restart the system.
- Solution:
 - ① Check whether the current session has writing authority for the System folder.
 - ② Check whether the HackShield driver file (EagleNT.sys) in the

System folder is read-only or inaccessible. If it is, delete the file and restart HackShield.
(If this trouble persists, contact AhnLab's Technical Support Team.)

DESCRIPTION

Stops anti-hacking and hacking tool detection functions.

SYNTAX

```
EHSVC_API  
int __stdcall  
_AhnHS_StopService( );
```

PARAMETERS

None.

RETURN VALUE

HS_ERR_OK (Value = 0x0000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- Description: HackShield is not initialized.
- Cause: Occurs if _AhnHS_Initialize function was not called or the function was called while HackShield was not initialized.
- Solution: This error only occurs in the development process so a special solution to this problem is not needed.

HS_ERR_SERVICE_NOT_RUNNING (Value = 0x301)

- Description: HackShield has not started.
- Cause: Occurs if the _AhnHS_StartService function is called while HackShield was not started. This error occurs only during the development process and special error handling is not required.
- Solution: This error occurs only during the development process so a special solution to this problem is not needed.

AhnHS_Uninitialize

DESCRIPTION

Releases the occupied memory in the system and initializes the parameters.

SYNTAX

```
EHSVC_API  
int __stdcall  
_AhnHS_Uninitialize ( );
```

PARAMETERS

None.

RETURN VALUE

HS_ERR_OK (Value = 0x0000)

- Description: Returned when the function is successfully called.
- Cause:
- Solution:

HS_ERR_SERVICE_STILL_RUNNING (Value = 0x302)

- Description: HackShield is running.
- Cause: Occurs when the _AhnHS_StopService function is called with HackShield not terminated. This error occurs only in the development process so a special solution to this problem is not needed.
- Solution:

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- Description: HackShield is not initialized.
- Cause: Occurs when the _AhnHS_Initialize function is not called or is called while HackShield is not initialized.
- Solution: This error occurs only during the development process so a special solution to this problem is not needed.

DESCRIPTION

Temporarily stops some HackShield functions. Affects only the keyboard protection function among the message hooking prevention functions. Used to temporarily stop the keyboard protection function that blocks the user from entering on a Microsoft Internet Explorer website for payment while a game is running. After temporarily stopping the function, the developer must reactivate the function using `_AhnHS_ResumeService`.

SYNTAX

```
EHSVC_API
int __stdcall
_AhnHS_PauseService (
    DWORD dwPauseOption
);
```

PARAMETERS

Parameter	Value	Description
dwPauseOption	DWORD	Currently, only the <code>AHNHS_CHKOPT_MESSAGEHOOK</code> option can be used. If other options are sent, an <code>HS_ERR_INVALID_PARAM</code> error will be returned.

RETURN VALUE

HS_ERR_OK (Value = 0x000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- Description: HackShield is not initialized.
- Cause: Occurs if the `_AhnHS_Initialize` function was not called or if the function was called while HackShield was not initialized.
- Solution: This error occurs only during the development process so a special solution to this problem is not needed.

HS_ERR_SERVICE_NOT_RUNNING (Value = 0x301)

- Description: HackShield has not started.
- Cause: Occurs when the _AhnHS_StartService function was called when HackShield was not started. This error occurs only during the development process and special error handling is not required.
- Solution: This error occurs only during the development process so a special solution to this problem is not needed.

HS_ERR_INVALID_PARAM (Value = 0x002)

- Description: Wrong parameters.
- Cause: Occurs when the dwPauseOption value does not equal AHNHS_CHKOPT_MESSAGEHOOK.
- Solution:

AhnHS_ResumeService

DESCRIPTION

Calls AhnHs_Pause service and resumes HackShield functions. Applicable only to the keyboard protection function among message hooking prevention functions. For more information about this function, see _AhnHS_PauseService part.

SYNTAX

```
EHSVC_API
int __stdcall
_AhnHS_ResumeService (
    DWORD dwResumeOption
);
```

PARAMETERS

Parameter	Value	Description
dwResumeOption	DWORD	Currently only the AHNHS_CHKOPT_MESSAGEHOOK option can be used. If other options are returned, an HS_ERR_INVALID_PARAM error will be returned.

RETURN VALUE

HS_ERR_OK (Value = 0x000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- Description: HackShield is not initialized.
- Cause: The _AhnHS_Initialize function was not called or the function was called while HackShield was not initialized.
- Solution: This error occurs only during the development process so a special solution to this problem is not needed.

HS_ERR_SERVICE_NOT_RUNNING (Value = 0x301)

- Description: HackShield has not started.
- Cause: Occurs if the _AhnHS_StartService function is called when HackShield is not started..

- Solution: This error occurs only during the development process so a special solution to this problem is not needed.

HS_ERR_INVALID_PARAM (Value = 0x002)

- Description: Wrong parameters.
- Cause: Occurs when dwPauseOption is not AHNHS_CHKOPT_MESSAGEHOOK.
- Solution:

PFN_AhnHS_Callback

DESCRIPTION

Callback function that sends the result of the anti-hacking and hacking tool detection functions. Returns the following events.

Events related to anti-hacking

Events related to hacking tool detection

STRUCTURE

```
typedef int  
(__stdcall* PFN_AhnHS_Callback) (  
    long lCode,  
    long lParamSize,  
    void* pParam  
);
```

PARAMETERS

Parameter	Value	Description
lCode	Long	Event Code
lParamSize	Long	Event Parameter Size
pParam	Void*	Event Parameter

EVENTS

AHNHS_ENGINE_DETECT_GAME_HACK

If a game hacking tool is detected in the system, this event will be reported by the callback function. The game developer can forcibly delete the executable file of the hacking tool or take other measures based on the information received about the hacking tool.

pParam: Name of the game hacking tool's executable file (including the file path)

lParamSize: Length of the executable file name of the detected hacking tool

AHNHS_ACTAPC_DETECT_ALREADYHOOKED

If the API function to be protected from anti-hacking is hooked while the anti-hacking function is running, this event will be returned. Hooking may occur in a normal program, not a hacking tool, so the game developer must decide the criteria for detecting the hacking tool.

pParam: ACTAPCPARAM_DETECT_HOOKFUNCTION*

lParamSize: Length of ACTAPCPARAM_DETECT_HOOKFUNCTION structure

ACTAPCPARAM_DETECT_HOOKFUNCTION

```
struct _ACTAPCPARAM_DETECT_HOOKFUNCTION
{
    char szFunctionName[128];
    char szModuleName[128];
} ACTAPCPARAM_DETECT_HOOKFUNCTION,
*PACTAPCPARAM_DETECT_HOOKFUNCTION;
```

szFunctionName: Name of the hooked function

szModuleName: Name of the hooked module

AHNHS_ACTAPC_DETECT_HOOKFUNCTION

When the Win32 function or protection function is hooked while the hooking blocking function is running, this event will be returned. When this event occurs, a hacking tool invasion probably occurred. The game developer must forcibly terminate the game program.

pParam: ACTAPCPARAM_DETECT_HOOKFUNCTION*

lParamSize: ACTAPCPARAM_DETECT_HOOKFUNCTION length of structure

AHNHS_ACTAPC_DETECT_AUTOMOUSE

This event occurs when an auto mouse program automatically enters data by manipulating the keyboard or mouse.

pParam – ACTAPCPARAM_DETECT_AUTOMOUSE

lParamSize – Size of ACTAPCPARAM_DETECT_AUTOMOUSE. The following structure pointer will be sent.

ACTAPCPARAM_DETECT_AUTOMOUSE

```
typedef struct
{
    BYTE   byDetectType;
    DWORD  dwPID;
    CHAR   szProcessName[16+1];
    CHAR   szAPIName[128];
} ACTAPCPARAM_DETECT_AUTOMOUSE,
*PACTAPCPARAM_DETECT_AUTOMOUSE;
```

The following describes byDetectType. dwPID, szProcessName, and szAPIName are not currently in use.

EAGLE_AUTOMOUSE_APCTYPE_SHAREDMEMORY_ALTERATION (3): changes the internal data of the library that blocks the auto mouse function. If a hacking program modified the HackShield hacking prevention data, the anti-hacking function will not work properly. In this case, the developer must forcibly terminate the game program.

EAGLE_AUTOMOUSE_APCTYPE_API_CALLED: Calls APIs related to keyboard and mouse.

EAGLE_AUTOMOUSE_APCTYPE_API_ALTERATION: API hooking manipulation

AHNHS_ACTAPC_DETECT_AUTOMACRO

This event occurs when an auto mouse program automatically enters data by manipulating the keyboard or mouse. Abnormal data is repeatedly entered, so the game program must be forcibly terminated.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_DRIVERFAILED

This event occurs if the anti-hacking driver was not loaded onto the system. If the anti-hacking driver was not loaded onto the system, the anti-hacking function will not operate properly. In this case, the game program must be immediately terminated. If the driver was improperly removed, the system may become unstable. In this case, the system must be restarted.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_SPEEDHACK

Occurs when the time changing speed in the system is abnormal. It is very likely that a hardware or software-type speed hack is running. If the game program is sensitive to time, it must be forcibly terminated. Hardware-type speed hacks directly manipulate hardware time, affecting the entire system. This kind of hardware-type speed hacks sometimes block the timer at the OS level depending on the Windows version.

pParam: (double *) Time data for the last five seconds

IPParamSize: Data count sent by pParam

AHNHS_ACTAPC_DETECT_MESSAGEHOOK

This event occurs when message hooking by a hacking program or other program is not blocked. Message hooking is one of the initial attempts made by hacking programs or utility programs to insert their codes in the game. If message hooking is blocked, the game developer can prevent hacking programs from progressing to the next stage.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_KDTRACE

This event occurs when a debugger trace is generated by a kernel-level or application-level debugger. In this case, it is very likely that a malicious user such as a hacker is debugging the game program. It is recommended that the game program be terminated when this event occurs.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_ABNORMAL_MEMORY_ACCESS

This event occurs when an unauthorized process accesses the game process memory. The process can be identified because the path and the name of the process trying to access the memory are visible.

pParam: Name of the detected game hacking tool's executable file (including the file path)

IPParamSize: Length of the detected hacking tool's executable file name

AHNHS_ACTAPC_DETECT_KDTRACE_CHANGED

This event occurs when the debugger trace blocking routine is changed. This means that the debugger program at the kernel level is running while the debugger trace blocking routine for the game program is activated. The game program must be forcibly terminated.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_MODULE_CHANGE

This event occurs when manipulation of the HackShield module is detected. In this case, HackShield may not properly operate, and the game program must be forcibly terminated.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_ENGINEFAILED

This event occurs if the HackShield heuristic engine file (3N.mhe) was deleted or improperly loaded. In this case, the HackShield ProcessScan function may not properly operate, and the game program must be forcibly terminated.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_CODEMISMATCH

This event occurs if the code area of Ehsvc.dll module (the HackShield interface module) is manipulated. In this case, HackShield functions may not work properly, and the game program must be forcibly terminated.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP

This event occurs when the memory area of the protected file is manipulated. In this case, the corresponding module may not operate properly, and the game program must be forcibly terminated.

pParam: Name of the manipulated module (Manipulated page starting address)

IPParamSize: 0

3. Server Side Detection (For HackShield Pro)

3.1. Overview

AntiCPSvr is a software development kit developed by AhnLab in 2005 to detect file/memory manipulation through server interfaces. Recent developments in computer and hacking technologies require strong crack prevention measures. Against this backdrop, checking the integrity of both servers and clients is emerging as a way to ensure safe program execution. AntiCPSvr monitors game cracks and HackShield's operational status to provide a fair gaming environment for users.

3.1.1. Functions

File Integrity Verification

Apply AntiCpSvr.dll and AntiCpSvrFunc.h in the server, and the HShield.lib and HShield.h, HackShield modules, in the client. The game developer can check whether the game's executable file is being manipulated in real time through communication between the server and client.

Packet Integrity

To prevent hack attacks that capture and generate packets in certain circumstances, a separate internal module exists to guarantee packet integrity. This allows the game developer to block packet capture and creation on the network. However, this function protects only the messages related to preventing server interface cracks.

Checking operational Status of HackShield

Apply AntiCPSvr.dll and AntiCpSvrFunc.h in the server, and the HShield.lib and HShield.h, HackShield modules in the client. The game developer can check whether the game's executable file is being manipulated in real time through communication between the server and client. The server must send a query message only after HackShield starts.

Memory Integrity

Apply AntiCpSvr.dll and AntiCpSvrFunc.h in the server, and the HShield.lib and HShield.h, HackShield modules in the client. The game developer can check whether the game's executable file is being manipulated in real time through communication between the server and client.

HackShield Engine Integrity Checking

Apply AntiCPSvr.dll and AntiCpSvrFunc.h in the server, and the HShield.lib and HShield.h, HackShield modules, in the client. The game developer can verify the integrity of the engine files (v3warpds.v3d and v3warpns.v3d) through communication between the server and client.

3.1.2. Features

Interface Function (API)

Interface DLLs and a library are provided to allow the game developer to easily use the AntiCPSvr functions and check the return values. The game developer can check the integrity of game files and HackShield's operation according to common policy.

File Information Creation Program

AntiCPSvrTool.exe provided by AntiCPSvr creates and stores file data and memory information which the client program uses to check file/memory integrity through communication between the client and server. File data and memory information can be stored in the server and this information is used in the file/memory crack prevention function through server interfaces. For more information, see '11.4. **AntiCpSvr Tool**'.

Test Program

Amazon.exe is a test program that uses the APIs provided by AntiCPSvr. Amazon.exe provides HackShield testing and AntiCPSvr test functions.

3.1.3. System Architecture

AntiCPSvr provides AntiCpSvr.dll, HShield.lib, and EHSvc.dll, and is applied in DLL and library format in the server and client. AntiCPSvr's general architecture and operating principles are as follows:

AntiCPSvr.dll (Interface dll)

Used by the server. Provides an API that creates request messages and checks client file and memory status based on the response message from the client.

HShield.lib (HackShield Library)

Used by the client. Provides an API that receives request messages from the server, encrypts the response data, and sends the encrypted response to the server.

AntiCpSvrTool.exe (File Information Creation Program)

Creates file data and memory information for the server to check the integrity of the game file and the memory integrity of the game. Different file data is created for each new file, and if multiple servers exist, each server must have different file data for greater security. Memory data is automatically created when the game program is executed while AntiCpSvrTool.exe is running.

HackShield.crc (Client CRC File)

Created by AntiCpSvrTool.exe. Designed to guarantee integrity of the client program.

HSPub.key (Server Interface Authentication Key File)

Used to authenticate the HackShield program used by the accessing client. Must be in the same folder that contains the HackShield.crc file.

3N.mhe (Heuristic engine File)

An engine file used by the client. If it is stored in the same folder containing the server's HackShield.crc file, clients using earlier versions of the 3N.mhe engine will be disconnected.

HShield.dat (HackShield Version File)

HackShield version management file used by the client. When stored in the same folder containing the server's HackShield.crc file, clients using earlier versions of HackShield will be disconnected.

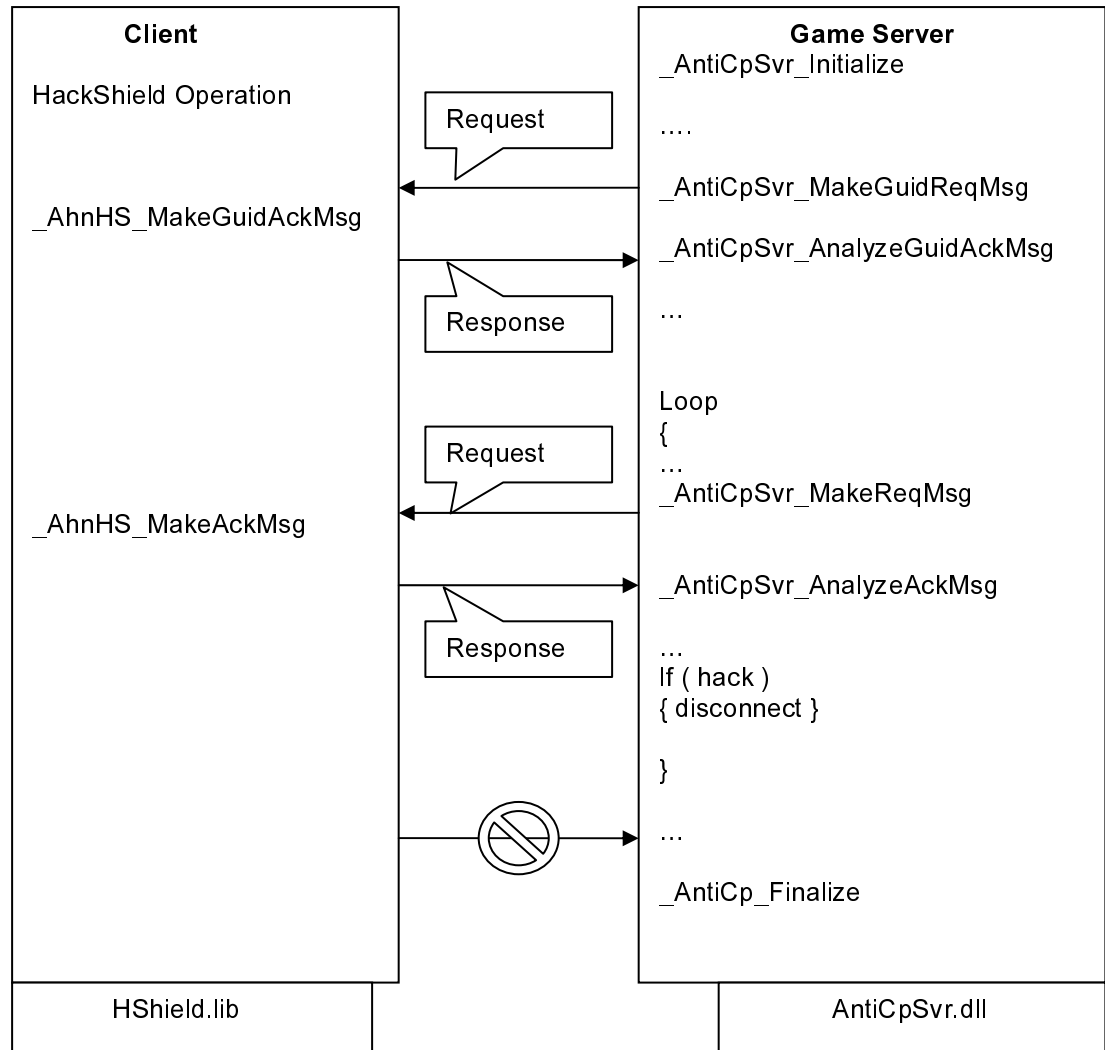


Figure 3-1 Operating Principles of AntiCpSvr

When the game client accesses the server, the server sends a GUID request message to check which version of the executable file is being used by the client, and receives a response message from the server. `_AntiCpSvr_AnalyzeGuidAckMsg` API verifies if the version is allowed by the server. If the version is allowed, the connection must be maintained. Otherwise, the connection must be released to prevent the game from running without the latest patch.

The game server regularly checks the request message and the response message according to the game developer's policy and checks for cracks in the executable file, memory, HackShield module, and engines. If a response message is not sent to the game server, this will be considered a hack attack.

3.2. Application Programming

This chapter describes how to check file and memory integrity using the APIs of AntiCPSvr.

Note

The sample codes in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

3.2.1. Programming Application

Do the following before programming with AntiCPSvr:

3.2.1.1. AntiCPSvr-related File

AntiCPSvr-related Files

Table 3-1 AntiCPSvr-related File

File Name	Installation Folder	Description
AntiCPSvrfunc.h	[Program folder] source	Header file to be used in the server
AntiCPSvr.dll	[Program folder] source	DLL file to be used in the server
AntiCPSvr.lib	[Program folder] source	Import library of AntiCPSvr.dll
HackShield.crc	[Program folder] execution	Client integrity verification file
HSPub.key	[Program folder] execution	Server Interface Authentication Key File
HShield.h	[Program folder] source	Header file to be used in the client. HackShield functions included.
HShield.lib	[Program folder] source	Library file to be used in the client. HackShield functions included.
EHSvc.dll	[Program folder] source	DLL file to be used in the client. HackShield functions included.

3.2.1.2. Application

Server Application

1. Create HackShield.crc using AntiCPSvrTool.exe.
(See [10.3.AntiCpSvr Tool](#).)
2. Copy the HSPub.key file to the folder containing the HackShield.crc file.
3. Include the import library (AntiCpSvr.lib) file of AntiCpSvr.dll in the project.
4. Include the provided AntiCPSvrFunc.h file in the source file.
5. Call _AntiCpSvr_Initialize and perform initialization.
6. Upon client connection, call _AntiCpSvr_MakeGuidReqMsg and create an encrypted version request message to be sent to the client.
7. Send the encrypted version request message to the client.
8. Receive the encrypted version response message from the client.
9. Call _AntiCpSvr_AnalyzeGuidAckMsg and analyze the encrypted version of the response message sent by the client. Unless ERROR_SUCCESS is displayed, disconnect.
10. Call _AntiCpSvr_MakeReqMsg and create an encrypted request message to send to the client.
11. Send the encrypted request message to the client.
12. Receive the encrypted response message from the client.
13. Call _AntiCpSvr_AnalyzeAckMsg and analyze the encrypted response message that the client sent. Unless ERROR_SUCCESS is displayed, disconnect.
14. When terminating the server program, call the _AntiCpSvr_Finalize function.

Client Application

1. Include the HShield.lib file in the project.
2. Include the provided HShield.h file in the source file.
3. Receive an encrypted version request message from the server.
4. Call _AhnHS_MakeGuidAckMsg and create an encrypted version of the response message sent to the server.
5. Get an encrypted request message from the server.
6. Call _AhnHS_MakeAckMsg and create an encrypted response message to be sent to the server.
7. Send the encrypted response message to the server.

3.3. Application Programming Interface

AntiCpSvr Initialize

DESCRIPTION

Loads the data with the game executable file, memory, HackShield module, and engine file data and performs initialization. HackShield.crc and HSPub.key files must exist in the lpszHashFilePath path.

SYNTAX

```
unsigned long __stdcall  
_AntiCpSvr_Initialize (  
    IN const char *lpszHashFilePath  
);
```

PARAMETERS

Parameter	Value	Description
lpszHashFilePath	const char *	Full path of CRC data file (HackShield.crc)

RETURN VALUE

ERROR_SUCCESS . (Value = 0x00000000)

- Description: Returned when the function is successfully called.
- Cause:
- Solution:

ERROR_ANTICPSVR_INIT_INVALIDPARAM (Value = 0x1C001)

- Description: Incorrect input data.
- Cause: Occurs when lpszHashFilePath is null.
- Solution: Check whether lpszHashFilePath is normal.

ERROR_ANTICPSVR_INIT_INSERTCRCDATATOLIST_FAIL (Value = 0x1C002)

- Description: Cannot add the CRC file.
- Cause: Cannot read the CRC file or memory space is insufficient.

- Solution:

ERROR_ANTICPSVR_INIT_READPUBLICKEY_FAIL (Value = 1C004)

- Description: HSPub.key file is not found or damaged.
- Cause: HSPub.key does not exist in the corresponding path or is damaged.
- Solution: Check whether the HSPub.key file exists in the corresponding path.
(In the Windows server, use “\” instead of “/” to indicate the folder path.)

Example

The following is an example of calling _AntiCpSvr_Initialize function.

```
Example
dwRet = _AntiCpSvr_Initialize ( m_strFileCrcDataPath // [in]
);
if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

AntiCpSvr_Finalize

DESCRIPTION

Releases dynamically allocated memory and cleans up the internally used data.

SYNTAX

```
void __stdcall  
_AntiCpSvr_Finalize( )
```

PARAMETERS

None.

RETURN VALUE

None.

Example

The following is an example of calling _AntiCpSvr_FinalizeFunction.

```
Example
```

```
_AntiCpSvr_Finalize ();
```

AntiCpSvr_MakeGuidReqMsg

DESCRIPTION

Creates an encrypted version request message to be sent to the client.

SYNTAX

```
unsigned long __stdcall
_AntiCpSvr_MakeGuidReqMsg (
    OUT unsigned char *pbyGuidReqMsg,
    OUT unsigned char *pbyGuidReqInfo
);
```

PARAMETERS

Parameter	Type	Description
pbyGuidReqMsg	unsigned char *	Encrypted version request message to be sent to the client. The buffer size must be SIZEOF_GUIDREQMSG defined in AntiCpSvrFunc.h.
pbyGuidReqInfo	unsigned char *	Needed for the analysis by the _AntiCpSvr_AnalyzeGuidAckMsg function. The buffer size must be SIZEOF_GUIDREQINFO defined in AntiCpSvrFunc.h.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned when the function is successfully called.
- Cause: Occurs when lpzHashFilePath is null.
- Solution: Check whether lpzHashFilePath is normal.

ERROR_ANTICPSVR_MAKEGUIDREQMSG_INVALIDPARAM (Value = 0x1C040)

- Description: Incorrect input data.
- Cause: Occurs when pbyGuidReqMsg and pbyGuidReqInfo are null.
- Solution:

ERROR_ANTICPSVR_MAKEGUIDREQMSG_MAKESKEY_FAIL (Value = 0x1C041)

- Description: Message encryption failure.

- Cause:
- Solution:

**ERROR_ANTICPSVR_MAKEGUIDREQMSG_INITCRYPT_FAIL
(Value = 0x1C042)**

- Description: Failed in initializing encryption/decryption.
- Cause:
- Solution:

**ERROR_ANTICPSVR_MAKEGUIDREQMSG_ENCRYPT_FAIL
(Value = 0x1C043)**

- Description: Encryption failure.
- Cause:
- Solution:

Example

The following is an example of calling _AntiCpSvr_MakeGuidReqMsg function.

```
Example
dwRet = _AntiCpSvr_MakeGuidReqMsg (
    byGuidReqMsg, // [out]
    byGuidReqInfo // [out]
);

if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

AntiCpSvr_AnalyzeGuidAckMsg

DESCRIPTION

Decrypts the encrypted version response message from the client and checks that the version is same as the one currently used by the client.

SYNTAX

```
unsigned long __stdcall
_AntiCpSvr_AnalyzeGuidAckMsg (
    IN unsigned char *pbyGuidAckMsg,
    IN unsigned char *pbyGuidReqInfo,
    OUT PHSHIELD_CLIENT_CONTEXT pCrclInfo
);
```

PARAMETERS

Parameter	Type	Description
pbyAckMsg	unsigned char *	Encrypted version response message sent by the client
pbyGuidReqInfo	unsigned char *	Version request message created by _AntiCpSvr_MakeGuidReqMsg function
pCrclInfo	PHSHIELD_CLIENT_CONTEXT	Structure pointer for the CRC to be used by the client.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned when the function is successfully called.
- Cause:
- Solution:

ERROR_ANTICPSVR_ANALGUIDACKMSG_INVALIDPARAM

(Value = 0x1c050)

- Description: Incorrect input data.
- Cause: Occurs when pbyAckMsg, pbyGuidReqInfo, and pCrclInfo are null.
- Solution: Check whether pbyAckMsg, pbyGuidReqInfo, and pCrclInfo are normal. (Declare CrclInfo as a structure, and return the address.)

ERROR_ANTICPSVR_MAKEGUIDREQMSG_MAKESKEY_FAIL
(Value = 0x1C041)

- Description: Message encryption failure.
- Cause:
- Solution:

ERROR_ANTICPSVR_MAKEGUIDREQMSG_INITCRYPT_FAIL
(Value = 0x1C042)

- Description: Failed in initializing encryption/decryption.
- Cause:
- Solution:

ERROR_ANTICPSVR_ANALGUIDACKMSG_DECRYPT_FAIL
(Value = 0x1c053)

- Description: Message decryption failure.
- Cause: The message may not have been properly received.
- Solution: Check that the buffer and size sent by the client arrived properly in the input buffer of the above function (pbyAckMsg).

The client sends encrypted data, and only properly encrypted data can be decrypted. Improperly encrypted data will not be decrypted, and an error will be returned.

ERROR_ANTICPSVR_ANALGUIDACKMSG_PACKET_ERROR
(Value = 0x1c054)

- Description: A packet error occurred.
- Cause: The server called MakeGuidReqMsg and the client responded using MakeGuidAckMsg.
If synchronization is not achieved here, an error may occur.
- Solution: The server and client must be always synchronized. Check whether packet loss occurred in the server request due to synchronization error or network trouble.

ERROR_ANTICPSVR_ANALGUIDACKMSG_DENIED_NEWSESSION
(Value = 0x1c055)

- Description: Existing sessions are allowed, but new session of the corresponding version is not allowed.
- Cause: Error is returned if connection is attempted with a previous client version, not the version which created the CRC.
- Solution: Create the CRC and restart the server if the error occurred in the development phase. However, if the error occurred during service, it means the client may not have been properly

patched or may have been infected with a virus.

**ERROR_ANTICPSVR_ANALGUIDACKMSG_GETGUIDFROMCRCFILE_ERROR
(Value = 0x1c056)**

- Description: Failed to read the CRC file.
- Cause: HackShield.crc file does not exist in the server or is damaged.
- Solution:

**ERROR_ANTICPSVR_ANALGUIDACKMSG_INSERTCRCDATATOLIST_FAIL
(Value = 0x1c057)**

- Description: CRC data updating failed.
- Cause:
- Solution:

**ERROR_ANTICPSVR_ANALGUIDACKMSG_INVALIDGUID
(Value = 0x1c058)**

- Description: Unsupported version.
- Cause: The server does not allow the client version.
- Solution: Create the CRC and restart the server if the error occurred in the development phase. However, if the error occurred during the service, it means the client may not have been properly patched or may have been infected with a virus.

**ERROR_ANTICPSVR_ANALGUIDACKMSG_HSHIELDDENIED_NEWSESSION
(Value = 0x1c059)**

- Description: The server does not support this HackShield version.
- Cause: The error occurred because the server does not support the client HShield.dat file version. This error occurs when the version is lower than the version of the HShield.dat file in the server.
- Solution: HackShield was not properly updated in the client or a hacking attempt was made.

**ERROR_ANTICPSVR_ANALGUIDACKMSG_INSERTHSHIELDCRCDATATOLIST_FAIL
(Value = 0x1c05B)**

- Description: Failed to update the HackShield version.
- Cause: The server failed in updating the HackShield version.
- Solution:

ERROR_ANTICPSVR_ANALGUIDACKMSG_NANODENIED_NEWSESSION

(Value = 0x1c05D)

- Description: The server does not support the heuristic engine (3N.mhe) version.
- Cause: The server does not support 3N.mhe file version of the client. This error occurs when the version is lower than the 3N.mhe file version in the server.
- Solution: HackShield was not properly updated in the client or a hacking attempt was made.

**ERROR_ANTICPSVR_ANALGUIDACKMSG_INSERTNANOCRCDATATOLIST_FAIL
(Value = 0x1c05E)**

- Description: Failed in updating 3N.mhe file version.
- Cause: The server failed in updating 3N.mhe file version.
- Solution:

Caution

It is recommended that the session be disconnected from the client and the game be stopped when the above error code is returned.

Example

The following is an example of calling `_AntiCpSvr_AnalyzeGuidAckMsg` function.

```
Example
HSHIELD_CLIENT_CONTEXT CrcInfo = {0, };

dwRet = _AntiCpSvr_AnalyzeGuidAckMsg ( byGuidAckMsg, // [in]
                                       byGuidReqInfo, // [in]
                                       &CrcInfo // [out]
                                       );

if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

The structure (CrcInfo) for the CRC data returned by the OUT parameter is used until the client disconnects from the session. The structure (CrcInfo) must be stored for each client and is used as an input parameter for `_AntiCpSvr_MakeReqMsg` and `_AntiCpSvr_AnalyzeAckMsg` functions.

AntiCpSvr_MakeReqMsg

DESCRIPTION

Creates an encrypted CRC request message to be sent to the client.

SYNTAX

```
DWORD __stdcall
_AntiCpSvr_MakeReqMsg (
    IN PHSIELD_CLIENT_CONTEXT pCrcInfo,
    OUT unsigned char *pbyReqMsg,
    OUT unsigned char *pbyReqInfo,
    IN unsigned long ulOption
);
```

PARAMETERS

Parameter	Type	Description
pCrcInfo	PHSHIELD_CLIENT_CONTEXT	Returned by _AntiCpSvr_AnalyzeGuidAckMsg function. Structure pointer for the CRC to be used by the client.
pbyReqMsg	unsigned char *	Encrypted CRC request message to be sent to the client. The buffer size must be SIZEOF_REQMSG defined in AntiCpSvrFunc.h.
pbyReqInfo	unsigned char *	Needed for analysis by the _AntiCpSvr_AnalyzeAckMsg function. Original CRC request message. The buffer size must be SIZEOF_REQINFO defined in AntiCpSvrFunc.h.
ulOption	unsigned long	Needed for analysis by the _AntiCpSvr_AnalyzeAckMsg function. A flag will be created indicating the information to be included in a request message. Defined as ANTICPSVR_CHECK_GAME_MEMORY, .ANTICPSVR_CHECK_HACKSHIELD_FILE, ANTICPSVR_CHECK_GAME_FILE, ANTICPSVR_CHECK_NANOENGINE_FILE, and ANTICPSVR_CHECK_ALL in AntiCpSvrFunc.h. However, for safety purposes, it is recommended that general safety be inspected using the ANTICPSVR_CHECK_ALL option and only ANTICPSVR_CHECK_GAME_MEMORY should be used for performance.

RETURN VALUE

ERROR_SUCCESS

- Description: Incorrect input data.
- Cause:
- Solution:

ERROR_ANTICPSVR_MAKEREQMSG_INVALIDPARAM

(Value = 0x1C010)

- Description: Incorrect input data.
- Cause: Occurs when pbyAckMsg, pbyGuidReqInfo, and pCrcInfo are null.
- Solution: This error may occur if the value of the third parameter, uloption, is "0" or the input parameter value is "Null."

ERROR_ANTICPSVR_MAKEREQMSG_MAKESKEY_FAIL

(Value = 0x1C011)

- Description: Message encryption failure.
- Cause:
- Solution:

ERROR_ANTICPSVR_MAKEREQMSG_INITCRYPT_FAIL

(Value = 0x1C012)

- Description: Failed in initializing encryption/decryption.
- Cause:
- Solution:

ERROR_ANTICPSVR_MAKEREQMSG_ENCRYPT_FAIL

(Value = 0x1C013)

- Description: Failed in message decryption.
- Cause:
- Solution:

ERROR_ANTICPSVR_MAKEREQMSG_GETRNDHASHINFO_FAIL

(Value = 0x1C014)

- Description: Failed to get CRC data.
- Cause: This error may occur when pCrcInfo is "Null" or has an incorrect value.

- Solution:

Example

The following is an example of calling `_AntiCpSvr_MakeReqMsg` function.

```
Example

// byClientInfo and CrcInfo are gained during GUID request.
// Maintained while the client is connected.

dwRet = _AntiCpSvr_MakeReqMsg ( &CrcInfo,           // [in]
                                byReqMsg,           // [out]
                                byReqInfo,          // [out]
                                ANTICPSVR_CHECK_ALL // [in]
                                );

if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

`byReqInfo` returned by OUT is used to analyze the response to the request message. `byReqInfo` must be multi-thread safe, in other words, `byReqInfo` must be maintained until the `_AntiCpSvr_AnalyzeAckMsg` function is called to analyze the response to the request message.

AntiCpSvr_AnalyzeAckMsg

DESCRIPTION

Decrypts the encrypted CRC response message from the client, and checks the game file, packet integrity, whether HackShield is running properly, HackShield module integrity, and memory integrity.

SYNTAX

```
unsigned long __stdcall  
_AntiCpSvr_AnalyzeAckMsg (  
    IN PSHIELD_CLIENT_CONTEXT pCrcInfo,  
    IN unsigned char *pbyAckMsg,  
    IN unsigned char *pbyReqInfo  
);
```

PARAMETERS

Parameter	Type	Description
pCrcInfo	PSHIELD_CLIENT_CONTEXT	Structure pointer for the CRC to be used by the client. Returned by _AntiCpSvr_AnalyzeGuidAckMsg function.
pbyAckMsg	unsigned char *	Encrypted CRC response message sent by the client.
pbyReqInfo	unsigned char *	CRC request message returned by _AntiCpSvr_MakeReqMsg function.

RETURN VALUE

ERROR_SUCCESS (Value = 0x000000)

- Description: Successfully returned.
- Cause:
- Solution:

ERROR_ANTICPSVR_ANALACKMSG_INVALIDPARAM (Value = 0x1C020)

- Description: Incorrect input data.
- Cause: Occurs when pbyAckMsg, pbyReqInfo, and pCrcInfo are null.
- Solution:

ERROR_ANTICPSVR_ANALACKMSG_MAKESKEY_FAIL

(Value = 0x1C021)

- Description: Message encryption failure.
- Cause:
- Solution:

ERROR_ANTICPSVR_ANALACKMSG_INITCRYPT_FAIL

(Value = 0x1C022)

- Description: Failed in initializing encryption/decryption.
- Cause:
- Solution:

ERROR_ANTICPSVR_ANALACKMSG_DECRYPT_FAIL

(Value = 0x1c023)

- Description: Message decryption failure.
- Cause: The message may not have been properly received.
- Solution: Check that the buffer size sent by the client arrived properly in the input buffer of the above function (pbyAckMsg).

The client sends encrypted data, and only properly encrypted data can be decrypted. Improperly encrypted data cannot be decrypted, and an error will be returned.

ERROR_ANTICPSVR_ANALACKMSG_HSHIELD_ERROR

(Value = 0x1c024)

- Description: HackShield module has been manipulated.
- Cause: The Ehsvc.dll and HShield.dat files must be the same version. If either of the files had a module manipulation, the above error will be returned.
- Things to Check: The HackShield file of the client and whether an update was done properly. The HackShield module may have been manipulated by a hack attack.

ERROR_ANTICPSVR_ANALACKMSG_PACKET_ERROR
(Value = 0x1c025)

- Description: A packet error occurred.
- Cause: The server calls MakeReqMsg, and the client responds using MakeAckMsg.
- Solution: The server and client must be always synchronized. Check whether packet loss occurred in the server request due to synchronization error or network trouble.

ERROR_ANTICPSVR_ANALACKMSG_FILECRC_ERROR ERROR
(Value = 0x1c026)

- Description: Client module manipulation was detected.
- Cause: This error will be returned if the client file was manipulated.
- Solution: Check if a file was manipulated or whether the client was properly patched.

ERROR_ANTICPSVR_ANALACKMSG_MEMORYCRC_ERROR
(Value = 0x1c027)

- Description: Memory manipulation has been detected.
- Cause: This error is returned when memory manipulation occurred.
- Solution:

ERROR_ANTICPSVR_ANALACKMSG_INVALIDSESSION_ERROR
(Value = 0x1c028)

- Description: This error occurs when an option was given to disconnect the current sessions upon file patch installation.
- Cause:
- Solution: This error occurs when an option was given to disconnect the current sessions upon file patch installation.

ERROR_ANTICPSVR_ANALACKMSG_NANOENGINECRC_ERROR
(Value = 0x1c02B)

- Description: 3N.mhe file manipulation was detected.
- Cause: This error will be returned if the 3N.mhe file is manipulated.
- Solution: Check if a file was manipulated or whether the client was properly patched.

Caution

It is recommended that the session be disconnected from the client and the game be stopped if the above error code is returned.

Example

The following is an example of calling the `_AntiCpSvr_AnalyzeAckMsg` function.

```
Example
// byClientInfo an CrcInfo are acquired during MakeReqMsg
request.
// Maintained while the client is connected.

dwRet = _AntiCpSvr_AnalyzeAckMsg ( &CrcInfo,    // [in]
                                   byAckMsg,      // [in]
                                   byReqInfo      // [in]
                                   );

if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

AhnHS_MakeGuidAckMsg

DESCRIPTION

Decrypts the encrypted version request message used and sent by the client, encrypts the current client file, and creates response messages.

SYNTAX

```
int __stdcall
_AhnHS_MakeGuidAckMsg (
    IN unsigned char *pbyGuidReqMsg,
    OUT unsigned char *pbyGuidAckMsg
);
```

PARAMETERS

Parameter	Type	Description
pbyGuidReqMsg	unsigned char *	Encrypted Version request message sent by the server. The buffer size must be SIZEOF_GUIDREQMSG defined in HShield.h.
pbyGuidAckMsg	unsigned char *	Encrypted version response message to be sent to the server. The buffer size must be SIZEOF_GUIDACKMSG defined in HShield.h.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Successfully returned.
- Cause:
- Solution:

ERROR_ANTICPNT_MAKEGUIDACKMSG_INVALIDPARAM
(Value = 0x10010)

- Description: Incorrect input data.
- Cause: Occurs when pbyGuidReqMsg and pbyGuidAckMsg are null.
- Solution:

ERROR_ANTICPNT_MAKEGUIDACKMSG_INITCRYPT_FAIL
(Value = 0x1C012)

- Description: Failed in initializing encryption/decryption.
- Cause:

- Solution:

ERROR_ANTICPCNT_MAKEGUIDACKMSG_DECRYPTMESSAGE_FAIL (Value = 0x10013)

- Description: Message decryption failure.
- Cause: The message may not have been properly received.
- Solution: Check whether the buffer size sent by the server properly arrived to the input buffer of the above function (pbyGuidReqMsg).

The server sends the encrypted buffer. Only a properly encrypted buffer can be decrypted. An improperly encrypted buffer will not be decrypted and will result in errors.

ERROR_ANTICPCNT_MAKEGUIDACKMSG_GETIDENTIFIER_FAIL (Value = 0x10014)

- Description: No GUID found.
- Cause: An error occurred in the HackShield module or the game client file while a GUID was being created.
- Solution: Check whether the HackShield module or game client file is normal.

ERROR_ANTICPCNT_MAKEGUIDACKMSG_ENCRYPTMESSAGE_FAIL (Value = 0x10016)

- Description: Message encryption failure.
- Cause:
- Solution:

Example

The following is an example of using `_AhnHS_MakeGuidAckMsg`.

```
Example

dwRet = _AhnHS_MakeGuidAckMsg( byGuidReqMsg,    // [in]
                               byGuidAckMsg      // [out]
                               );

if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

DESCRIPTION

Decrypts the encrypted CRC request message used by the client and sent by the server, gets game file information, and creates encrypted CRC response messages.

SYNTAX

```
int __stdcall
_AhnHS_MakeAckMsg (
    IN unsigned char *pbyReqMsg,
    OUT unsigned char *pbyAckMsg
);
```

PARAMETERS

Parameter	Type	Description
pbyReqMsg	unsigned char *	Encrypted CRC request message sent by the server. The buffer size must be <code>SIZEOF_REQMSG</code> defined in <code>HShield.h</code> .
pbyAckMsg	unsigned char *	Encrypted CRC response message to send to the server. The buffer size must be <code>SIZEOF_ACKMSG</code> defined in <code>HShield.h</code> .

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Successfully returned.
- Cause:
- Solution:

ERROR_ANTICPNT_MAKEACKMSG_INVALIDPARAM

(Value = 0x10000)

- Description: Incorrect input data.
- Cause: Occurs when `pbyReqMsg` and `pbyAckMsg` are null.
- Solution:

**ERROR_ANTICPNT_MAKEACKMSG_INITCRYPT_
(Value = 0x10012)**

- Description: Failed in initializing encryption/decryption.
- Cause:
- Solution:

Example

The following is an example of using `_AhnHS_MakeAckMsg`.

```
Example

dwRet = _AhnHS_MakeAckMsg( byReqMsg,      // [in]
                           byAckMsg      // [out]
                           );

if( dwRet != ERROR_SUCCESS )
{
    // error
}
```

AhnHS_SaveFuncAddress

DESCRIPTION

Receives the function pointer of the protected game client as an argument, and stores the CRC of the corresponding functions as a HackShield.crc file. To guarantee memory integrity, the server must manage the HackShield.crc file created by this function. If the function pointer is returned as an argument, a maximum of 32 functions can be protected. The number of the functions returned as the first argument must match the number of the following function pointers.

Caution

This function must exist in EXE and other functions of the same name (overloading) are not supported.

SYNTAX

```
int __stdcall
_AhnHS_SaveFuncAddress (
    IN unsigned int unNumberOfFunc,
    ...
);
```

PARAMETERS

Parameter	Type	Description
unNumberOfFunc	unsigned int	Number of functions to protect
...	...	Variable argument. Input the function pointer.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Successfully returned.
- Cause:
- Solution:

ERROR_ANTICPNT_SAVEFUNCADDRESS_INVALIDPARAM

(Value = 0x10020)

- Description: Incorrect input data.
- Cause: Incorrect input format.
- Solution:

**ERROR_ANTICPCNT_SAVEFUNCADDRESS_INITCRYPT_FAIL
(Value = 0x10023)**

- Description: Failed in initializing encryption/decryption.
- Cause:
- Solution:

**ERROR_ANTICPCNT_SAVEFUNCADDRESS_DECRYPTMESSAGE_FAIL
(Value = 0x10024)**

- Description: Message decryption failure.
- Cause:
- Solution:

**ERROR_ANTICPCNT_SAVEFUNCADDRESS_ENCRYPTMESSAGE_FAIL FAIL
(Value = 0x10029)**

- Description: Message encryption failure.
- Cause:
- Solution:

4. Extended Server Side Detection

(For HackShield Pro, 4.2 and higher versions)

4.1. Overview

Extended Server Side Detection Interface (AntiCpX) is an advanced version of the server interface software development kit that provided file/memory manipulation detection functions. The most significant improvement is that the extended server interface protects the entire code area while the old server interfaces protected only the code loaded on the memory in each function. The extended server interface is supported in HackShield Pro 4.2 and higher versions.

4.1.1. Functions

Extended Server Side Detection (AntiCpX) includes functions of the old server interfaces and provides enhanced “memory integrity function.”

File Integrity Verification

Apply AntiCpXSvr.dll and AntiCpXSvr.h in the server, and the HShield.lib and HShield.h, HackShield modules, in the client. The game developer can check in real time whether the game executable file is being manipulated through communication between the server and client.

Packet Integrity

To prevent hack attacks that capture and generate packets in certain circumstances, a separate module runs to guarantee packet integrity. The game developer can block packet capturing and creation on the network. However, this function only protects messages related to preventing server interface cracks.

Operational State of HackShield

Apply AntiCPXSvr.dll and AntiCpXSvr.h in the server, and the HShield.lib and HShield.h, HackShield modules, in the client. The game developer can easily check how HackShield is running through communication between the server and client. The server must send a query message only after HackShield starts.

Memory Integrity (Extended Server Interface)

Apply AntiCpXSvr.dll and AntiCpXSvr.h in the server, and the HShield.lib and HShield.h, HackShield modules, in the client. The game developer can check in real time whether the game process memory is being manipulated through communication between the server and client. The extended server interface protects the entire code area while the former server interfaces protected only the code loaded on the memory for each function.

HackShield Engine Integrity Verification

Apply AntiCPXSvr.dll and AntiCpXSvr.h, in the server, and the HShield.lib and HShield.h, HackShield modules, in the client. The game developer can check the integrity of the heuristic engine (3N.mhe) file through communication between the server and client.

4.1.2. Features

Interface Function (API)

Interface DLLs and a library are provided so the game developer can use the extended server interface (AntiCpX) functions and check the return values. The game developer can check the integrity of the game file and how HackShield is running by using the interface DLLs and the library according to the developer's policy.

File Information Creation Program

HSBGen.exe provided by the extended server interface (AntiCpX) creates and stores file data and memory information for the client program to check file/memory integrity through communication between the client and server. The game developer can store file/memory information in the server and use this data to prevent file/memory cracks through the server interfaces. For more information, see '11.5 HSBGen Tool (For HackShield Pro 4.2 or higher versions).

4.1.3. System Architecture

Extended Server Interface (AntiCpX) provides AntiCPXSvr.dll, HShield.lib, and EHSvc.dll is applied as DLL and library in the server and client. General architecture and operating principles of the extended server interface (AntiCpX) are as follows:

AntiCPXSvr.dll (Interface dll)

Used by the server. Provides an API that creates request messages and checks client files and memory status based on the response message from the client.

HShield.lib (HackShield Library)

Used by the client. Provides an API that receives request messages from the server, encrypts the response data, and sends the encrypted response to the server.

HSBGen.exe (File Information Creation Program)

Creates file data and memory information for the server to check the integrity of the game file and game memory. Different file data are created for each file creation, so if there are multiple servers, each server must have different file data for greater security. Memory data is automatically created when the game program is executed while AntiCpSvrTool.exe is running.

AntiCpx.hsb (Client CRC File)

Created by HSBGen.exe. Designed to guarantee integrity of the client program.

HSPub.key (Server Interface Authentication Key File)

Used to authenticate the accessing client's HackShield. Must be stored in the folder that contains the AntiCpx.hsb file.

3N.mhe (Heuristic engine File)

Engine file used by the client. If stored in the folder that contains the AntiCpx.hsb file of the server, clients using previous versions of 3N.mhe engine will be disconnected.

HShield.dat (HackShield Version File)

HackShield version management file used by the client. If stored in the folder that contains the AntiCpx.hsb file of the server, clients using previous versions of HackShield will be disconnected.

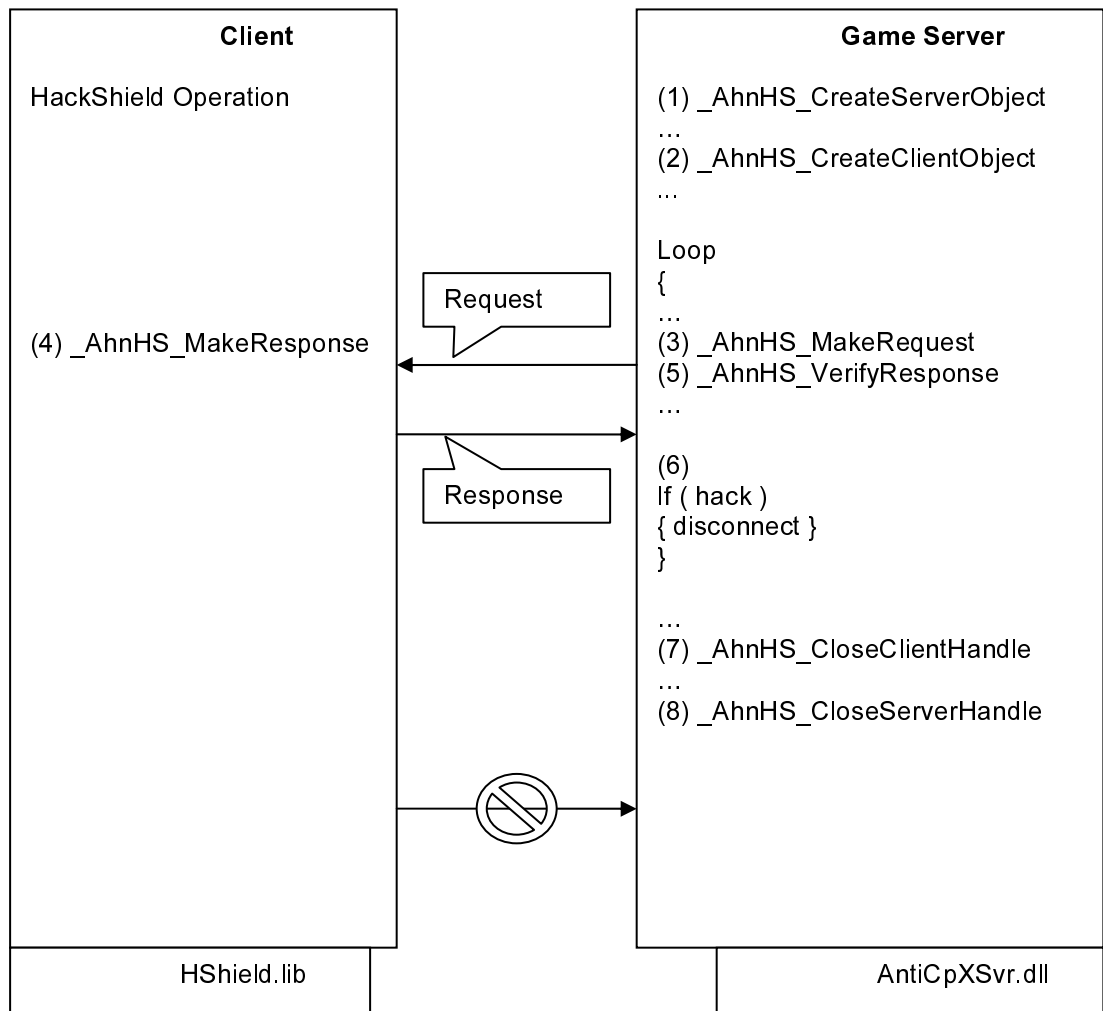


Figure 4-1 Operating Principles of AntiCpX

When the game server is first executed, AHNHS_SERVER_HANDLE will be created by the _AhnHS_CreateServerObject function. Then, whenever a client is accessed, the _AhnHS_CreateClientObject function and server handle are collected as parameters and the AHNHS_CLIENT_HANDLE is created. To check whether the client is being manipulated, the _AhnHS_MakeRequest function regularly creates and sends request messages.

The client creates a response message to the server's request. The response is created by the _AhnHS_MakeResponse function. The validity of the client's response is verified by the _AhnHS_VerifyResponse function.

4.2. Application Programming

This chapter describes how to check file and memory integrity using the APIs provided by the extended server interface (AntiCpX).

Note

The sample codes contained in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

4.2.1. Programming Application

Do the following using the Extended Server Side Detection (AntiCpX) before starting programming:

4.2.1.1. AntiCpXSvr-related File

AntiCpXSvr-related File

Table 4-1 AntiCpXSvr-related File

File Name	Installation Folder	Description
AntiCpXSvr.h	[Program folder] source	Header file to be used in the server
AntiCpXSvr.dll	[Program folder] source	DLL file to be used in the server
AntiCpXSvr.lib	[Program folder] source	Import library of AntiCpXSvr.dll
AntiCpx.hsb	[Program folder] execution	Client integrity verification file
HSPub.key	[Program folder] execution	Server Interface Authentication Key File
HShield.h	[Program folder] source	Header file to be used in the client. HackShield functions included.
HShield.lib	[Program folder] source	Library file to be used in the client. HackShield functions included.
EHSvc.dll	[Program folder] source	DLL file to be used in the client. HackShield functions included.

4.2.1.2.Application

Server Application

1. Create AntiCpx.hsb using HSBGen.exe.
(See [10.5.HSBGen Tool](#).)
2. Copy the HSPub.key file to the folder containing the AntiCpx.hsb file.
3. When the game server is first executed, create the AHNHS_SERVER_HANDLE using the _AhnHS_CreateServerObject function. This handle must be maintained until the game server is terminated. (Note: The server handle is used to make the client handle in the following 2 steps.)
4. When a client accesses, the _AhnHS_CreateClientObject function and the server handle will be collected as parameters and the AHNHS_CLIENT_HANDLE will be created. This handle must be maintained during the session in which the client is connected to the network. (Note: The client handle is used to make a request message in the following three steps.)
5. To monitor if the client is being manipulated, a request message must be created and sent. Create a request message using the _AhnHS_MakeRequest function, and use the client handle as the parameter.
6. The client will create a proper response to the server's request. A response is created by the _AhnHS_MakeResponse function.
7. Check whether the response from the client is valid. The validity of the response message is checked by _AhnHS_VerifyResponse function.
8. If error codes other than ERROR_SUCCESS are returned by the _AhnHS_VerifyResponse function, proper measures must be taken and the game client must be disconnected.
9. When the client is disconnected (or session is terminated), the client handle created in Step 2 must be closed.
10. When the server process is terminated, the server handle created in Step 1 must be closed (with the client handle being closed.)

Client Application

1. Include the HShield.lib file in the project.
2. Include the provided HShield.h file in the source file.
3. Receive an encrypted version request message from the server.
4. Call _AhnHS_MakeResponse and create an encrypted version response to be sent to the server.
5. Send the encrypted response to the server.

4.3. Application Programming Interface

AhnHS_CreateServerObject

DESCRIPTION

Creates a server handle by loading the .hsb file created by HSBGen.exe. Usually, one server handle is created for the server process that services one game, and the server handle is maintained until the game server process is terminated.

SYNTAX

```
AHNHS_SERVER_HANDLE __stdcall  
_AhnHS_CreateServerObject (  
    IN const char *pszFilePath  
);
```

PARAMETERS

Parameter	Value	Description
pszFilePath	const char *	Full path of HackShield Briefcase (.hsb)

RETURN VALUE

If a server handle is not properly created, null data will be returned. This may occur in the following cases:

- ① When the path of the HackShield Briefcase (.hsb) file is not correct or the file does not exist.
- ② The HSPub.key file does not exist in the path
- ③ System resources (memory) are insufficient.

Example

The following is an example of calling the _AhnHS_CreateServerObject function

```
Example  
  
// The file path may contain '\\' , not '/'.  
// The path must include the file name.  
  
strcpy(g_szHsbFilePath,  
    "C:\\\\GameServer\\\\HShield\\\\anticpx.hsb" );  
  
hServer = _AhnHS_CreateServerObject (g_szHsbFilePath);  
  
if ( hServer == ANTICPX_INVALID_HANDLE_VALUE )
```

```
{  
    // error  
}
```

AhnHS_CloseServerHandle

DESCRIPTION

Closes the server handle.

SYNTAX

```
void __stdcall  
_AhnHS_CloseServerHandle (  
    IN AHNHS_SERVER_HANDLE hServer  
);
```

PARAMETERS

Parameter	Value	Description
pszFilePath	AHNHS_SERVER_HANDLE	Handle created by _AhnHS_CreateServerObject function.

RETURN VALUE

None.

Example

The following is an example of calling _AhnHS_CloseServerHandle function.

```
Example
```

```
_AhnHS_CloseServerHandle ( hServer );
```

AhnHS_CreateClientObject

DESCRIPTION

Receives server handle input, and creates a client handle. A client handle will be created each time the client connects. The client handle is maintained and recycled while the session is maintained.

SYNTAX

```
AHNHS_CLIENT_HANDLE  
_AhnHS_CreateClientObject (  
    IN AHNHS_SERVER_HANDLE hServer  
);
```

PARAMETERS

Parameter	Type	Description
hServer	AHNHS_SERVER_HANDLE	Server handle

RETURN VALUE

Client handle(Client Handle)

Example

The following is an example of calling the _AhnHS_CreateClientObject function.

```
Example  
  
hClient = _AhnHS_CreateClientObject ( hServer );  
  
if ( hClient == ANTICPX_INVALID_HANDLE_VALUE )  
{  
    // error  
}
```

AhnHS_CloseClientHandle

DESCRIPTION

The created client handle must be closed when the client session is terminated. At this time, the memory and system resources allocated to the client handle must also be released.

SYNTAX

```
void __stdcall  
_AhnHS_CloseClientHandle (  
    IN AHNHS_CLIENT_HANDLE hClient  
);
```

PARAMETERS

Parameter	Type	Description
hClient	AHNHS_CLIENT_HANDLE	Client handle(Client Handle)

RETURN VALUE

None.

Example

The following is an example of calling the _AhnHS_CloseClientHandle function.

Example

```
_AhnHS_CloseClientHandle ( hClient );
```

AhnHS_MakeRequest

DESCRIPTION

Creates a request message by inputting a client handle for the current session. The request message is displayed as the AHNHS_TRANS_BUFFER structure, and the member parameters are as follows:

```
typedef struct _AHNHS_TRANS_BUFFER
{
    unsigned short nLength;
    unsigned char byBuffer[ANTICPX_TRANS_BUFFER_MAX]; // Maximum
    buffer size for the packets

} AHNHS_TRANS_BUFFER, *PAHNHS_TRANS_BUFFER;
```

nLength ; Buffer length used in the creation of the request message
byBuffer ; Maximum byte buffer that may be used for the creation of the request message

Caution

byBuffer indicates the maximum buffer sizes available for the creation of the request message. Data must be sent on the network by nLength.

SYNTAX

```
unsigned long __stdcall
_AhnHS_MakeRequest (
    IN AHNHS_CLIENT_HANDLE hClient,
    OUT PAHNHS_TRANS_BUFFER pRequestBuffer
);
```

PARAMETERS

Parameter	Type	Description
hClient	AHNHS_CLIENT_HANDLE	Client handle
pRequestBuffer	PAHNHS_TRANS_BUFFER	Data buffer/length to send

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

ERROR_ANTICPXSVR_INVALID_PARAMETER (Value = 0xE9040003)

- Description: Incorrect input data.
- Cause:
- Solution:

ERROR_ANTICPXSVR_BAD_FORMAT (Value = 0xE9040004)

- Description: HSB file reading failed.
- Cause:
- Solution:

ERROR_ANTICPXSVR_NOT_YET_RECEIVED_RESPONSE (Value = 0xE9040005)

- Description: No response to the previous request message has arrived.
- Cause: Once the `_AhnHS_MakeRequest` function is called, the Ack message must arrive from the client and the `_AhnHS_VerifyResponse` function must be called again. The error occurs if this synchronization process was not completed.
- Solution: Check for any synchronization issues in the client session management.

ERROR_ANTICPXSVR_NOT_ENOUGH_MEMORY (Value = 0xE9040007)

- Description: Insufficient memory space.
- Cause:
- Solution:

ERROR_ANTICPXSVR_BAD_MESSAGE (Value = 0xE9040008)

- Description: Buffer encryption failed.
- Cause:
- Solution:

Example

The following is an example of calling the `_AhnHS_MakeRequest` function.

```
Example
AHNHS_TRANS_BUFFER stReqTransBuf;

ulRet = _AhnHS_MakeRequest ( hClient, &stReqTransBuf );
```

```
if ( ulRet != ERROR_SUCCESS )  
    return ulRet;  
  
bytesSent = send( ConnectSocket, stReqTransBuf.byBuffer,  
stReqTransBuf.nLength, 0 );  
  
. . .
```

AhnHS_VerifyResponse

DESCRIPTION

Checks the client response to the request message sent by the _AhnHS_MakeRequest function.

SYNTAX

```
unsigned long __stdcall
_AhnHS_VerifyResponse (
    IN AHNHS_CLIENT_HANDLE hClient,
    IN unsigned char *pbyResponse,
    IN unsigned long nResponseLength
);
```

PARAMETERS

Parameter	Type	Description
hClient	AHNHS_CLIENT_HANDLE	Client handle
pbyResponse	char *	Data buffer received from the client
nResponseLength	unsigned long	Data length received from the client

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

ERROR_ANTICPXSVR_INVALID_PARAMETER (Value = 0xE9040003)

- Description: Incorrect input data.
- Cause:
- Solution:

ERROR_ANTICPXSVR_BAD_FORMAT (Value = 0xE9040004)

- Description: Incorrect format.
- Cause:

- Solution:

ERROR_ANTICPXSVR_NOT_YET_RECEIVED_RESPONSE
(Value = 0xE9040005)

- Description: No response to the request message has arrived.
- Cause: Once the _AhnHS_MakeRequest function is called, the Ack message must arrive from the client and the _AhnHS_VerifyResponse function must be called again. The error occurs if this synchronization process was not completed.
- Solution: Check for any synchronization issues in the client session management.
-

ERROR_ANTICPXSVR_NO_WAITING

- Description: Message response is not waited for.
- Cause: Once the _AhnHS_MakeRequest function is called, the Ack message must arrive from the client and the _AhnHS_VerifyResponse function must be called again. The error occurs if this synchronization process was not completed.
- Solution: Check for any synchronization issues in the client session management.

ERROR_ANTICPXSVR_NOT_ENOUGH_MEMORY
(Value = 0xE9040007)

- Description: Insufficient memory space.
- Cause:
- Solution:

ERROR_ANTICPXSVR_BAD_MESSAGE (Value = 0xE9040008)

- Description: Message encryption/decryption failure.
- Cause: Incorrect buffer value from the client.
- Solution: Check if the buffer value from the client arrived through pbyResponse.

ERROR_ANTICPXSVR_REPLY_ATTACK (Value = 0xE9040009)

- Description: Retransmission attack for packet analysis was detected.
- Cause:
- Solution:

ERROR_ANTICPXSVR_HSHIELD_FILE_ATTACK

- Description: HackShield Module manipulation has been detected.
- Cause: Occurs when lpszHashFilePath is null.
- Solution: Check whether lpszHashFilePath is normal.
HackShield Module manipulation has been detected. (Value = 0xE904000A)

ERROR_ANTICPXSVR_CLIENT_FILE_ATTACK

- Description: Incorrect input data.
- Cause: Occurs when lpszHashFilePath is null.
- Solution: Check whether lpszHashFilePath is normal.
Client File manipulation has been detected. (Value = 0xE904000B)

ERROR_ANTICPXSVR_MEMORY_ATTACK

- Description: Incorrect input data.
- Cause: Occurs when lpszHashFilePath is null.
- Solution: Check whether lpszHashFilePath is normal.
Memory manipulation has been detected. (Value = 0xE904000C)

ERROR_ANTICPXSVR_OLD_VERSION_CLIENT_EXPIRED

- Description: Incorrect input data.
- Cause: Occurs when lpszHashFilePath is null.
- Solution: Check whether lpszHashFilePath is normal.
The old-version client is no longer supported.
(If HSB file was updated without server interruption and the HSB policy does not support the old version client, this error may occur. This error is due to the policy and is not an error) (Value = 0xE904000D)

ERROR_ANTICPXSVR_UNKNOWN_CLIENT (Value = 0xE904000E)

- Description: Does not pair with the client specified during creation of the HSB file.
- Cause: The version is not same as the client version used to create hsb by HSBGen.
- Solution: Create the hsb file again and conduct the test again if the error occurred in the development phase. If the error occurred to a certain user in the service stage, check whether the client file was properly patched or if it was infected with viruses.

ERROR_ANTICPXSVR_NANOENGINE_FILE_ATTACK

(Value = 0xE9040010)

- Description: 3N.mhe file manipulation has been detected.
- Cause:
- Solution:

ERROR_ANTICPXSVR_INVALID_HACKSHIELD_VERSION

(Value = 0xE9040011)

- Description: The server does not support this HackShield version.
- Cause:
- Solution:

ERROR_ANTICPXSVR_INVALID_ENGINE_VERSION

(Value = 0xE9040012)

- Description: Not supported heuristic engine version.
- Cause:
- Solution:

ERROR_ANTICPXSVR_UNKNOWN

(Value = 0xE90400FF)

- Description: Undefined error.
- Cause:
- Solution:

Caution

ERROR_ANTICPXSVR_BAD_MESSAGE, ERROR_ANTICPXSVR_REPLY_ATTACK,
ERROR_ANTICPXSVR_HSHIELD_FILE_ATTACK,
ERROR_ANTICPXSVR_CLIENT_FILE_ATTACK,
ERROR_ANTICPXSVR_MEMORY_ATTACK,
ERROR_ANTICPXSVR_NANOENGINE_FILE_ATTACK

It is recommended that the session should be disconnected from the client and the game should be stopped if the above error code is returned. Other message may be added depending on the game developer's policies.

Example

The following is an example of calling `_AhnHS_VerifyResponse` function.

```
Example

ulRet = _AhnHS_VerifyResponse ( hClient,
                                stResponseBuf.byBuffer,
                                stResponseBuf.nLength );

if ( ulRet != ERROR_SUCCESS )
{
    // error
}
```

AhnHS_MakeResponse

DESCRIPTION

Used by the client. Decrypts the encrypted version request message from the server, encrypts the current client file version, and creates a response message.

SYNTAX

```
int __stdcall
_AhnHS_MakeResponse (
    unsigned char *pbyRequest,
    unsigned long ulRequestLength,
    PAHNHS_TRANS_BUFFER pResponseBuffer
);
```

PARAMETERS

Parameter	Type	Description
pbyRequest	unsigned char *	[IN] Request Message buffer
ulRequestLength	unsigned long	[IN] Request Message length
pResponseBuffer	PAHNHS_TRANS_BUFFER	[OUT] Response Message buffer

RETURN VALUE

ERROR_SUCCESS . (Value = 0x00000000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

AHNLAB_DEFINED_ERROR_CODE

(Value = 0x00010010 ~ 0x0001D016)

- Description: An error code defined by AhnLab is returned.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_INVALID_PARAMETER . (Value = 0xE4010001)

- Description: Incorrect parameters.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_INVALID_ADDRESS (Value = 0xE4010002)

- Description: Wrong memory address accessed.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_NOT_ENOUGH_MEMORY (Value = 0xE40100013)

- Description: Insufficient memory space.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_CRC_TABLE_INIT_FAILED (Value = 0xE4010004)

- Description: Failed in initialization.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_BAD_LENGTH (Value = 0xE4010005)

- Description: Incorrect message length.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_INSUFFICIENT_BUFFER (Value = 0xE4010006)

- Description: Buffer is too small.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_NOT_SUPPORTED (Value = 0xE4010007)

- Description: Not supported in the current version.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_FILE_NOT_FOUND (Value = 0xE4010008)

- Description: Cannot find the file.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_INVALID_MESSAGE_SIZE (Value = 0xE4010009)

- Description: Incorrect message size.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_BAD_FORMAT (Value = 0xE401000A)

- Description: Incorrect format.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_DEBUGGER_DETECTED (Value = 0xE401000B)

- Description: Debugging detected.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_BAD_HSHIELD_MODULE (Value = 0xE401000C)

- Description: Incorrect HackShield module path or wrong HackShield module.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_BAD_CLIENT_FILE (Value = 0xE401000D)

- Description: Wrong client module.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_BAD_REQUEST (Value = 0xE401000E)

- Description: Incorrect request message from the server.
- Cause: Buffer was improperly received from the server and decryption failed.
- Solution: Check that the buffer value and length sent from the server returned by pbyRequest are accurate.

HS_ERR_ANTICPXCNT_HSHIELD_CORE_ENGINE_NOT_WORKING

(Value = 0xE401000F)

- Description: HackShield core engine is not properly running.
- Cause:
- Solution:

HS_ERR_ANTICPXCNT_UNKNOWN . (Value = 0xE40100FF)

- Description: Returned when the system does not operate properly due to hack attacks.
- Cause:
- Solution:

Example

The following is an example of _AhnHS_MakeResponse in use.

```
Example
ulRet = _AhnHS_MakeResponse ( stRequestBuf.byBuffer,
                              stRequestBuf.nLength,
                              &stResponseBuf );

if ( ulRet != ERROR_SUCCESS )
{
    // error
}
```

5. Real-time Decryption of Executable file (For HackShield Pro)

5.1. Overview

The real-time decryption function selects and reads important file data that are related to execution, and decrypts only these files in real time. Therefore, it is impossible to view or change the other contents of the files that contain the decrypted data. Reverse engineering technologies will protect executable files from potential hack attacks and provide different protection methods from those of the existing packers, making it difficult for hackers to analyze files with existing attack patterns or methods.

5.1.1. Functions

Hsp Launcher (HspL)

Executes the encrypted game client, and runs HackShield to protect HspL and the game client. Includes Hsp driver and provides management functions (installation, start, stop, and deletion.)

Driver

Executes the encrypted game client and allows only the authorized processes to perform decryption. Checks whether the game client is protected by HackShield.

HackShield Cryptor

Encrypts the executable file to be distributed. Provides project management functions to allow systematic management of the encrypted files and efficient distribution. For more information, see '**11.2. Using HackShield Cryptor Tool**'.

5.1.2. Features

Interface Function (API)

Provides interface library for the game developer to use HspLauncherLib functions and get the return values. The game developer can easily decrypt the executable files using the interface library according to the developer's policy.

Executable file Encryption Program

Encrypts files using HspCryptor.exe, a file encryption tool. Can decrypt the encrypted file in real time using the API provided by HspLauncherLib.

5.1.3. System Architecture

The real-time decryption function is provided as an SDK library and an executable file that decrypts the encrypted file in real time. The general architecture and operating principles of the real-time decryption function are as follows:

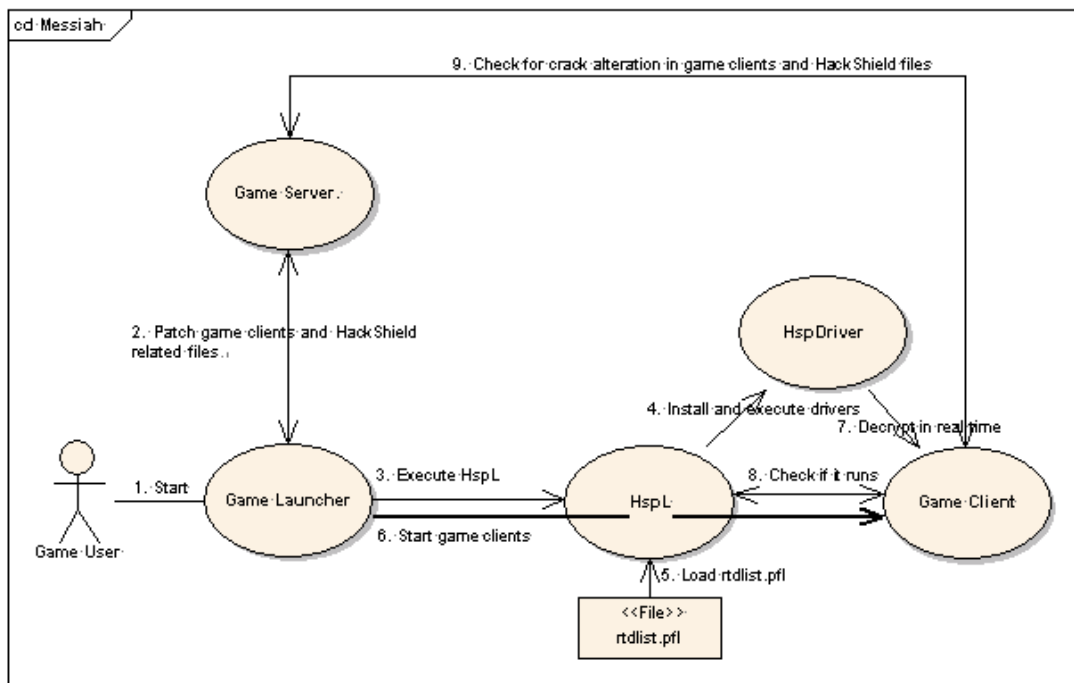


Figure 5-1 General Architecture and Operating Principles of Real-time Decryption

1. The game user starts the game launcher.
2. The game launcher requests the game server to send the latest versions of the game client file, HspL, and HackShield file.
3. After all files are downloaded, HspL executes HackShield to protect the game.
4. If HackShield is properly running, Hsp driver will be executed.
5. Before the game client is executed, data needed for real-time decryption will be loaded from the rtdlist.pfl file. The rtdlist.pfl file is created when HspCryptor encrypts important files, and must be stored in the same folder as HspL.

6. Start the game client through the HspL.
7. The Hsp driver will decrypt the game client in real time.
8. HspL regularly checks whether the child processes (game clients) that it generated exist.
9. The game server regularly checks whether files were manipulated in the game client and the HackShield-related modules.

Caution

If all child processes (game clients) created by HspL are terminated, or if a hack attack arises, HspL is automatically terminated. This is to allow the game launcher or the game client to operate independently of HspL shutting down.

Note that **HspL only checks whether the child processes were terminated and cannot detect whether the processes executed by the child processes were terminated.** Therefore, when a child process is created and the child process executes another process and terminates it, HspL may also be unintentionally terminated. HspL functions without a user interface.

5.2. Application Programming

5.2.1. Programming Procedure

The game developer can implement real-time decryption of executable files as follows.

1. Preparation: Check the list of the provided HspLauncherLib files, and copy necessary files. Apply HspLauncherLib to game launcher in most cases.
2. Calling HspL Start Function: Write a function to start HspL, and call the function before decrypting the executable file in real time. If HspL is not running, the executable file cannot be decrypted. HspL will be terminated if hacking occurs or if the child processes executed by HspL are terminated. HspL can be terminated only when the game file execution function (_AhnHsp_StartGame) is called.
3. Calling Game File Execution Function: If the executable file is encrypted and HspL is running, write a code that calls the function to decrypt the file in real time. In this way, the game developer can decrypt important files.
4. Test whether the written source code is running properly.
5. Distribute to users.

5.2.2. Preparation

HspLauncherLib is generally applied to programs that function as game launchers. Do the following before starting programming:

5.2.3. HspLauncherLib-related Files

HsCryptLib-related Files

Table 5-1 HsCryptLib-related Files

File Name	Installation Folder	Description
HspLauncherLib.h	[Program source folder]	Header File
HspLauncherLib.lib	[Program source folder]	Library File : Multi-thread and single thread library provided.

Compiler Setting

The project file of the game launcher program that uses HspLauncherLib must include the HspLauncherLib.lib file in the library or source code list. The game developer must check whether the project that will use HspLauncherLib has a multi-thread library or a single-thread library, and apply the appropriate HspLauncherLib.lib.

5.2.4. _AhnHsp_StartLauncher

When preparations for programming have been completed, call _AhnHsp_StartLauncher. Only after the _AhnHsp_StartLauncher function is successfully called can the _AhnHsp_StartGame function decrypt the executable file in real time.

The following is an example of calling the _AhnHsp_StartLauncher function.

```
Example

DWORD dwRet = ERROR_SUCCESS;
CHAR szFilePath[MAX_PATH] = { 0, };
...
dwRet = AhnHsp_StartLauncher( szFilePath ); // HspL full path.
if( dwRet != ERROR_SUCCESS )
{
    //error handling.
}
```

In the example, szFilePath is the full path of the HspL to be executed, and HspL must be in the game's root folder.

Calling _AhnHsp_StartLauncher will execute HspL, and the game launcher can decrypt the executable file in real time through HspL.

Caution

When `ERROR_STARTLAUNCHER_WAITEVENT_FAIL(0x4B005)` is returned after `_AhnHsp_StartLauncher` is called, the game launcher may determine that a communication timeout occurred with the HackShield launcher. However, when HackShield is first executed, it launches HspL to protect itself. Most timeout errors occur when HackShield operates improperly and fails to establish communication. Therefore, if `ERROR_STARTLAUNCHER_WAITEVENT_FAIL` error occurs, the hacking program should be terminated before the game is executed.

5.2.5. `_AhnHsp_StartGame`

Decrypts the game's encrypted executable file through HspL. HspL must be running, and the target file must have been encrypted by HspCryptor.

Example

```
#define MAX_PATH_LEN          1024
//defined in HspLauncherLib.h.

typedef struct _AHN_GAMEEXECINFO
{
    CHAR szFilePath[MAX_PATH_LEN];
    CHAR szCommandLine[MAX_PATH_LEN];
} AHN_GAMEEXECINFO, *PAHN_GAMEEXECINFO;
// defined in HspLauncherLib.h.

CHAR szFilePath[MAX_PATH_LEN] = { 0, };
CHAR szCommandLine[MAX_PATH_LEN] = { 0, };
AHN_GAMEEXECINFO GameExecInfo;
...
// copy target file path.
sprintf( GameExecInfo.szFilePath, "%s", szFilePath );
// copy game information
sprintf( GameExecInfo.szCommandLine, "%s", szCommandLine );

//=====
//   Execute   Game   Program   through   the   Hackshield
Launcher(HspL.exe)
//=====
dwRet = _AhnHsp_StartGame( &GameExecInfo );
if( dwRet != ERROR_SUCCESS )
{
    error handling;
}
```

Caution

The second member in the AHN_GAMEEXECINFO structure contains data used for sending information (such as user ID) to the game client needed for the game launcher to execute the game. If the second member contains multiple data, a blank space is used to separate two pieces of data.

Example: `sprintf(GameExecInfo.szCommandLine, "%s", "GameInfo1 GameInfo2 GameInfo3");`

When the game client uses this option, game information can be acquired by `GetCommandLine()` function.

5.2.6. `_AhnHsp_CloseHandle`

Closes HspL when the game launcher is terminated.

Example

```
_AhnHsp_CloseHandle( );
```

5.3. Application Programming Interface

AhnHsp_StartLauncher

DESCRIPTION

Starts HspL, which must be stored in the game's root folder. In general, the game launcher uses this function. The HspL executed by this function decrypts the encrypted game file in real time.

SYNTAX

```
DWORD __stdcall  
_AhnHsp_StartLauncher(  
    IN LPCSTR szHspLauncherPath  
);
```

PARAMETERS

Parameter	Value	Description
szHspLauncherPath		Full path of HspL.exe

RETURN VALUE

ERROR_SUCCESS . (Value = 0x0000)

- Description: Returned when the HackShield launcher was successfully executed.
- Cause:
- Solution:

ERROR_STARTLAUNCHER_INVALIDPARAM (Value = 0x 4B001)

- Description: Returned when incorrect parameters are inputted.
- Cause:
- Solution:

ERROR_STARTLAUNCHER_MAKEMEMMAPNAME_FAIL(Value = 0x 4B002)

- Description: Returned when the memory map name was not created.

- Cause:
- Solution:

ERROR_STARTLAUNCHER_CREATEPROCESS (Value = 0x4B003)

- Description: Returned when the HackShield launcher process was not created.

- Cause:
- Solution:

ERROR_STARTLAUNCHER_SHAREMEMINIT_FAIL (Value = 0x4B004)

- Description: Returned when the shared memory map was not initialized.
- Cause:
- Solution:

ERROR_STARTLAUNCHER_WAITEVENT_FAIL (Value = 0x4B005)

- Description: Returned when event standby time is exceeded.
- Cause: When HspL is first executed, it launches HackShield to protect itself. If HspL detects a hack attack, HackShield will not be executed and HspL will not be created. The game launcher waits for an HspL event and then a standby timeout error occurs.
- Solution:

Others

WIN32 Defined Error

AhnHsp_StartGame

DESCRIPTION

Executes the encrypted game's executable file through the HspL. The inputted parameter includes the AHN_GAMEEXECINFO structure consisting of the full path of the game executable file and the options required for game execution. In general, the game developer provides options including user ID and separates different pieces of information with an empty space.

SYNTAX

```
DWORD __stdcall  
_AhnHsp_StartGame(  
    IN OUT PAHN_GAMEEXECINFO pAhnGameExecInfo  
);
```

PARAMETERS

Parameter	Value	Description
pAhnGameExecInfo		PAHN_GAMEEXECINFO - szFilePath: Full path of the file to execute - szCommandLine: Options required for game execution

RETURN VALUE

ERROR_SUCCESS . (Value = 0x0000)

- Description: Returned when HackShield launcher was successfully executed.
- Cause:
- Solution:

ERROR_STARTLAUNCHER_INVALIDPARAM (Value = 0x 4B001)

- Description: Returned when incorrect parameters are inputted.
- Cause:
- Solution:

ERROR_STARTLAUNCHER_ALEADYSTART (Value = 0x 4B009)

- Description: Returned when HackShield launcher was started.
- Cause:

- Solution:

ERROR_STARTLAUNCHER_INVALIDSESSION (Value = 0x 4B00C)

- Description: Returned when an invalid session value arises upon creation of the HackShield launcher and the session.
- Cause:
- Solution:

ERROR_MAKEREQSESSION_WAITEVENT_FAIL (Value = 0x 4B011)

- Description: Returned when an event standby timeout occurs between the HackShield launcher and the game launcher.
- Cause:
- Solution:

ERROR_MAKEREQSESSION_SETEVENT_FAIL (Value = 0x 4B019)

- Description: Returned when event setting fails between the HackShield launcher and the game launcher.
- Cause:
- Solution:

ERROR_STARTGAME_INVALIDPARAM (Value = 0x 4B028)

- Description: Returned when incorrect parameters are inputted.
- Cause:
- Solution:

ERROR_STARTGAME_MAKEFILEKEY_FAIL (Value = 0x 4B029)

- Description: Returned when file key creation fails.
- Cause:
- Solution:

ERROR_STARTGAME_SHAREMEMINIT_FAIL (Value = 0x4B02A)

- Description: Returned when shared Memory map initialization fails.
- Cause:
- Solution:

ERROR_STARTGAME_WAITEVENT_FAIL (Value = 0x4B02B)

- Description: Returned when event standby timeout occurs during data exchange between the game launcher and the HackShield launcher.
- Cause:
- Solution:

ERROR_STARTGAME_INITCRYPT_FAIL (Value = 0x4B02F)

- Description: Returned when encryption/decryption initialization fails.
- Cause:
- Solution:

ERROR_STARTGAME_ENCRYPT_FAIL (Value = 0x4B040)

- Description: Returned when data encryption failed.
- Cause:
- Solution:

ERROR_STARTGAME_HSPL_NOTEXEC (Value = 0x4B041)

- Description: Returned when the HackShield launcher is not running.
- Cause:
- Solution:

Others

WIN32 Defined Error

AhnHsp_CloseHandle

DESCRIPTION

Called to close HspL. Closes the handle that was used.

SYNTAX

```
void __stdcall  
_AhnHsp_CloseHandle( );
```

PARAMETERS

None.

RETURN VALUE

None.

6. Data File/Message Encryption

6.1. Overview

HsCryptoUtil is a data encryption/decryption SDK. Recent developments require encryption technologies that are stronger than ever. To reflect the current needs, Advanced Encryption Standard (AES), which is stronger than the Data Encryption Standard (DES), has been adopted as the benchmark. HsCryptoUtil provides stronger data encryption/decryption using 128bit AES.

6.1.1. Function

Data Encryption/Decryption

The file and message encryption/decryption library provides HsCryptLib.lib (for Windows), libhscrypt.so (for Linux), and HsCryptLib.h for the game developer to easily encrypt/decrypt messages and files.

6.1.2. Features

Interface Function (API)

Provides interface library so the developer can easily use HsCryptLib functions and check the return values. The game developer can easily encrypt/decrypt files and messages using the provided interface libraries.

Data Encryption Program

Provides HsCryptoUtil.exe, a file encryption tool, and allows for decryption of the encrypted files using the APIs of HsCryptLib.

6.1.3. System Architecture

HsCryptLib is provided as an SDL library, not as an independent executable file. The general architecture and operating principles of HsCryptLib are as follows:

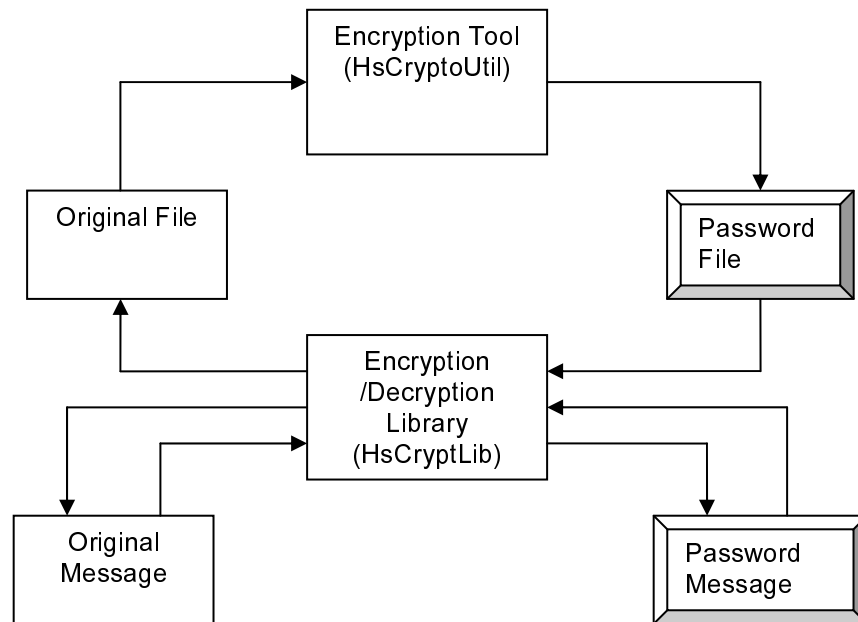


Figure 6-1 General Architecture and Operating Principles of HsCryptLib

Interface Library: HsCryptLib.lib (for Windows), libhscrypt.so (for Linux)

An interface library file that provides APIs to encrypt/decrypt the data.

HsCryptoUtil Program

An encryption program that provides a function to encrypt files using HsCryptLib.lib.

6.2. Application Programming

6.2.1. Programming Procedure

The game developer can implement HsCryptLib as follows:

1. Preparation: Check the list of the provided HsCryptLib file, and copy the necessary files.
2. Calling Encryption/Decryption Initialization Function: Create a function to initialize encryption/decryption and call the function before data decryption/encryption. If encryption/decryption is not initialized, file and message encryption/decryption will not be possible.
3. Calling File Decryption Function: Write a code to call a function that decrypts part or the entire encrypted file. The decrypted data will be outputted to the buffer.
4. Calling Message Encryption Function: Write a code to call a function that encrypts messages. The encrypted data will be outputted to the buffer.
5. Calling Message Decryption Function: Write a code to call a function that decrypts messages. The decrypted data will be outputted to the buffer.
6. Test whether the written source code properly operates.
7. Distribute to users.

6.2.2. Preparation

Do the following using HsCryptLib before starting programming:

6.2.3. HsCryptLib File

HsCryptLib File

Table 6-1 HsCryptLib File

File Name	Installation Folder	Description
HsCryptLib.h	[Program source folder]	Header File
HsCryptLib.lib	[Program source folder]	Windows Library File : Multi-thread and single thread library provided
libhscrypt.so	[Program source folder]	Linux Library File

Compiler Setting

The project file of the encryption/decryption program using HsCryptLib must have the HsCryptLib.lib (libhscrypt.so) file included in the library or source code. However, for projects using Windows HsCryptLib, it must be checked whether a multi-thread library or single-thread library is being used and the proper HsCryptLib.lib must be applied.

6.2.4. _HsCrypt_InitCrypt

When preparations are completed for programming, call _HsCrypt_InitCrypt. The _HsCrypt_InitCrypt function must be successfully called before the encryption/decryption keys used in data encryption/decryption can be created.

The following is an example of calling the _HsCrypt_InitCrypt function.

```
Example

typedef struct _HSCRYPT_KEYINFO
{
    BYTE          byInitKey[HSCRYPTLIB_INITKEY_SIZE];          //
Initialization key
    BYTE          AesEncKey[HSCRYPTLIB_KEY_SIZE];              //
//encryption key
    BYTE          AesDecKey[HSCRYPTLIB_KEY_SIZE];              //
//Decryption key
} HSCRYPT_KEYINFO, *PHSCRYPT_KEYINFO;

HSCRYPT_KEYINFO HsKeyInfo;
memcpy( HsKeyInfo.byInitKey, pbyInitKey, HSCRYPTLIB_INITKEY_SIZE );

dwRet = _HsCrypt_InitCrypt ( &HsKeyInfo );
```

In the above example, the HSCRYPT_KEYINFO structure is declared and the initialization key is inputted in byInitKey. The size of the initialization key in HsKeyInfo.byInitKey must be set as 16 bytes.

Calling _HsCrypt_InitCrypt will allocate the key values to AesEncKey (encryption key) and AesDecKey (Decryption key) in the HSCRYPT_KEYINFO structure.

These key values will be used in the encryption/decryption of messages and files.

Caution

The encryption and decryption keys created by the initialization key must match each other for file and message encryption/decryption.

6.2.5. `_HsCrypt_GetEncMsg`

Encrypts messages. Initializes encryption/decryption and calls `_HsCrypt_GetEncMsg` to encrypt data. The encrypted data will be outputted to the buffer.

Example

```
dwRet = _HsCrypt_GetEncMsg (
byPlainMsg,           // [in] to encrypt buffer
sizeof(byPlainMsg),   // [in] to encrypt Size
HsKeyInfo.AesEncKey,  // [in] Encryption key
byEncMsg              // [out] encrypted buffer
);
```

Note

The message must be the same size before and after encryption.

6.2.6. `_HsCrypt_GetDecMsg`

Decrypts messages. Initializes encryption/decryption and calls `_HsCrypt_GetDecMsg` to decrypt the data. The decrypted data will be outputted to the buffer.

Example

```
dwRet = _HsCrypt_GetDecMsg (
    byEncMsg,           // [in] Buffer to decrypt
    sizeof(byEncMsg),   // [in] Decryption size
    HsKeyInfo.AesDecKey, // [in] Decryption key
    byDecMsg            // [out] Decrypted buffer
);
```

Note

The message must be the same size before and after decryption.

6.2.7. _HsCrypt_FRead

This function decrypts part or the entire file using the file structure pointer. It initializes the encryption/decryption key and calls the fseek function to move to the file pointer and encrypt the data. The decrypted data will be outputted to the buffer.

```
Example
dwRet = _HsCrypt_FRead (
    byPlainBuf,          // [out]Decrypted buffer
    dwDecSize,           // [in] Decryption Size
    InputStream,         // [in] File pointer to read
    HsKeyInfo.AesDecKey, // [in] Decryption key
    &dwReadLen           // [out] Decrypted Size
);
```

Performs decryption/encryption using the block password (Block Cipher). The game developer can specify a block to decrypt and store only the necessary part in the buffer for return.

To decrypt the entire file, set the beginning of the file as the file structure pointer using the fseek function and input the entire file size as the second parameter of _HsCrypt_FRead.

Caution

Input the decryption key that corresponds to the key that encrypted the file. If key management is difficult, use the key that initialized the encryption/decryption key and the decryption key created by _HsCrypt_InitCrypt.

6.3. Application Programming Interface

_HsCrypt_InitCrypt

DESCRIPTION

Initializes the encryption/decryption function.

SYNTAX

```
DWORD __stdcall  
_HsCrypt_InitCrypt (  
    IN OUT PHSCRYPT_KEYINFO pHsKeyInfo  
);
```

PARAMETERS

Parameter	Description
PHSCRYPT_KEYINFO	[in][out]
Structure Definition	Encryption/Decryption key structure
typedef struct _HSCRYPT_KEYINFO	
{	
BYTE byInitKey[HSCRYPTLIB_INITKEY_SIZE];	
BYTE AesEncKey[HSCRYPTLIB_KEY_SIZE];	Initialization Key
BYTE AesDecKey[HSCRYPTLIB_KEY_SIZE];	(16bytes)
} HSCRYPT_KEYINFO, *PHSCRYPT_KEYINFO;	encryption key
	(550bytes)
	Decryption key
	(550bytes)

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned after initialization success.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_INITCRYPT_INVALIDPARAM (Value = 0x0001B002)

- Description: Returned when incorrect parameters were inputted.
- Cause:
- Solution:

Others

WIN32 Defined Error (Value = WIN32 Defined)

REMARKS

This function must be called to set the data needed for encryption/decryption. Allocate `byInitKey` of the `HSCRYPT_KEYINFO` structure, and call the `_HsCrypt_InitCrypt` function to get `AesEncKey` (encryption key) and `AesDecKey` (Decryption key). The game developer can encrypt/decrypt messages and files using the key.

HsCrypt_GetEncMsg

DESCRIPTION

Encrypts the message and output to the encrypted data buffer.

SYNTAX

```
DWORD __stdcall  
_HsCrypt_GetEncMsg (  
    IN PBYTE pbyInput,  
    IN UINT nInLength,  
    IN PBYTE pAesEncKey,  
    OUT PBYTE pbyOutput  
);
```

PARAMETERS

Parameter	Description
pbyInput	[in] Buffer to encrypt
nInLength	[in] Size to encrypt
pAesEncKey	[in] Encryption key
pbyOutput	[out] Encrypted buffer

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned after successful message encryption.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_GETENCMMSG_INVALIDPARAM (Value = 0x0001B003)

- Description: Returned when incorrect parameters are inputted.
- Cause:
- Solution:

Others

WIN32 Defined Error (Value = WIN32 Defined)

HsCrypt_GetDecMsg

DESCRIPTION

Decrypts the message and outputs to the decrypted data buffer.

SYNTAX

```
DWORD __stdcall  
_HsCrypt_GetDecMsg (  
    IN PBYTE pbyInput,  
    IN UINT nInLength,  
    IN PBYTE pAesDecKey,  
    OUT PBYTE pbyOutput  
);
```

PARAMETERS

Parameter	Description
pbyInput	[in] Buffer to decrypt
nInLength	[in] Size to decrypt
pAesEncKey	[in] Decryption key
pbyOutput	[out] Decrypted buffer

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned after successful message encryption.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_GETDECMMSG_INVALIDPARAM (Value = 0x0001B004)

- Description: Returned when incorrect parameters are inputted.
- Cause:
- Solution:

Others

WIN32 Defined Error (Value = WIN32 Defined)

HsCrypt_FRead

DESCRIPTION

Only decrypts the desired parts of the file and outputs the data to the buffer.

SYNTAX

```
DWORD __stdcall
_HsCrypt_FRead (
    OUT LPVOID lpOutBuffer,
    IN DWORD dwDecryptSize,
    IN FILE *pInputStream,
    IN PBYTE pAesDecKey,
    OUT PDWORD pdwReadLen
);
```

PARAMETERS

Parameter	Description
lpOutBuffer	[out] Decrypted buffer
dwDecryptSize	[in] Size to decrypt
pInputStream	[in] File structure pointer to decrypt
pAesDecKey	[in] Decryption key
pdwReadLen	[out] Decrypted size

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned after successful decryption.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_INVALIDPARAM (Value = 0x0001B005)

- Description: Returned when incorrect parameters are inputted.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_GETFILELEN (Value = 0x0001B009)

- Description: Returned when the file size is not acquired.

- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_SIZEZERO (Value = 0x0001B00B)

- Description: Returned when the file size is 0.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_GETPOSITION (Value = 0x0001B00A)

- Description: Returned when the current file pointer location was not acquired.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_FSEEK (Value = 0x0001B00C)

- Description: Returned when acquiring the location of the current file pointer block failed.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_DECRYPT_RANGE (Value = 0x0001B006)

- Description: Returned when the size of the block to decrypt is larger than the file.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_DECRYPT_FREAD (Value = 0x0001B007)

- Description: Returned when file reading fails.
- Cause:
- Solution:

ERROR_HSCRYPTLIB_FREAD_DECRYPT_GETDECMMSG (Value = 0x0001B008)

- Description: Returned when message decryption fails.

- Cause:
- Solution:

ERROR_HSCRYPTLIB_EXCEPTION (Value = 0x0001B001)

- Description: Returned when an exception occurs.
- Cause:
- Solution:

Others

WIN32 Defined Error (Value = WIN32 Defined)

7. User Authority Support

7.1. Overview

HsUserUtil provides various functions that enable HackShield game hacking protection functions for the users logged in with user accounts (not administrator accounts) on the NT-series OS.

7.1.1. Functions

Support for Non-administrator Account Game

HsUserUtil.lib allows users logged onto general user accounts (not administrator accounts) to run the game client and HackShield game hacking protection functions.

7.1.2. Features

Interface Function (API)

Provides an interface library so the developer can easily use HsUserUtil functions and the returned results. Using the provided interface library, the game developer can allow general user accounts to run the game client and HackShield depending on the game developer's policies.

Test Program

Provides HsUserUtilTest.exe, a test game client program implemented by using the APIs of HsUserUtil. Game developers can check the functions and the sample code of HsUserUtil by referring to the test program and can apply this to the development of client programs.

7.1.3. System Architecture

HsUserUtil is provided as an SDL library, not as an independent executable file. The general architecture and operating principles of HsUserUtil are as follows:

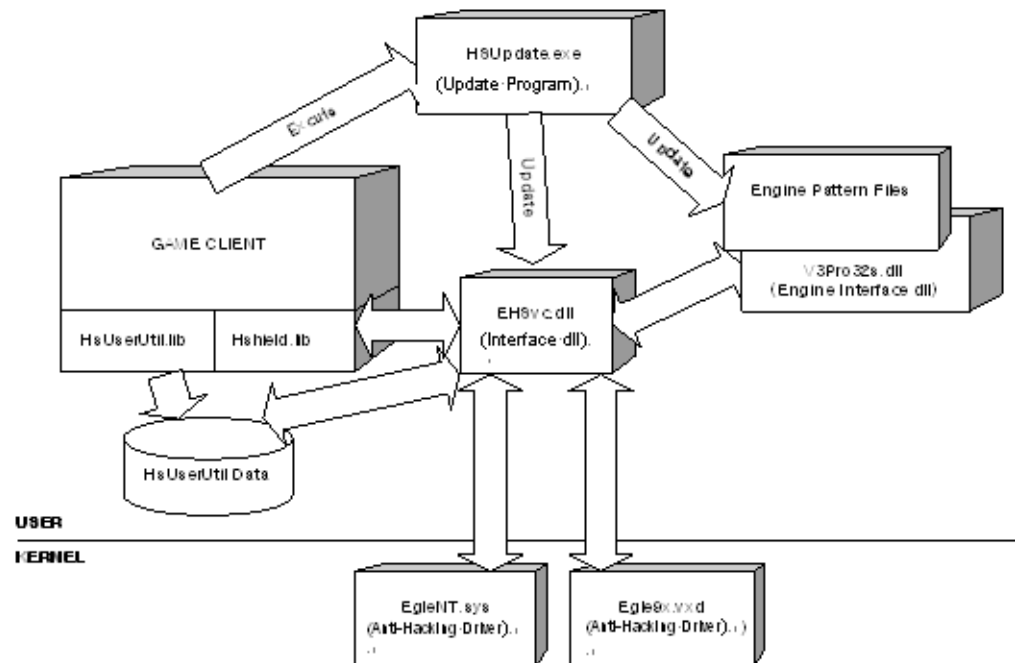


Figure 7-1 General Architecture and Operating Principles of HsUserUtil

HsUserUtil.lib (Interface Library)

As an interface file, provides APIs that allow a general user account to start the game program.

HsUserUtil Data

Stores shadow account information and related data allowing general user accounts to start the game program.

EhSvc.dll (HackShield Service Module)

HackShield service module that is loaded to the actual game program and performs the hacking protection function. If the game is running under general user authority, it uses HsUserUtil data to help the game run properly and ensure the game protection function works properly.

7.2. Application Programming

7.2.1. Programming Procedure

The game client developer must implement HsUserUtil functions in the following order.

Note

This document describes a case in which HSUserUtil is used for the game client program. If there is a game launcher program in addition to the game client program, the game developer can also use HSUserUtil library in the game launcher program. HSUserUtil library must be applied according to the game structure.

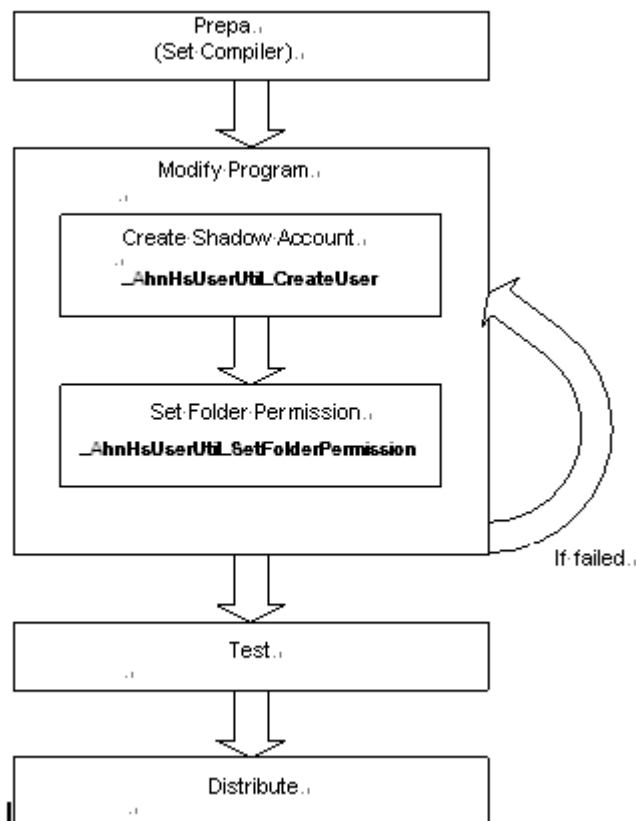


Figure 7-2 HsUserUtil Programming Procedure

1. Prepare: Check the list of the provided HsUserUtil files, and copy necessary files.
2. Write shadow account creation function: Write a function that calls the shadow account creation function so that general user accounts can execute the game and use the game hacking protection function.
3. Write NTFS authority setting function: After calling the service starting function, write an NTFS authority setting function calling code so that general user accounts can also write files for the game.

4. Test whether the source code is properly operating.
5. Distribute to client users.

7.2.2. Preparation

Do the following using HsUserUtil before starting programming:

7.2.2.1.HsUserUtil File

HsUserUtil File

Table 7-1 HsUserUtil File

File Name	Installation Folder	Description
HsUserUtil.h	[Game folder] source	Header File
HsUserUtil.lib	[Game folder] source	Library File

Compiler Setting

Include the HsUserUtil.lib file in the project file of the game client program that uses HsUserUtil in the library of the source code list.

7.2.3. _AhnHsUserUtil_CreateUser

When preparations are completed for programming, call _AhnHsUserUtil_CreateUser. After the _AhnHsUserUtil_CreateUser function is successfully called, general user accounts can execute the game and use the game hacking protection function.

At this time, the _AhnHsUserUtil_CreateUser function is called by the administrator account in the game program or game launcher program.

The following is an example of calling the _AhnHsUserUtil_CreateUser function.

Example

```
dwRet = _AhnHsUserUtil_CreateUser ( );
```

Note

_AhnHsUserUtil_CreateUser creates a new user only when shadow account user information does not exist or the user did not log in with user information. A logic for deleting accounts complying with the shadow account naming rule is included so that unnecessary accounts are automatically deleted when a new shadow

account is created.

7.2.4. AhnHsUserUtil_SetFolderPermission

If the game program is installed in the NTFS volume, a general user authority holder will not have file writing authority and will not be able to properly run the game. For example, even if the update program for the game module was executed, a general user authority holder may not be able to write the latest files in the game folder or may fail to write game data during game execution.

To give general user accounts file writing authority in the game installation folder, the game developer must use the AhnHsUserUtil_SetFolderPermission function and give the NTFS writing authority. The following is an example of calling this function.

Example

```
dwRet = AhnHsUserUtil_SetFolderPermission("game path");
```

To give NTFS writing authority to a general user account, the full path, not an alternative or incorrect path, must be given.

If the game installation path is not the NTFS volume or if the function is called in Windows 95, 98, or ME PC that does not use the NTFS file system, the function will not be invalid.

When this function is executed, NTFS writing authority will be given to the user group for the corresponding path. However, this function must be called by the administrator authority.

Caution

The game developer must note that the game client program execution path is different depending on the user. Changing the NTFS authority in the folder without knowing where the game will be installed – C:\, background, or Windows folder – may cause security vulnerabilities.

7.3. Application Programming Interface

_AhnHsUserUtil_CreateUser

DESCRIPTION

Creates a shadow account to prevent game hacking for the users logged on with general user authority.

SYNTAX

```
DWORD __stdcall  
_AhnHsUserUtil_CreateUser ( );
```

PARAMETERS

None.

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- Description: Returned after the shadow account is successfully created.
- Cause:
- Solution:

HSUSERUTIL_ERR_NOT_ADMIN (Value = 0x0005A002)

- Description: Returned when the currently logged-on account is not the Administrator account.
- Cause:
- Solution: This error must not be processed if the general user authority can call `_AhnHsUserUtil_CreateUser`, although this may depend on the application type.

HSUSERUTIL_ERR_NOT_NT (Value = 0x0005A003)

- Description: Returned when the system is not NT series.
- Cause:
- Solution:

HSUSERUTIL_ERR_DELSHADOWACNT_FAIL (Value = 0x0005A005)

- Description: Returned when shadow account was not deleted.
- Cause:
- Solution:

HSUSERUTIL_ERR_DELHIDEIDINFO_FAIL (Value = 0x0005A006)

- Description: Returned when shadow account deletion fails on the Windows XP startup screen.
- Cause:
- Solution:

HSUSERUTIL_ERR_DELSHADOWACNTINFO_FAIL (Value = 0x0005A007)

- Description: Returned when shadow account deletion failed.
- Cause:
- Solution:

HSUSERUTIL_ERR_ADDSHADOWACNT_FAIL (Value = 0x0005A008)

- Description: Returned when a shadow account was not created.
- Cause:
- Solution:

REMARKS

The `_AhnHsUserUtil_CreateUser` function has administrator authority and can create a shadow account when logged on. If a user is logged in with general user authority and wishes to use HackShield game protection functions, this function must be called at least once and a shadow account must be created.

AhnHsUserUtil_SetFolderPermission

DESCRIPTION

Gives NTFS writing authority to general user accounts for directories that have game clients installed so that users logged on with general user accounts can write files for update and execution.

SYNTAX

```
DWORD __stdcall  
_AhnHsUserUtil_SetFolderPermission (  
    LPTSTR szPath  
);
```

PARAMETERS

Parameter	Value	Description
szPath		Full path to set the NTFS authority Example: C:\Program Files\My Company\My Game

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- Description: Returned when NTFS authority was not given to the folder.
- Cause:
- Solution:

HSUSERUTIL_ERR_NOT_ADMIN (Value = 0x0005A002)

- Description: Returned when the currently logged-on account is not the Administrator account.
- Cause:
- Solution: This error must not be processed if the general user authority can call `_AhnHsUserUtil_CreateUser` although it may differ depending on the application type.

HSUSERUTIL_ERR_SETFLDRPERMISSION_FAIL (Value = 0x0005A10A)

- Description: Returned when NTFS authority was not given to the folder.
- Cause:
- Solution:

HSUSERUTIL_ERR_GETGROUPSID_FAIL (Value = 0x0005A10B)

- Description: Returned when a group SID was not acquired
- Cause:
- Solution:

HSUSERUTIL_ERR_GETSECINFO_FAIL (Value = 0x0005A10C)

- Description: Returned when SD and DACL are not acquired.
- Cause:
- Solution:

HSUSERUTIL_ERR_ADDNEWACE_FAIL (Value = 0x0005A10D)

- Description: Returned when new ACE was not created.
- Cause:
- Solution:

HSUSERUTIL_ERR_ADDNEWDACL_FAIL (Value = 0x0005A10E)

- Description: Returned when a new DACL was not created.
- Cause:
- Solution:

HSUSERUTIL_ERR_COPYOLDDACL_FAIL (Value = 0x0005A10F)

- Description: Returned when existing DACL was not copied to the new DACL.
- Cause:
- Solution:

HSUSERUTIL_ERR_ADDNEWACETONEWDACL_FAIL (Value = 0x0005A110)

- Description: Returned when a new ACE was not added to DACL.
- Cause:
- Solution:

REMARKS

If you can arbitrarily define the path of the game client program, this function must be used with caution. If you installed the game in a root directory such as the C: or D: drive, calling this function will give the NTFS authority over the entire drive. This may create security vulnerabilities, so this function must be used with care.

If the folder has multiple folders and files, it may take a few seconds or minutes to set the NTFS authority after calling this function. However, this occurs only when NTFS authority is initially given and does not have any effect once the authority is given.

8. HackShield Update (For HackShield Pro)

8.1. Overview

The HackShield update function is designed to help the game developer build within the patch period and easily update HackShield through the launcher. HackShield updates support quick HackShield module and engine patches and allow for prompt measures against hacking tools.

8.1.1. Functions

HackShield Module Update

HackShield updates are provided to allow easy updating of the HackShield module when a HackShield patch is released.

Engine Update

HackShield updates can update signatures and the heuristic engines to allow for an immediate response to hacking tools.

8.1.2. Features

Interface Function (API)

Called by the game launcher or executable file through the use of the APIs in HSUpChk.lib to update HackShield modules and engines.

Update File Creation Program

Provides update server configuration file to configure the server to be updated and builds multiple update servers through multiple URLs.

Test Program

Provides Amazon.exe, a test program implemented by HSUpChk.lib APIs. Amazon.exe provides HackShield test and HackShield update test functions.

8.1.3. System Architecture

HackShield update provides HSUpChk.lib for the client to call the API and perform updates. The general architecture and operating principles of HackShield updates are as follows:

HSUpChk.lib (HackShield update Library)

Used by the client. When the update function is called, HsUpdate.exe will be also called to update HackShield module.

HSUpdate.env (Update Environment File)

Created by the HSUpSetEnv.exe tool. Contains update server configuration information.

HSUpSetEnv.exe (Update Environment File Creation Program)

Creates the HSUpdate.env update configuration file. Configures the update server.

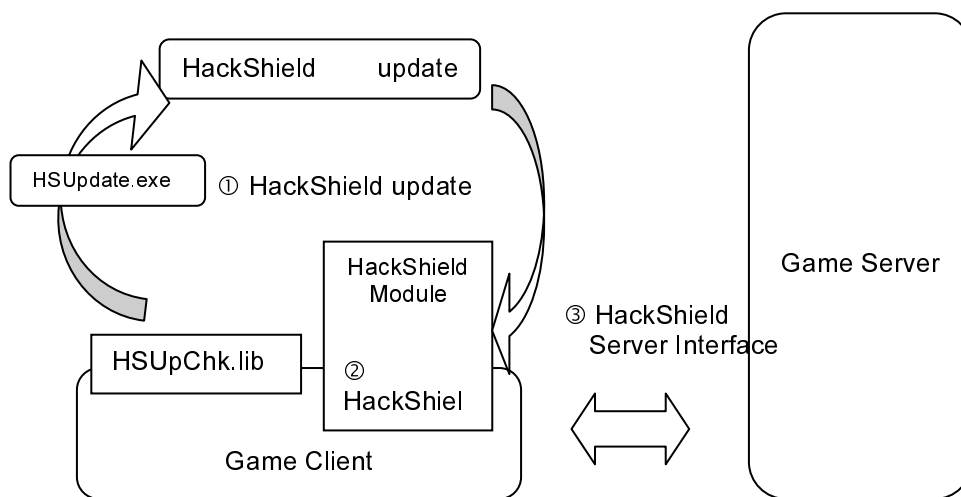


Figure 8-1 Operating Principles of HackShield update

- ① Calls `_AhnHS_HSUpdate`, an API provided by `HSUpchk.lib`, and accesses the HackShield update server to update the HackShield module and engine.
- ② Calls `_AhnHS_Initialize` and `_AhnHS_Start` functions provided by `HSshield.lib` to execute HackShield.
- ③ Accesses the game server and starts server interfaces.

8.2. Application Programming

This chapter describes how to update HackShield using APIs provided by HSUpChk.lib.

Note

The sample codes contained in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

8.2.1. Programming Application

Do the following using HSUpChk.lib before starting programming.

8.2.1.1. Update-related File

Update-related File

Table 8-1 update-related File

File Name	Installation Folder	Description
HSUpChk.lib	[Program source folder]	Header file to be used by the server
HSUpChk.h	[Program source folder]	DLL file to be used by the server
AhnUpCtl.dll	[Game]\HShield	Update dll
AhnUpGS.dll	[Game]\HShield	Update dll
AspINet.dll	[Game]\HShield	Update dll
Bz32Ex.dll	[Game]\HShield	Compression dll
HSInst.dll	[Game]\HShield	UI dll
HSUpdate.env	[Game]\HShield	Update Configuration File
HSUpdate.exe	[Game]\HShield	Update Executable file
mspatcha.dll	[Game]\HShield	MS patch dll
V3Hunt.dll	[Game]\HShield	Update dll

Table 8-2 Files to be Installed in the Update Server

PatchSet Path: [HackShield SDK] \PatchSet\

File Name	Description
ahn.ui	Update Information File (Engine File Version)
ahn.uic	Update Information File (Engine File Version)
ahni2.dll	update File
ahnupctl.dll	update File
autoup.exe	update File
v3bz32.dll	update File
patch\39\3n.mh-	Heuristic engine File (Compressed File)
patch\39\ahn.ui	Update Information File (Patch File Version)
patch\39\ahn.uic	Update Information File (Patch File Version)
patch\39\ahnupctl.dl-	update File (Compressed File)
patch\39\ahnupgs.dl-	update File (Compressed File)
patch\39\aspinet.dl-	update File (Compressed File)
patch\39\bz32ex.dl-	update File (Compressed File)
patch\39\ehsvc.dl-	HackShield Interface DLL (Compressed File)
patch\39\hshield.da-	HackShield-related dat file (Compressed File)
patch\39\hsinst.dl-	update File (Compressed File)
patch\39\hsupdate.ex-	update Executable file (Compressed File)
patch\39\mspatcha.dl-	update File (Compressed File)
patch\39\psapi.dl-	Process Status Helper DLL (Compressed File)
patch\39\v3hunt.dl-	update File (Compressed File)
patch\39\v3inetgs.dl-	update File (Compressed File)
win\eb\b_v3_echo_sl\v3pr o32s.dl-	Hacking Tool detection engine Interface DLL (Compressed File)
win\eb\b_sign_sl\v3warp ds.v3-	Hacking Tool Pattern engine File (Compressed File)
win\eb\b_sign_sl\v3warp ns.v3-	Hacking Tool Pattern engine File (Compressed File)

8.2.1.2.Application

Server Application

1. Configures an update server that is accessible through FTP or HTTP through IIS setting.

Note

FTP must allow separate account setting and anonymous access.

FTP must allow passive-mode access. To check whether passive-mode access is allowed, access through the IE and check the file list.

2. Copy all files under [HackShield SDK] \PatchSet folder to the server.
For the file list, see Table 8-2.

Client Application

1. Include the HSUpChk.lib library file in the project.
2. Include the provided HSUpChk.h file in the source file.
3. Write the HackShield update function (_AhnHS_HSUpdate).

```
DWORD dwRet = 0;
TCHAR szFullFilePath[MAX_PATH];

// Define a path for HackShield folder.
_tcscat ( szFullFilePath, _T( "\\HShield" ) );

// Call _AhnHS_HSUpdate function.
dwRet = _AhnHS_HSUpdate ( szFullFilePath, 1000 * 600 );
if ( dwRet != ERROR_SUCCESS )
{
    // error
    switch ( dwRet )
    {
        case 0x30000010:
            // HS_ERR_ENVFILE_NOTREAD
            break;
        case 0x30000020:
            // HS_ERR_ENVFILE_NOTWRITE
            break;
        case 0x30000030:
            // HS_ERR_NETWORK_CONNECT_FAIL
            break;
        ...
    }
}
```

4. Copy all files in the [HackShield SDK]\Bin\Update folder to the [Game Directory]\HShield folder of the target client.
5. Create Update folder in the [Game Directory]\HShield\ folder, and copy ahn.ui, ahn.uic, ahni2.dll, ahnupctl.dll, autoup.exe, and v3bz32.dll file to the [HackShield SDK]\Bin\Update folder.

After copying, the folder file structure is as follows:

```
Example - Installation Directory Structure for Smart Update
Module

[Game Directory]\HShield\

3N.mhe
AhnUpCtl.dll
AhnUpGS.dll
AspINet.dll
Bz32Ex.dll
EhSvc.dll
HShield.dat
HSInst.dll
HSUpdate.env
```

```
HSUpdate.exe
mspatcha.dll
psapi.dll
V3Hunt.dll
V3InetGS.dll
v3pro32s.dll
v3warpds.v3d
v3warpns.v3d

[Game Directory]\HShield\Update\
ahn.ui
ahn.uic
ahni2.dll
ahnupctl.dll
autoup.exe
v3bz32.dll
```

6. Execute the [HackShield SDK]\Bin\Util\HSUpSetEnv.exe File.

Caution

HSUpSetEnv.exe sets the update setting file. This tool must not be distributed.

7. Create HSUpdate.env by referring to [11.8 HSUpSetEnv Tool Usage](#).
8. Copy HSUpdate.env file to [GameDirectory]\HShield folder.

Caution

For normal updates, the parameter first set to _AhnHS_HSUpdate function ([Game Directory]\HShield) must have all update modules in the [HackShield SDK]\Bin\Update folder.

8.3. Application Programming Interface

AhnHS_HSUpdate

DESCRIPTION

Updates HackShield files.

SYNTAX

```
DWORD __stdcall  
_AhnHS_HSUpdate(  
    LPCTSTR szUpdateDir,  
    DWORD dwTimeOut  
);
```

PARAMETERS

Parameter	Value	Description
szUpdateDir	LPCTSTR	Folder with the update file installed
dwTimeOut	DWORD (milliseconds)	Update timeout. Set as INFINITE when this value is 0. * The recommended value is 60000 (10 minutes) depending on the network status. It is recommended to avoid 0.

szUpdateDir

The [in] szUpdateDir parameter sets the absolute paths of the distributed HackShield-related files. The folder must have all update modules. The GetModuleFileName function should be used to set a correct absolute path. The GetCurrentDirectory function may not return the desired path.

dwTimeOut

Means timeout after calling the [in] Function. The second argument is in milliseconds. If no response arrives in this time, the call will be considered a failure. 10 minutes should be given for standby considering network speed. If the update fails, check the game execution status according to the game developer's policy.

RETURN VALUE

HACKSHIELD_ERROR_SUCESS (Value = 0x00000000)

- Description: Successful update
- Cause:
- Solution:

HSERROR_ENVFILE_NOTREAD (Value = 0x30000010)

- Description: Cannot read HSUpdate.env file.
- Cause:
- Solution:

HSERROR_ENVFILE_NOTWRITE (Value = 0x30000020)

- Description: Cannot write HSUpdate.env file.
- Cause:
- Solution:

HSERROR_NETWORK_CONNECT_FAIL (Value = 0x30000030)

- Description: Cannot connect to the server.
- Cause:
- Solution:

HSERROR_LIB_NOTEDIT_REG (Value = 0x30000050)

- Description: An error occurred while inputting the network result.
- Cause:
- Solution:

HSERROR_NOTFINDFILE (Value = 0x30000060)

- Description: Cannot file HackShield update program-related files.
- Cause:
- Solution:

HSERROR_PROTECT_LISTLOAD_FAIL (Value = 0x30000070)

- Description: Cannot file HSUpdate.pt authentication file.
- Cause:
- Solution:

HSERROR_HSUPDATE_TIMEOUT (Value = 0x30000090)

- Description: Update timeout occurred in HackShield update program.
- Cause:
- Solution:

REMARKS

The update function must be called before calling the initialization function (_AhnHS_Initialize). Can be called by an executable file like a game process launcher apart from the HackShield function.

9. Monitoring Service (For HackShield Pro)

9.1. Overview

AhnLab's HackShield Hacking Monitoring System centrally monitors hack attacks and errors in the program that is running HackShield. Hack attacks and errors can cause great harm to the entire service so it is critical to monitor hack attacks and errors in real time and take measures as early as possible. AhnLab's HackShield Hacking Monitoring System provides the report function and provides useful statistical data on the service situation.

9.1.1. Functions

Real-time Hacking/error Monitoring

Monitors hacking and errors in the HackShield-applied program and the data that HackShield provided in real time.

Report

Provides various statistical reports based on the collected log data.

Policy Management

Sets various policies depending on the situation and backs up the DB.

9.1.2. Features

Interface Function (API)

Provides an interface DLL and library so the developer can easily use Ehsvc functions and check the results. Using the provided interface DLL and library, developers can check the integrity of the game file and the operation of HackShield.

Monitoring Server Program

Provides a monitoring program so the developer can manage and check the Oracle DB in real time, so that hacking and error data from the client can be checked in real time.

Test Program

Provides Amazon.exe, a test program, implemented by the Ehsvc APIs. Amazon.exe, provides existing HackShield test and Ehsvc test functions.

9.1.3. System Architecture

Provides HShield.lib and EHsvc.dll which are applied to the server and client. The general architecture and the operating principles of the monitoring service are as follows.

Ehsvc.dll (Interface dll)

Provides an API that can set basic information to notify the monitoring server of a hacking or error occurrence.

HShield.lib (HackShield Library)

Provides an API that can set basic information to notify the monitoring server of a hacking or error occurrence.

HSMS_Setup.exe (Monitoring Server Side Installation File)

A server-side program that manages the data from the client in the DB and allows easy management on the web.

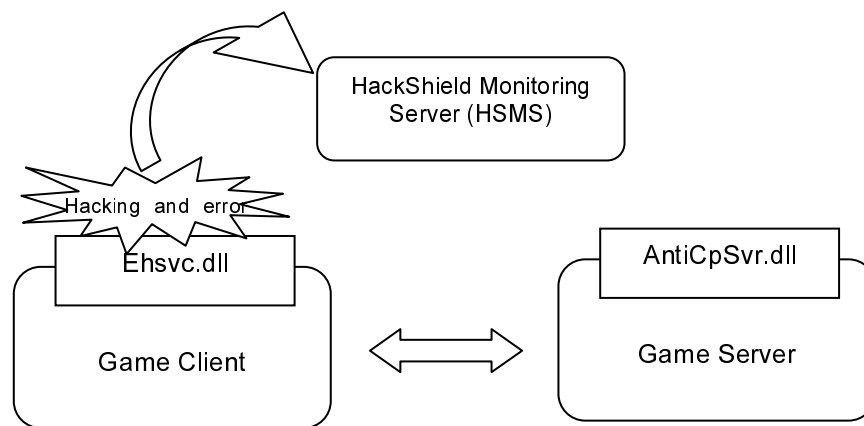


Figure 9-1 Operating Principles of the Monitoring Service

When the game client calls the `_AhnHS_StartMonitor` function, the client will notify the monitoring server of any hacking and error data.

The monitoring server manages the client program using a certain game code. The data that the client sends in relation to the game code will be managed in the DB. The developer can easily access the DB through the web.

9.2. Application Programming

Sets monitoring server information and user information using the API provided by Ehsvc.

Note

The sample codes contained in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

9.2.1. Programming Application

Do the following before starting programming.

9.2.1.1. Monitoring-related File

Monitoring-related File

Table 9-1 Monitoring-related File

File Name	Installation Folder	Description
HShield.h	[Program source folder]	Header file to be used in the client. HackShield functions included.
HShield.lib	[Program source folder]	Library file to be used in the client. HackShield function included.
EHSvc.dll	[Program source folder]	Dll file to be used in the client. HackShield functions included.

9.2.1.2.Application

Server Application

1. Install the HackShield Monitoring Server Program in the server by referring to the HackShield Monitoring Server Installation Guide.
2. Run the server by referring to the HackShield Monitoring Server Operation Guide.

Client Application

1. Include HShield.lib file in the project.
2. Include the provided HShield.h file in the source file.
3. Create a parameter for the AHNHS_EXT_ERRORINFO structure defined in HShield.h and initialize it.

```
AHNHS_EXT_ERRORINFO HsExtError = { 0, };
```

4. Substitute the data in szServer(Monitoring Serveraddress), szUserId(User account), and szGameVersion(game Version) in HsExtError structure.

```
HsExtError.szServer = "127.0.0.1" //Monitoring address
HsExtError.szUserId = "Test" //User ID
HsExtError.szGameVersion = "3.0.0.1" //Game Version
```

5. Send the structure parameters created in the above step and the full path of Ehsvc.dll as parameters, and call the _AhnHS_StartMonitor function.

Caution

Must be called before the _AhnHS_Initialize function. Otherwise, no error will be sent before HackShield is executed.

```
lstrcat (szFullFileName, _T("\\HShield\\Ehsvc.dll" ));

dwRet = _AhnHS_StartMonitor ( HsExtError // [in]
                             szFullFileName // [in]
                             );

if( dwRet != ERROR_SUCCESS )
{
    // If a fail occurs, it means monitoring is not executed.
    // Store only error logs.
}

dwRet = _AhnHS_Initialize ( ...
```

6. If the user ID cannot be acquired in Step 4, call the _AhnHS_SetUserId function when the user ID is available to set the user ID.

9.3. Application Programming Interface

AhnHS_StartMonitor

DESCRIPTION

Sets server information to send the data if hacking or errors occur.
Called before _AhnHS_Initialize function.

SYNTAX

```
int  
__sdtdll  
_Ahn_StartMonitor ( IN AHNHS_EXT_ERRORINFO HsExtErrorInfo,  
                    IN LPCSTR szFileName  
                    );
```

PARAMETERS

Parameter	Value	Description
HsExtErrorInfo	AHNHS_EXT_ERRORINFO	Structure with server URL address, User ID, and game version
szFileName	LPCSTR	Full path of Ehsvc.dll

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- Description: Returned when the function was successfully called.
- Cause:
- Solution:

HS_ERR_INVALID_FILES (Value = 0x1C001)

- Description: Incorrect data input.
- Cause:
- Solution:

HS_ERR_UNKNOWN (Value = 0x1C002)

- Description: Unknown error.

- Cause:
- Solution:

Example

The following is an example of calling the `_AntiCpSvr_StartMonitor` function.

```

Example
=====
AHNHS_EXT_ERRORINFO HsExtError;          //Structure defined in
HShield.h

HsExtError.szServer = "127.0.0.1"        //Monitoring address
HsExtError.szUserId = "Test"             //User ID
HsExtError.szGameVersion = "3.0.0.1"     //Game Version

lstrcat (szFullFileName, _T("\\HShield\\Ehsvc.dll" ));

dwRet = _AhnHS_StartMonitor (  HsExtError      // [in]
                               szFullFileName  // [in]
                               );

if( dwRet != ERROR_SUCCESS )
{
    // If a fail occurs, it means monitoring is not executed.
    // Store only error logs.
}

dwRet = _AhnHS_Initialize ( ...
=====

```

AhnHS_SetUserId

DESCRIPTION

Stores the user ID of the message to be sent to the monitoring server. Although _AhnHS_StartMonitor receives user ID, user information may not be provided when HackShield is initialized. Call this function when the user ID is provided to get user information. Before the ID is acquired, error data is sent without ID information.

SYNTAX

```
void __stdcall  
_AhnHS_SetUserId ( IN LPCSTR szUserID )
```

PARAMETERS

Parameter	Value	Description
szUserUD	LPCSTR	User information of the game client

RETURN VALUE

None.

Example

The following is an example of calling the _AhnHS_SetUserId function.

Example

```
AhnHS_SetUserId ( IN LPCSTR szUserID );
```


10. LMP (For HackShield Pro)

10.1.Overview

The Local Memory Protection (LMP) function detects memory manipulation at the client end when Packer 2 cannot protect the server interface memory. This function is similar to the memory manipulation detection function at the existing server interface. However, the LMP function allows the client to protect the memory without passing through the server.

10.1.1. Functions

Memory Protection of Executable file

Detects manipulation of the code area in the memory of the executable file.

Memory Protection of DLL File

Detects manipulation in the code area of the specified DLL file, and even protects the files not directly built by the corresponding DLL.

Note

To protect the DLL file, only statically loaded files can be protected. Dynamically loaded DLL files cannot be projected. (Dynamic loading by LoadLibrary is not supported by the current version.)

Memory Integrity Verification

Checks the hooking status of the system file and verifies memory integrity to prevent memory modification before the LMP function starts to run.

10.1.2. Features

Options

The corresponding function will be executed when the AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION option is activated in the initialization part.

Callback Message

If memory manipulation is detected by the corresponding function, the callback function of HackShield will send the AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP message.

Test Program

Provides Amazon.exe, a test program, implemented by the APIs provided by Ehsvc. Amazon.exe provides an existing HackShield test function and the Ehsvc test function.

10.1.3. System Architecture

Inputs the section data in the module to be protected by CSInspector.exe and provides HShield.lib and EHSvc.dll which are applied to the client.

Ehsvc.dll (Interface dll)

Provides an API that can set basic information for the monitoring server in case a hack attack or error occurs.

HShield.lib (HackShield Library)

Provides an API that can set basic information to notify a hack attack or error to the monitoring server.

CSInspector.exe (Protection Module Setting Utility)

Inputs section information to the client file or third-party module to be protected for HackShield to protect the code area of the corresponding module.

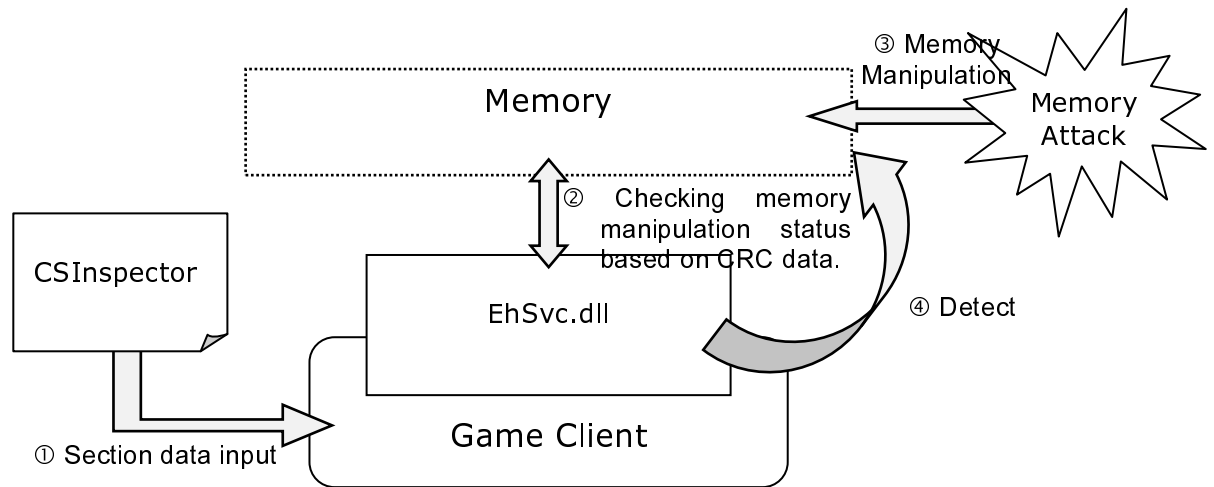


Figure 10-1 Operating Principles of Local Memory Protection

Operating principles of the Local Memory Protection function are as follows:

- ① Store the section data of the game client of DLL in a certain PE structure area using the CSInspector.exe² tool provided by HackShield.
- ② After HackShield is executed, load the section data to the PE structure of the game client or DLL, and check whether memory manipulation has occurred before execution.
- ③ Create CRC data of the current memory based on section information.
- ④ Regularly check memory manipulation status based on CRC information.

² CSInspector.exe is similar to HSBGen.exe, which extracts file/memory CRC by interfacing with the existing server and extracts the section data of the game client. However, unlike HSBGen.exe, CSInspector.exe does not store the file/memory CRC as a separate file but stores the section data of the game client in a certain space of the PE format in the game client. The code area data extracted by CSInspector.exe is not simple CRC, but provides detailed data about the game client such as the size of the starting address of the important part of the PE format section.

10.2.Application Programming

You can also use the LMP function through the HackShield initialization function (_AhnHS_Initonlize) provided by Ehsvc.

Note

The sample codes contained in this document are in the C/C++ language based on Microsoft Visual C++ 6.0. The programming language may change depending on the characteristics of each program and system environment.

10.2.1.Programming Application

Before using the LMP function, the developer must do the following.

10.2.1.1.LMP-related Files

LMP-related Files

Table 10-1 LMP-related File

위 치	File Name	Installation Folder	Description
[SDK]\Include\	HShield.h	[Program source folder]	Header file to be used by the client
[SDK]\Lib	HSheild.lib	[Program source folder]	Library file to be used by the client
[SDK]\Bin\Util	CSInspector.exe		Utility to set in the module subject to protection

10.2.1.2.Application

Client Application

1. Include HShield.lib file in the project.
2. Include the provided HShield.h file in the source file.
3. When calling _AhnHS_Initialize function, add the following option to the fifth parameter in the source.

```
// Define the option flag to call _AhnHS_Initialize function.
// Add to the existing option.)
dwOption = AHNHS_CHKOPT_ALL |
AHNHS_USE_LOG_FILE |
AHNHS_ALLOW_SVCHOST_OPENPROCESS |
AHNHS_ALLOW_LSASS_OPENPROCESS |
AHNHS_ALLOW_CSRSS_OPENPROCESS |
AHNHS_DONOT_TERMINATE_PROCESS |
AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION;

// Initialize HackShield service by calling
// _AhnHS_Initialize Function.
nRet = _AhnHS_Initialize ( szFullFilePath,
                          HS_CallbackProc,    //
                          CallbackFunction
                          1000,                // game code
                          "B228F291B7D7FAD361D7A4B7",
                          // License key
                          dwOption,
                          // option flag
                          AHNHS_SPEEDHACK_SENSING_RATIO_NORMAL
                          );
...

```

4. Add AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP in the event transmission function part in relation to HackShield.

```
int __stdcall HS_CallbackProc ( long lCode, long lParamSize,
void* pParam )
{
    TCHAR szMsg[MAX_PATH];

    // Display a corresponding error message.
    switch ( lCode )
    {
        // LMP Callback
        // Note: changed module name and the page address are returned
        // by LMP Callback but they don't need to be exposed to the user.
        case AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP:
            wsprintf(szMsg, "Memory
                                manipulation has been detected in the
                                module.\n" );
            MessageBox( NULL, szMsg, szTitle, MB_OK );
    }
}

```

```
break;  
.  
.  
.  
}
```

The following are sent as parameters:

- ICode
AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP(0x10705)
- pParam
"Manipulated module name (ModuleBaseaddress): Manipulated actual page address"
The name and the page address of the manipulated module are sent.
There is no need to show the information to the user.

Application of Protection Module

When distributing packed modules, apply CSInspector.exe first before applying the packer.

1. Use for the module which is designed to protect CSInspector.exe provided for the LMP function.
2. The CSInspector.exe tool operates as a command-line. Enter the following on the command-line input window:
C:\> CSInspector.exe Target.exe
3. If there is an additional DLL file to protect, apply CSInspector.exe in the same way.
C:\> CSInspector.exe Target.dll
4. After normal execution, SUCCESS message will be displayed.
5. Perform packing or CRC extraction after executing CSInspector.exe.

Note

For more information about CSInspector, refer to [11.9 CSInspector Tool \(For HackShield Pro, Version 5.1 and above\)](#)

Notes on Server Interfaces

When using old server interfaces and Extended Server Side Detection with the LMP, note the following:

1. Old Server Interfaces

- To apply the LMP function to the game client file, use CSInspector.exe before creating HackShield.crc with AntiCpSvrTool.exe.
- If old server interfaces are in use, apply the LMP function to the corresponding module and create HackShield.crc again with AntiCpSvrTool.exe.

2. Extended Server Side Detection

To apply the LMP function to the game client file, use CSInspector.exe first before creating anticpx.hsb with HSBGen.exe.

- If Extended Server Side Detection are in use, apply the LMP function to the corresponding module and create anticpx.hsb again with HSBGen.exe.

Supported Packers

Currently, the LMP supports the following packets.

Table 10-2 Packets Supported by LMP

Packer name	Version	Remarks
Themida	1.9.x	
Armadilo	5.x	
ASProtect	1.3x	

11.Tools

11.1.HackShield Cryptor Tool (For HackShield Pro)

11.1.1. What Is HackShield Cryptor?

HackShield Cryptor is designed to encrypt newly added executable files (modules) to HackShield Pro and important execution modules (game and HackShield). Encrypted files are decrypted when the game is executed.

11.1.2. Functions

HackShield Cryptor is used to prevent game module analysis and protect important modules during game execution. HackShield Cryptor is designed to help game developers easily encrypt files and make execution information lists of the encrypted files.

Deliverables of the HackShield Cryptor include encrypted executable files and execution information list files. The execution information list file (rdtlist.pfl) is an encrypted file and stores important information about protected files (modules). HackShield conducts real-time decryption based on this information when executing the game.

11.2.Using HackShield Cryptor Tool

11.2.1. Screen Description

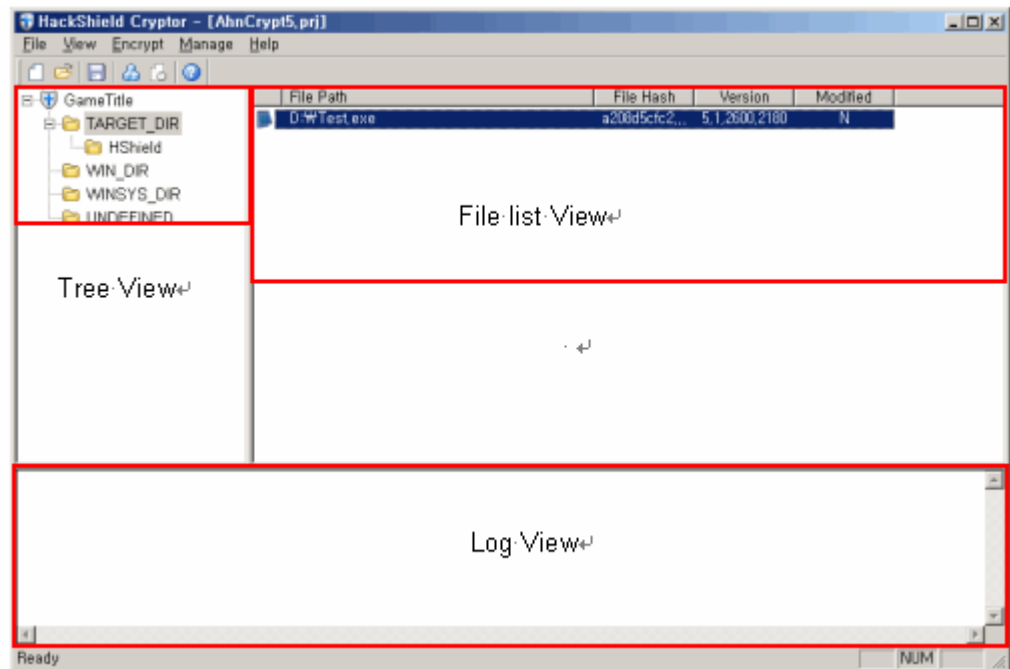


Figure 11-1 HackShield Cryptor

Tree View: Displays the folders where the game is installed, where files needed for the game are installed, and where HackShield is installed, as well the WINDOWS folder, WINSYS folder, and undefined folders.

File List: You can select files to perform encryption and encryption.

Log View: Displays logs during the build process.

11.2.2. Menu Description

Menus include File, View, Encrypt, Manage, and Help. The menus are described below.

11.2.2.1. File

New Project

Creates a new project. The shortcut key is **Ctrl + N**, and you can also select the corresponding icon on the toolbar.



Open

Opens an existing file. The shortcut key is **Ctrl + O**, and you can also select the corresponding icon on the toolbar.



Save

Saves the changes in the project. The shortcut key is **Ctrl + S**, and you can also select the corresponding icon on the toolbar.



Save As

Saves the current project as a new name.

Recent Files

Displays the list of recent project files. You can open the project by clicking on the corresponding project name.

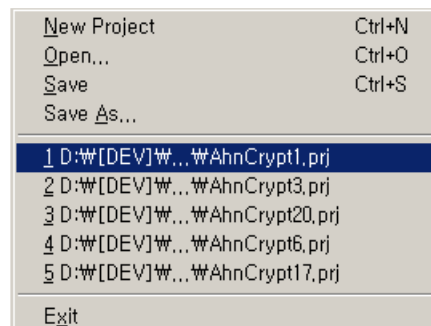


Figure 11-2 Recent Project File List

Exit

Exits the program.

11.2.2.2.View

Tools

Displays or hides the toolbar. If you select the Tools menu, the toolbar will be displayed, and if you deselect the Tools menu, the toolbar will disappear from the screen.

Status Bar

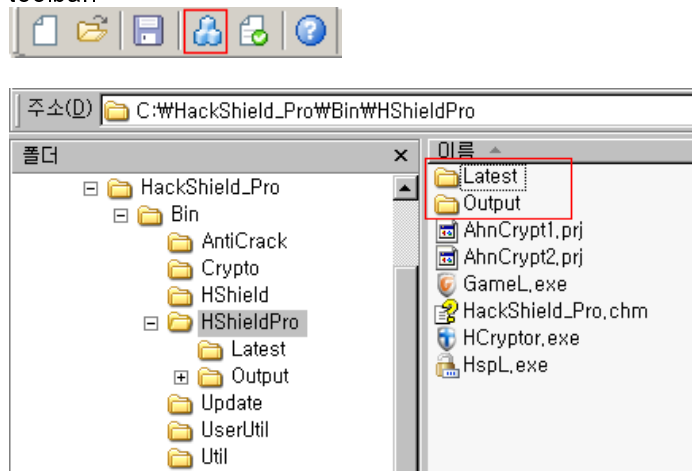
Displays or hides the status bar. If you select the Status Bar menu, the status bar will be displayed, and if you deselect the Status Bar menu, the status bar will disappear from the screen. The status bar provides a simple description of the selected menu.

11.2.2.3.Encrypt

Encrypts files added to the project and creates an execution information list file.

Build

Encrypts all added files. Stores the original file in the **Latest** folder and the backup file in the **Output** folder. The Latest folder and Output folder are created in the **HackShield Pro Installation Folder\Bin\HShieldPro** folder. The latest information files are stored in the Latest folder, and executable files are stored by version in the Output folder. You can also select the corresponding icon on the toolbar.



Caution

Add 'Ehsvc.dll' file to allow building. Otherwise, the following error will occur.



Figure 11-3 Ehsvc.dll File Error

If no error occurs in the build process, rtdlist.pfl file will be created which contains real-time encryption/decryption information. The rtdlist.pfl file is encrypted.

Verify

Verifies whether the file created as a result of the build process (execution information list file) can be used in HackShield Pro. The created file contains encrypted data about the files needed for real-time encryption/decryption. Decrypts the encrypted execution information list file, and compares it with the actual file for verification. If an error occurs, it verifies whether the files added to the list have been executed. It performs the build process again followed by verification. You can select either a menu or the corresponding icon on the toolbar.



11.2.2.4.Manage

Version Management

Sets and manages the project version.

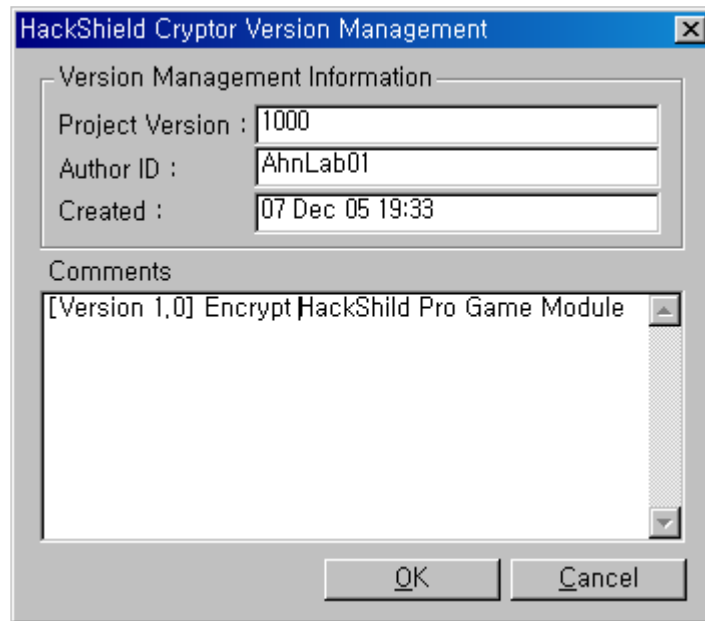


Figure 11-4 HackShield Cryptor Version Management

Project Version: The version increases by 1 for each build. You can change the version number. When the **Output** folder is created, the **gameName_VersionNumber** folder will be created.

Author ID: Enter the name of the person who created the current project version.

Created: Date of project modification. The system date will be automatically entered.

Comment: Enter the reason changing the project – who modified the encryption file and why. The record will provide useful information for encryption of other executable files.

11.2.2.5.Help

About HackShield Cryptor

Displays the version and copyright data of the HackShield Cryptor tool.



Figure 11-5 HackShield Cryptor Version

Help

Executes the Help file of the HackShield Cryptor tool. You can refer to the Help file for information related to the tool, and select the corresponding menu item or the corresponding icon on the toolbar.



11.2.3. Using Tree View

Renaming Game Folder

Right-click on the highest HackShield folder, and select **Rename**.

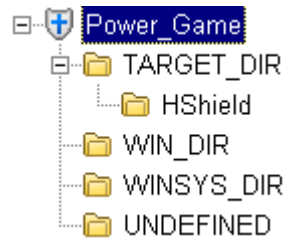


Figure 11-6 Renaming Game Folder

Caution

You cannot rename or delete the following folders:

TARGET_DIR, HShield, WIN_DIR, WINSYS_DIR, UNDEFINED.

Creating Folder

1. Right-click on a folder under which a folder will be created, and select **Create New Folder**.

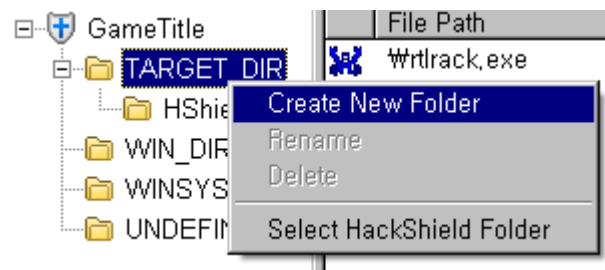


Figure 11-7 Creating Folder

2. Input the name of the folder to create.

Renaming Folder

1. Right-click on the folder to rename, and select **Rename**.

2. Enter the new name.

Deleting Folder

When a folder is deleted, all files stored under the folder will be also deleted.

Right-click on the folder to delete, and select **Delete**.

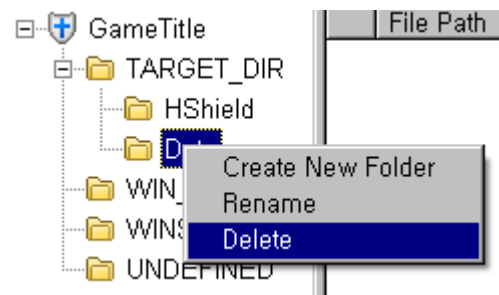


Figure 11-8 Deleting Folder

11.2.4. Using File List

The file list displays file path, file hash, version, and modified status of the file. You can add or delete file on the file list.

As shown below, the file list describes each file.




	File Path	File Hash	Version	Modified
	\\HShield\\poedit...	8f417e55af...	N/A	N
	\\HShield\\WEHSvc...	934def16e0...	3,3,41,0	N
	\\HShield\\AgntR...	0c200e59f9...	1,0,0,9	N

Figure 11-9 File List

File Path: Displays the full path of the added file.

File Hash: Hash data is used to check whether the added file was converted into hash data. If the hash data are different, **Y** will be displayed in the **Modified** field.

Version: Displays the file version. If no version is set in the file or the version is encrypted, N/A will be displayed. You can also check the file version on Windows Explorer. Right-click on the file to check the version, and select **Properties**. You can also check the version by clicking on the **Version** tab.

Modified: If the file data (hash value) has been changed, **Y** will be displayed under Modified.

Adding File: Right-click on the folder to add a file on the **tree view**, and select **Add**. Select a file to add from the **Open** window, and click on the **Open** menu. You can select multiple files.

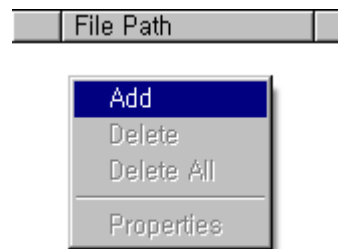


Figure 11-10 Adding File

Caution

Note that adding the same file to the same path will cause an error.

Deleting File

Right-click on the file to delete it from the list, and select **Delete**. You can delete multiple files at the same time by selecting them using the CTRL key. Right-click on the files to delete, and select **Delete**.

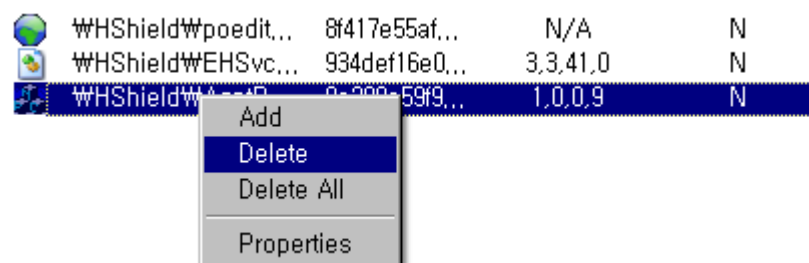


Figure 11-11 Deleting File

Deleting All Files

To delete all files on the file list, right-click and select **Delete All**.

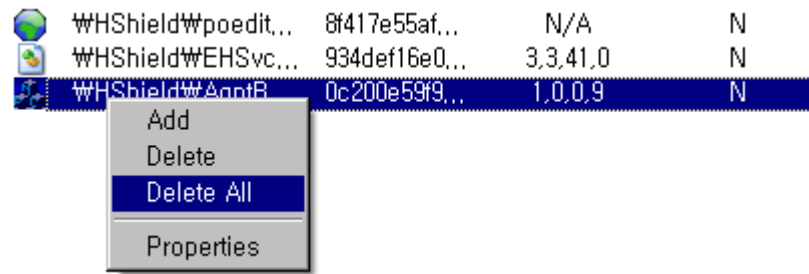


Figure 11-12 Deleting All Files

Viewing File Properties

Right-click on the file to view the properties, and select **Properties**. You can view the properties of the file in the same way as in Windows Explorer.

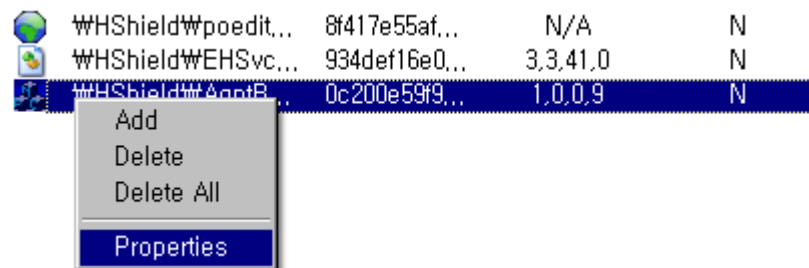


Figure 11-13 Viewing File Properties

11.2.5. Checking Log View

The log view displays errors and alarm messages that are created while using the HackShield Cryptor tool. You can check where the errors occur.

Error Display and Notification

Error or notification messages are displayed if project properties that are different from the configured information are displayed on the log view during file addition or deletion in project build or build verification.

Automatic Storing of Log Contents

After the build process, the contents displayed on the log view are stored as HspLog.txt files in the Latest folder and the Output folder in the same path as the Program Installation folder.

You can check which files have been added and built in the previous version by referring to the log file.

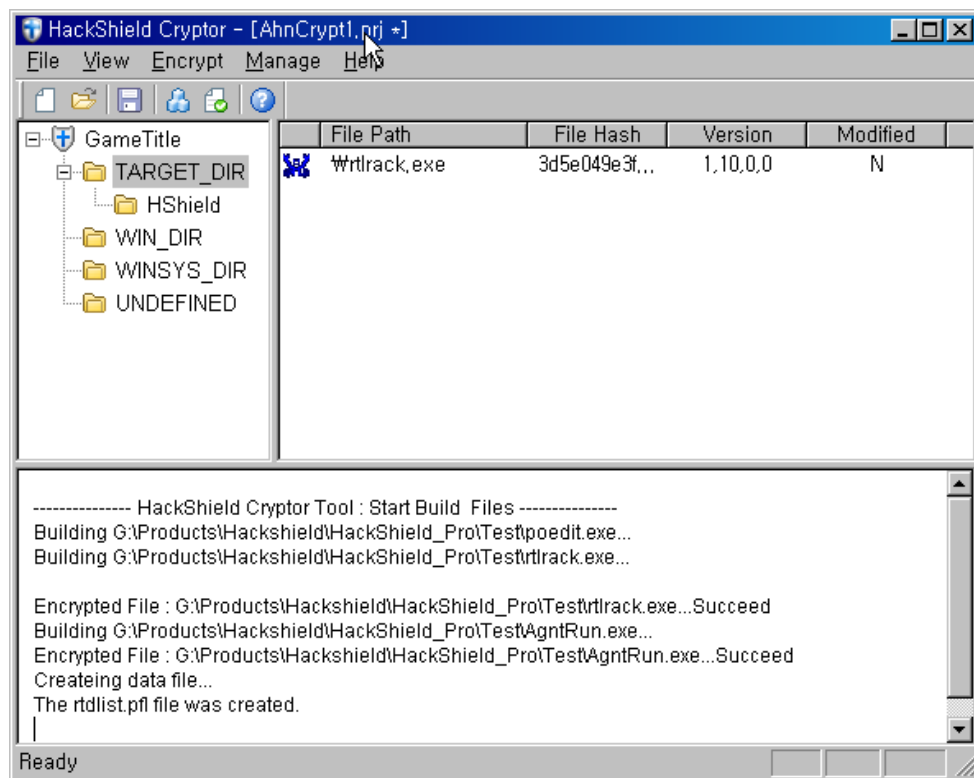
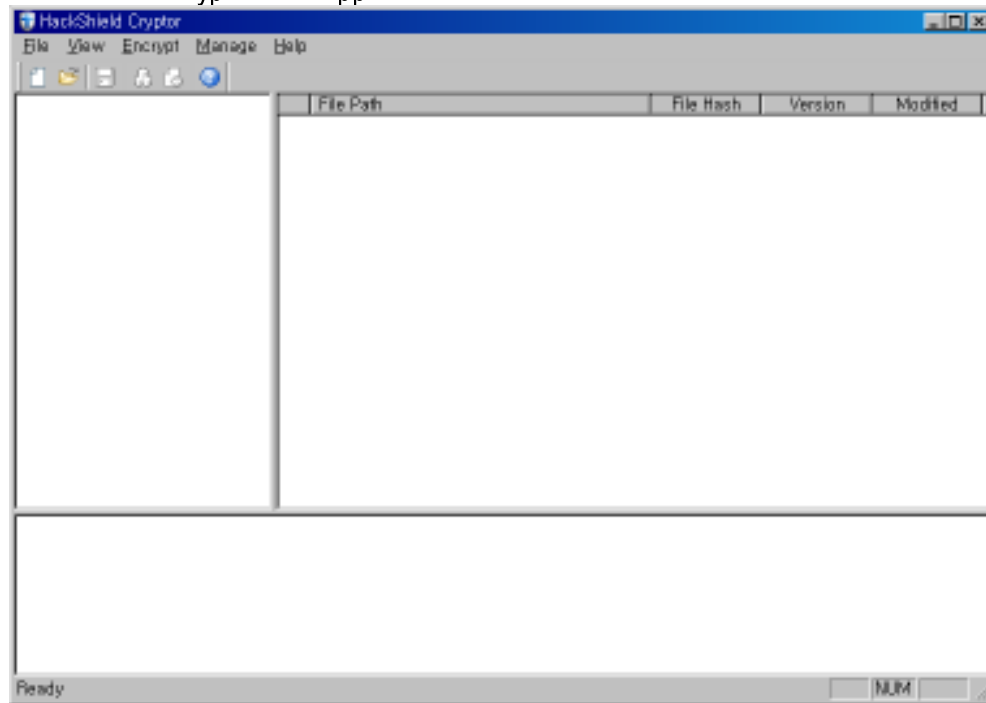


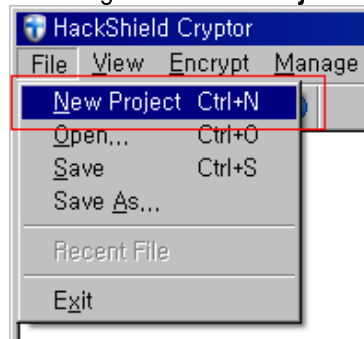
Figure 11-14 Log View

11.2.6. Encrypting Executable files Using HackShield Cryptor Tool

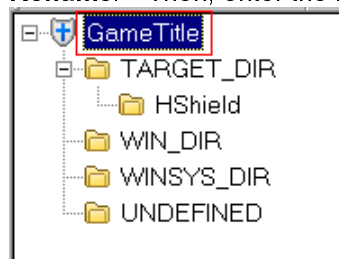
1. Double-click on **AhnLab HackShield Installation Folder\HCryptor.exe**. <HackShield Cryptor> will appear.



2. To encrypt the executable file using the HackShield Cryptor, create a project by selecting **File->New Project**.



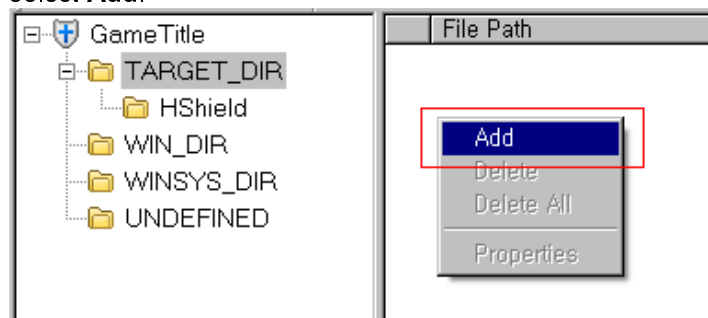
3. After a project is created, the tree view with the default being **GameTitle** will be created. To rename the tree view folder, right-click on **GameTitle** and select **Rename**. Then, enter the new name.



4. Select **TARGET_DIR** on the tree view.



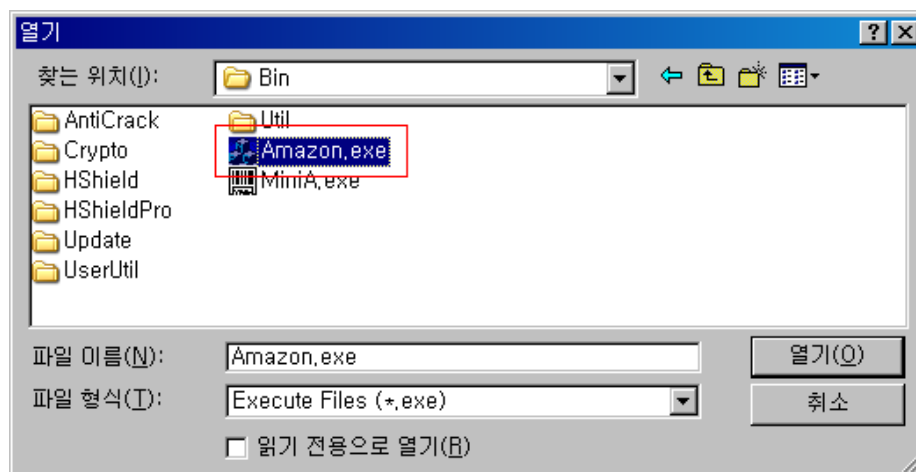
5. Add a game executable file to encrypt. To add an executable file, select **TARGET_DIR** on the tree view. Right click on the file list on the right side, and select **Add**.



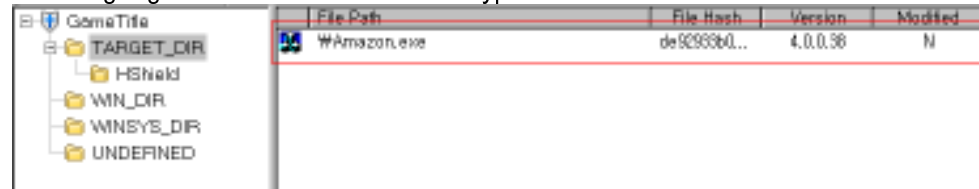
6. If you select the **Add** menu, the Add window will appear. Select the game executable file to encrypt, and click **Open**.

Note

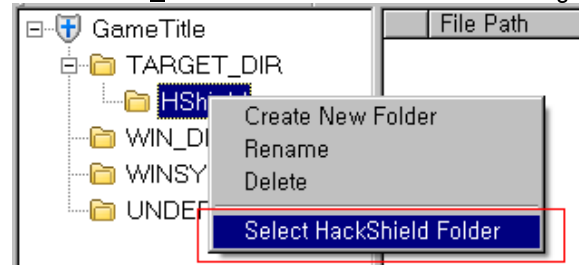
Amazon.exe was used as an example of the game executable file as the HackShield test program.



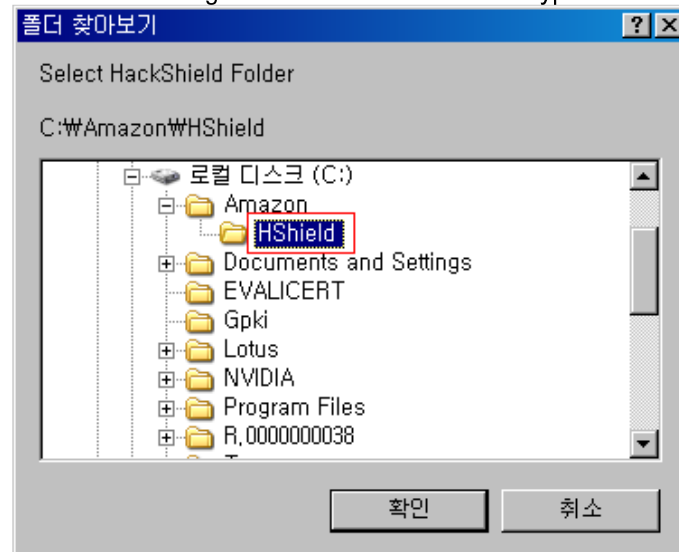
7. Check the file path, hash value, version, and the file modification status by selecting a game executable file to encrypt.



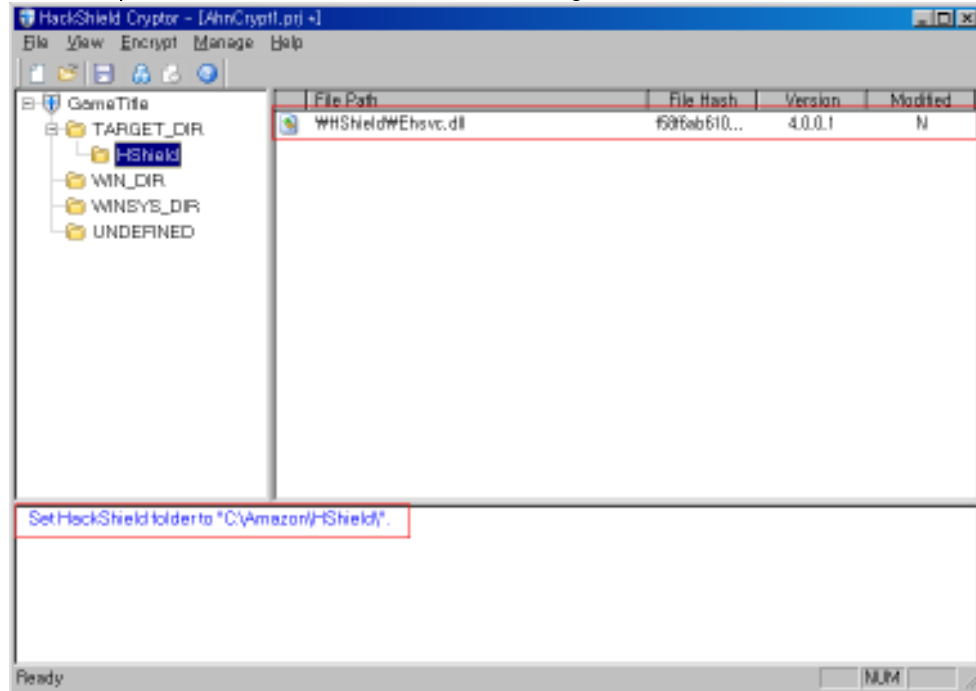
8. Include the HackShield folder in the project by right-clicking the **HShield** folder in **TARGET_DIR** on the tree view and selecting **Select HackShield Folder**.



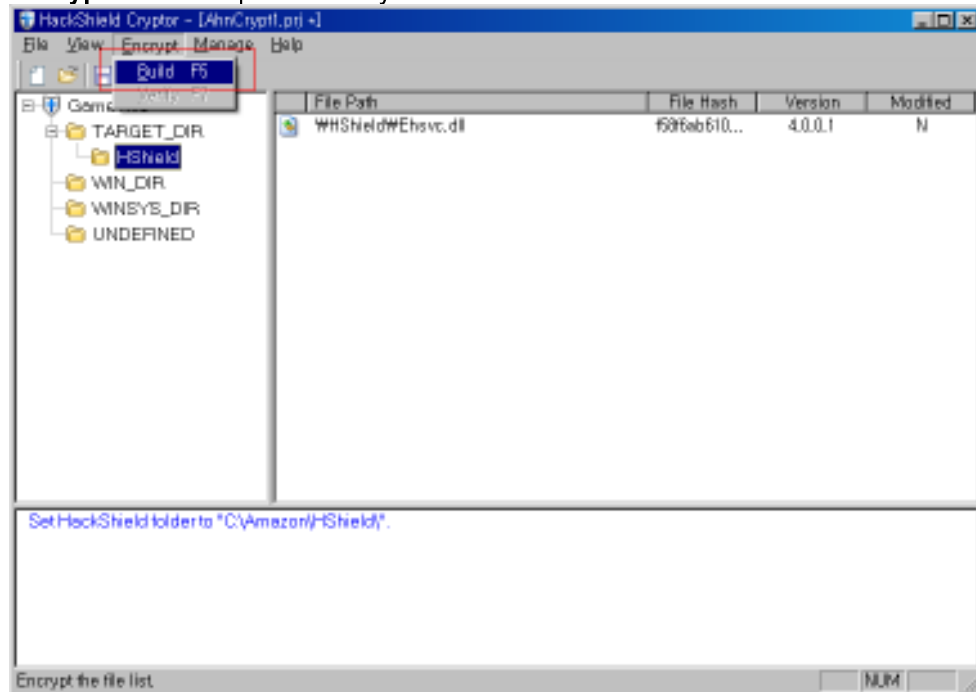
9. Select **Select HackShield Folder**. Then, the **Search Folder** window will appear. Select **HShield** folder where HackShield-related files are stored under the folder of the game executable file to encrypt.



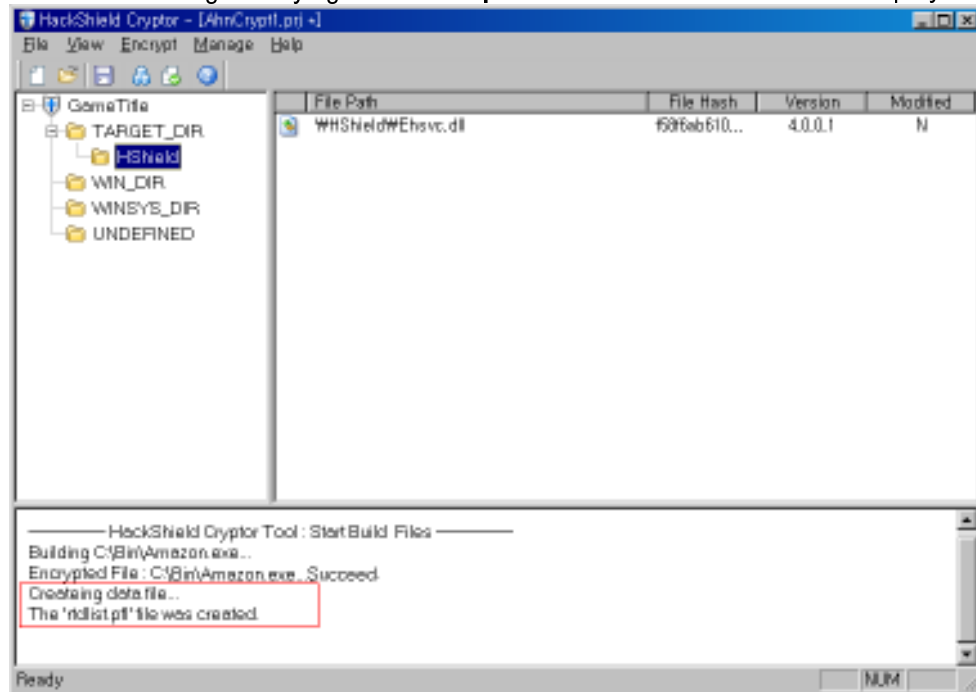
10. You can check the hash value of Ehsvc.dll, version, and file modification status of HackShield-related files by selecting HShield folder. You can also check the absolute path of the HackShield folder on the log view.



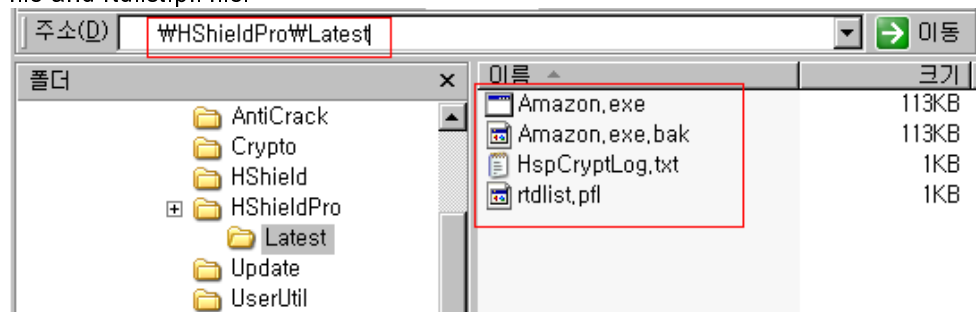
11. Include the game executable file to encrypt and HShield in the project, and encrypt the executable file. To encrypt the executable file, select **Encrypt->Build** or press **F5** key.



12. If you select **Build**, progress will be displayed on the log view in real time and the result message notifying that **rtdlist.pfl** file has been created will be displayed.



13. After the **rtdlist.pfl** file is created, the **Latest** folder will be created in the **HShieldPro** folder. The **Latest** folder will contain the encrypted game executable file and **rtdlist.pfl** file.



- **Encrypted game Executable file(Amazon.exe)**: Game executable file encrypted by HackShield Cryptor tool
- Original game executable file (Amazon.exe.bak): Original file before encryption by the HackShield Cryptor tool.
- HspCryptLog.txt: Stores the build results.
- **rtdlist.pfl**: Environment file created based on Amazon.exe and HackShield Pro-related file.

Caution

For game distribution, the encrypted game executable file (Amazon.exe) and **rtdlist.pfl** file must be in the same path.

11.2.7. Executing HCryptor.exe

HCryptor.exe specifies arguments needed for the result (Latest/Output) and provides a function to automatically create the result. The following describes the usage procedure.

Usage:>

HCryptor.exe PRJ File Path

Caution

Before automatic execution, manually execute HCryptor.exe and create PRJ environment file.

Example 1:>

When C:\HS_SDK\HShieldPro\Game.prj file is read for automatic build, the following will be executed.



```
C:\WINDOWS\system32\cmd.exe
C:\HS_SDK\HShieldPro>hcryptor c:\hs_sdk\HShieldPro\Game.prj
```

11.3. AntiCpSvr Tool_(For HackShield Pro)

11.3.1. Functions

You can create a CRC information file (HackShield.crc) in the server and check game files, memory status, and the integrity of HackShield. You can also control the previous version by patching a new version or setting options without restarting the server.

Different CRC information files are created each time. If there are multiple game servers, different CRC information files must be applied for stronger security. However, the same game client must have the same memory data.

11.4. AntiCpSvr Tool

11.4.1. Creating CRC Information File Based on Manual UI Setting

1. Enter the **CRC file path**. Click the [...] button and select a path under which the HackShield.crc file will be created. Then, the path of the selected file will be displayed.

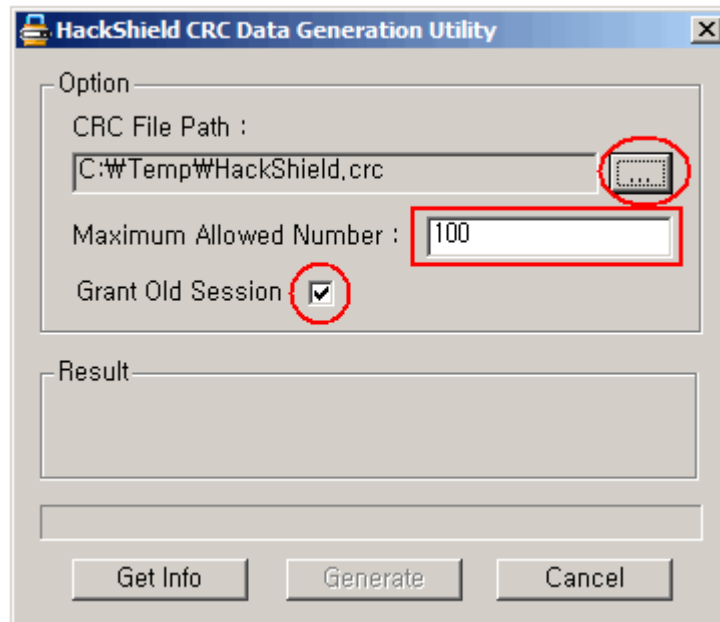


Figure 11-15 HackShield CRC Information Creation Tool

2. Enter the **Maximum Allowed Number**. Enter the number of the previous versions to allow when a newly created CRC file is applied to the server and a client tries to access the server. For example, if you enter 1, only clients with the latest version will be allowed to access. If you enter 2, only clients with the latest version and the previous version will be allowed to access the server.
3. Decide whether to select **Grant Old Session**. Decide whether to allow clients with a version not included in the maximum allowed number, to access a server in which a newly created CRC file is applied. You can allow the clients to access the server by selecting Grant Old Session, or deny access by previous-version clients by deselecting Grand Old Session.
4. Get information by selecting **GetInfo**.
5. Execute a game to patch in the game client until the `_AhnHS_SaveFuncAddress` function is called. Execute the game until the **Generate** button is activated.
6. If the **Generate** button is activated, terminate the game. After the game is completely terminated, wait for 2 or 3 seconds and create HackShield.crc by

selecting **Generate**. At this time, newly patched game data is stored in Ehsvc.dll, a HackShield module. Select **Generate** only after the game is completed.

After the HackShield.crc file is created, patch the CRC file to the server and patch the game client file which was used to create the CRC file and Ehsvc.dll file.

11.4.2. Automatically Creating CRC Information File

The AntiCpSvrTool specifies argument data for creating the CRC file and provides automatic CRC file creation function. The following describes the usage:

Usage:>

AntiCpSvrTool.exe **Game path to extract CRC data, Path of CRC file to create, Grant Old Session (1=true 0=false), Maximum Allowed Number**

Caution

Arguments are separated by ','.

Example:>

Extract CRC data of **Game.exe**, and set **Grant Old Session = true / Maximum Allowed Number = 5** to create **HackShield.crc** file. Then, the following will be executed.

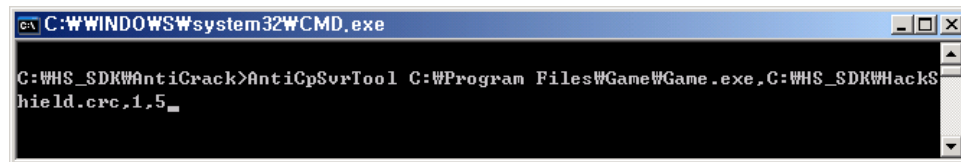


Figure 11-16 Command-line Type AntiCpSvrTool.exe

11.5. HSBGen Tool (For HackShield Pro 4.2 or higher versions)

11.5.1. Functions

You can check the game file, memory status, and integrity of HackShield by creating HackShield Briefcase File (AntiCpX.hsb) in the server. You can control the previous version by patching a new version or setting options without restarting the server.

Different HSB information files are created each time. If there are multiple game servers, different HSB information files must be applied for stronger security. However, the same game client must have the same memory data.

11.6. Using HSBGen Tool

11.6.1. Creating HSB Information File Based on Manual UI Settings

Create AntiCpX.hsb file using \Bin\AntiCrack\HSBGen.exe.

Caution

The original client executable file must be used before the packing process.

You can create the AntiCpX.hsb file using HSBGen.exe.

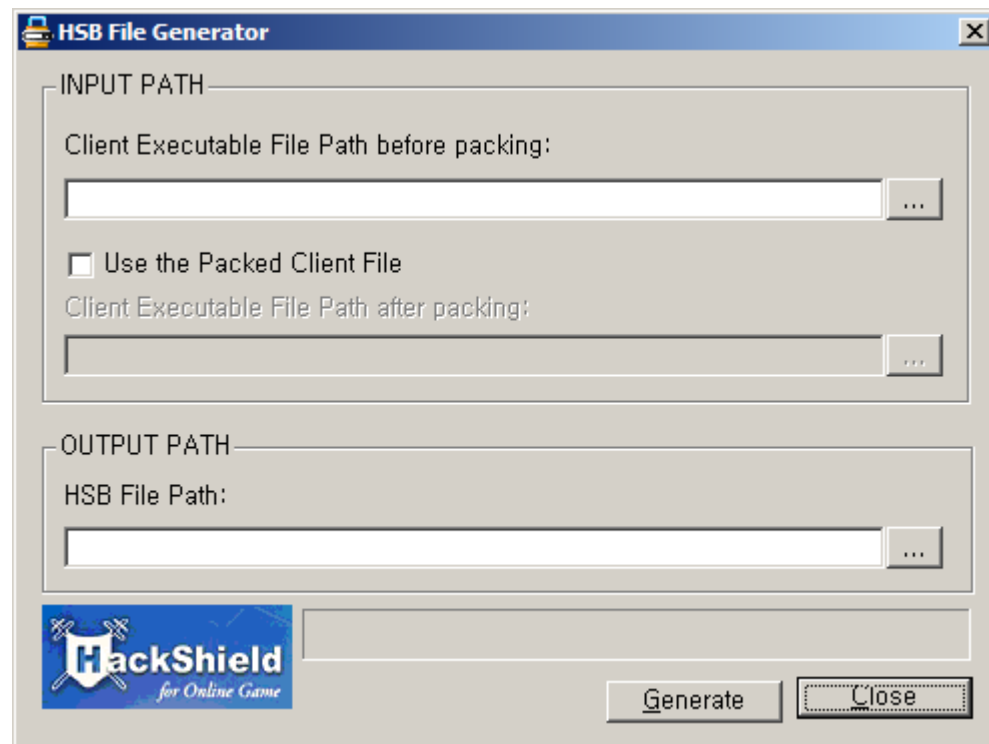


Figure 11-17 HSB File Generator

1. If AhnLab provides the **HSBGen.ini** file, the file must be stored in the same folder as **HSBGen.exe**. (Extended Server Side Detection must be set differently for each game.)
2. Set the path of the original executable file that was not yet packed in the **Client Executable File Path before packing** field. Select a file by clicking the [...] button.
3. To distributed the packed client executable file, enter the path of the unpacked file.
4. Select the **Use the Packed Client File** field, and enter the path of the packed file in the **Client Executable File Path after packing** field.

5. Set the full path of the **AntiCpX.hsb** file in the **HSB File Path** field. Click the [...] button to open the file creation window. (Note: If you set the same path as the executable file, an error message will be displayed which is normal.)
6. After completing all settings, click the **Generate** button. The progress state will be displayed and the result message will be displayed.

After the anticpx.hsb file is created, patch the anticpx.hsb file to the server and patch the game client file which was used to create the anticpx.hsb file and Ehsvc.dll file.

11.6.2. Automatically Creating HSB Information File

HSBGen specifies the argument data needed to create the HSB file and provides an automatic HSB file creation function. The following describes the usage:

Usage:>

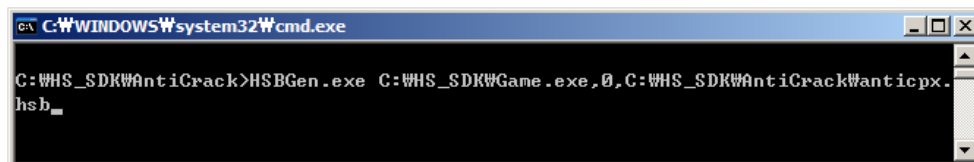
Game path to extract HSBGen.exe HSB data, packed executable file support status (1 = Supported / 0 = Not Supported X), packed game path to extract HSB data (used only when the packed executable file is supported), HSB file path

Caution

Arguments are separated by ','.

Example 1:>

If the game is not packed,
the execution path of the game is C:\HS_SDK\Game.exe,
the HackShield folder is C:\HS_SDK\HShield, and
the HSB file creation path is C:\HS_SDK\AntiCrack\game.hsb.



```
C:\WINDOWS\system32\cmd.exe
C:\HS_SDK\AntiCrack>HSBGen.exe C:\HS_SDK\Game.exe,0,C:\HS_SDK\AntiCrack\anticpx.hsb
```

Figure 11-18 Command-line Type HSBGen.exe (for General Files)

Example 2:>

If the game is not packed,
the execution path of the unpacked original game is C:\HS_SDK\Game.exe,
the execution path of the packed game is C:\HS_SDK\Game_packed.exe, and
HackShield folder will be C:\HS_SDK\HShield.
If the HSB file path is C:\HS_SDK\AntiCrack\game.hsb, execution will be made as follows:



```
C:\WINDOWS\system32\cmd.exe
C:\HS_SDK\AntiCrack>HSBGen.exe C:\HS_SDK\Game.exe,1,C:\HS_SDK\Game_packed.exe,C:\HS_SDK\AntiCrack\anticpx.hsb
```

Figure 11-19 Command-line Type HSBGen.exe (for Packed Files)

11.7.HSUpSetEnv Tool (For HackShield Pro 5.1 or higher versions)

11.7.1. Functions

Creates an update setting file (HSUpdate.env).
You can select FTP or HTTP and use multiple URLs.

11.8. Using HSUpSetEnv Tool

11.8.1. Creating HSUpdate.env File

Creates HSUpdate.env file using [HackShield SDK]\Bin\Util\HSUpSetEnv.exe. The following describes the usage.

Usage:>

Execute the [HackShield SDK]\Bin\Util\HSUpSetEnv.exe file.

Caution

HSUpSetEnv.exe is a tool that creates an update setting file. This tool must not be distributed.

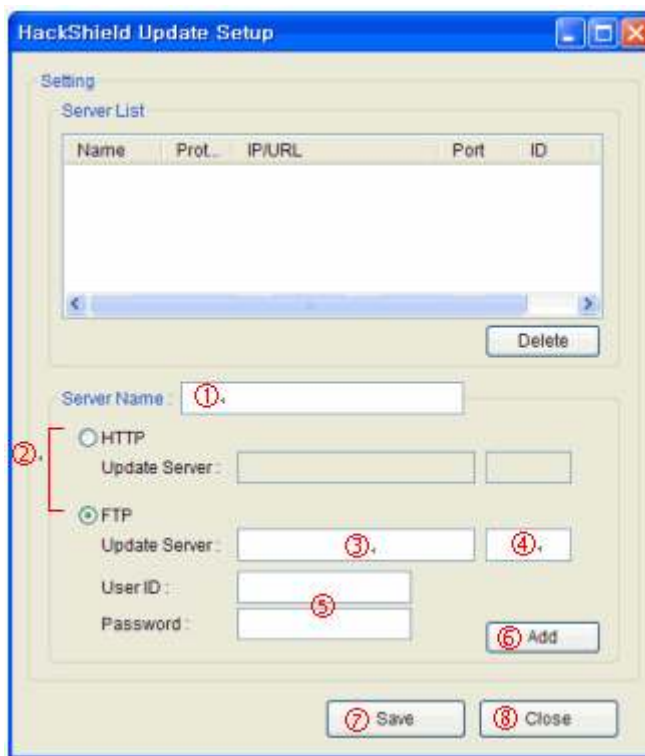


Figure 11-20 HSUpSetEnv.exe

- 1) Enter the update server name.
- 2) Select HTTP or FTP depending on the protocol the update server will use.
- 3) Input the server address. Specify the folder where the patch set is stored.
- 4) Enter the previously set port number.
- 5) Set account information for the FTP. Otherwise, an anonymous connection can be made.
- 6) Add the server to the server list by clicking on the Add button. If multiple update servers are needed, repeat the above procedures.

- 7) When you click the Save button, the HSUpdate.env file will be created with the following message displayed.



- 8) Terminate the tool after setup completion.

11.9. CSInspector Tool (For HackShield Pro 5.1 or higher versions)

11.9.1. Functions

Assigns certain EXE files or DLL files to be protected. CSInspector must be executed before the subject files are packed. When used with the server interface functions, CSInspector must be executed before AntiCpSvrTool.exe or HSBGen.exe. Currently, CSInspector is provided only as a command-line form.

CSInspector must be used with the AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION option in the client.

11.10. CSInspector Tool

11.10.1. Setting the File to Protect

Assigns a certain file to be protected using \Bin\Util\CSInspector.exe. EXE files and DLL files can be set. The following describes the usage procedure:

Usage:>
CSInspector.exe target file

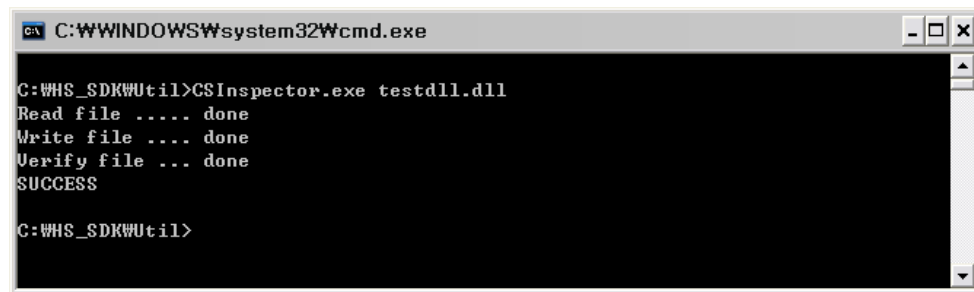
Caution

Must be conducted before packing or CRC creation.

Only statically loaded DLL files can be protected. (DLL files dynamically loaded by LoadLibrary are not supported.)

Example:>

If the target file is testdll.dll and CSInspector.exe is in the C:\HS_SDK\Util folder, do the following:

A screenshot of a Windows command prompt window. The title bar reads 'C:\WINDOWS\system32\cmd.exe'. The command prompt shows the following text:

```
C:\HS_SDK\Util>CSInspector.exe testdll.dll
Read file ..... done
Write file .... done
Verify file ... done
SUCCESS

C:\HS_SDK\Util>
```

Figure 11-21 CSInspector.exe

12. Appendix

12.1.FAQ

Can I control the anti-hacking driver at the kernel?

The anti-hacking driver of HackShield on the kernel keeps checking the operational status in EhSvc.dll. If an error occurs in the driver, it will be notified through the callback function.

SoftICE is installed in the system, but HackShield is not normalized.

How can I set the initialization options for development and debugging?

To debug the game client during the development or test process, use HShield.lib, Ehsvc.dll, and HShield.dat designed for developers.

(If the Ehsvc.dll and HShield.dat file are used with the release version, server interface errors may occur.)

- Path: \Developer\

Can a user logged in with user authority (not administrator authority) in Windows 2000 or Windows XP use HackShield?

HackShield runs the anti-hacking driver at the kernel level. Therefore, in the default setting, only users with administrator authority can run HackShield. To use HackShield's game hacking prevention function with a general user account, a shadow account for HackShield must be created.

During the game service, HackShield's stopping function (_AhnHS_StopService, _AhnHS_Uninitialize) was not called and the game program was improperly terminated. Can I start HackShield again by starting the game program again?

If the game program is terminated without calling the HackShield stopping function, the anti-hacking driver will not be unloaded in the system. If the game program is restarted by the HackShield initialization function, the existing driver will be forcibly unloaded and the new driver will be loaded for normal initialization.

If a new hacking tool is detected, how can I block it?

If a new hacking tool is detected, following information will be sent to AhnLab. AhnLab analyzes the hacking tools and the results are reflected in the weekly engine updates or emergency engine.

- Games with hacking tools running
- OS version
- Simple description of the hacking tool
- Hacking tool executable file

12.2.Index

—	
_AhnHsp_StartGame	110, 113
_AhnHsp_StartLauncher	108
_AhnHS_HSUpdate	143
_AhnHS_MakeAckMsg	74
_AhnHS_MakeGuidAckMsg	72, 97
_AhnHS_PauseService	42
_AhnHS_ResumeService	44
_AhnHS_SaveFuncAddress	76
_AhnHS_Uninitialize	30
_AhnHsp_StartGame	106, 107
_AhnHsp_StartLauncher	105
_AhnHsUserUtil_CreateUser	130, 132
_AhnHsUserUtil_SetFolderPermission	131, 134
_AntiCpSvr_Initialize	56, 84, 150
_AntiCpSvr_AnalyzeAckMsg	68, 92
_AntiCpSvr_AnalyzeGuidAckMsg	61, 88
_AntiCpSvr_Finalize	58, 86, 152
_AntiCpSvr_MakeGuidReqMsg	59, 87
_AntiCpSvr_MakeReqMsg	65, 89
_HsCrypt_FRead	119, 124
_HsCrypt_GetDecMsg	118, 123
_HsCrypt_GetEncMsg	118, 122
_HsCrypt_InitCrypt	117, 120

A

ACTAPCPARAM_DETECT_AUTOMOUSE	47
ACTAPCPARAM_DETECT_HOOKFUNCTION	46, 47
AHNHS	
_SPEEDHACK_SENSING_RATIO_HIGH	33
AHNHS	
_SPEEDHACK_SENSING_RATIO_HIGH	33
AHNHS	
_SPEEDHACK_SENSING_RATIO_LOW	33
AHNHS	
_SPEEDHACK_SENSING_RATIO_LOW	33
AHNHS	
_SPEEDHACK_SENSING_RATIO_NO	33
AhnHS_StopService	40
AhnHS_Uninitialize	41
AHNHS_ACTAPC_DETECT_ALREADYHOOKED	46

AHNHS_ACTAPC_DETECT_AUTOMOUSE	48
AHNHS_ACTAPC_DETECT_AUTOMOUSE	47
AHNHS_ACTAPC_DETECT_DRIVERFAILED	48
AHNHS_ACTAPC_DETECT_HOOKFUNCTION	47
AHNHS_ACTAPC_DETECT_KDTRACE	48, 49
AHNHS_ACTAPC_DETECT_KDTRACE	49
AHNHS_ACTAPC_DETECT_MESSAGEHOOK	48
AHNHS_ACTAPC_DETECT_SPEEDHACK	48
AHNHS_ALLOW_CSRSS_OPENPROCESS	32
AHNHS_ALLOW_LSASS_OPENPROCESS	32
AHNHS_ALLOW_SVCHOST_OPENPROCESS	32
AHNHS_CHKOPT_ALL	26, 32
AHNHS_CHKOPT_AUTOMOUSE	32
AHNHS_CHKOPT_KDTARCE	32
AHNHS_CHKOPT_MESSAGEHOOK	32
AHNHS_CHKOPT_OPENPROCESS	32
AHNHS_CHKOPT_PROCESSSCAN	32
AHNHS_CHKOPT_READWRITEPROCESS	32
AHNHS_CHKOPT_SPEEDHACK	32
AHNHS_DISPLAY_HACKSHIELD_LOG	33
AHNHS_DONOT_TERMINATE_PROCESS	33
AHNHS_ENGINE_DETECT_GAME_HACK	46
AhnHS_Initialize	31
AhnHS_StartService	37
AHNHS_USE_LOG_FILE	27, 32
AHNLAB_DEFINED_ERROR_CODE	69, 97
AntiCPSvr.dll	51, 79, 147, 154
AntiCpSvrTool.exe	51, 79

D

Data Encryption Program	114
-------------------------------	-----

E

Eagle9x.vxd or EagleNT.sys	13
EhSvc.dll	128

EHSvc.dll.....	13
ERROR_ANTICPXSVR_CLIENT_FILE_	
ATTACK (0xE904000B)	94
ERROR_ANTICPXSVR_HSHIELD_FILE	
_ATTACK (0xE904000A)	94
ERROR_ANTICPXSVR_INVALID_PARA	
METER (0xE9040003)	90, 92
ERROR_ANTICPXSVR_MEMORY_ATT	
ACK (0xE904000C)	94
ERROR_ANTICPXSVR_NOT_YET_REC	
EIVED_RESPONSE (0xE9040005) 90,	93
ERROR_ANTICPXSVR_OLD_VERSION	
_CLIENT_EXPIRED (0xE904000D) .	94
ERROR_ANTICPXSVR_UNKNOWN_CL	
IENT (0xE904000E)	94
ERROR_HSCRYPTLIB_EXCEPTION126	
ERROR_HSCRYPTLIB_FREAD_DECRY	
PT_FREAD	125
ERROR_HSCRYPTLIB_FREAD_DECRY	
PT_GETDECMMSG	126
ERROR_HSCRYPTLIB_FREAD_DECRY	
PT_RANGE	125
ERROR_HSCRYPTLIB_FREAD_FSEEK	
.....	125
ERROR_HSCRYPTLIB_FREAD_GETFI	
LELEN	124
ERROR_HSCRYPTLIB_FREAD_GETPO	
SITION	125
ERROR_HSCRYPTLIB_FREAD_INVALI	
DPARAM	124
ERROR_HSCRYPTLIB_FREAD_SIZEZE	
RO	125
ERROR_HSCRYPTLIB_GETDECMMSG	
NVALIDPARAM	123
ERROR_HSCRYPTLIB_GETENCMMSG	
NVALIDPARAM	122
ERROR_HSCRYPTLIB_INITCRYPT_IN	
VALIDPARAM	121
ERROR_STARTGAME_ENCRYPT_FAIL	
.....	112
ERROR_STARTGAME_INITCRYPT_FAI	
L	112
ERROR_STARTGAME_INVALIDPARAM	
.....	111
ERROR_STARTGAME_MAKEFILEKEY_	
FAIL	112
ERROR_STARTGAME_SHAREMEMINI	
T_FAIL	112
ERROR_STARTGAME_WAITEVENT_F	
AIL	112
ERROR_STARTLAUNCHER_CREATEP	
ROCESS	109
ERROR_STARTLAUNCHER_INVALIDP	
ARAM	108, 110
ERROR_STARTLAUNCHER_MAKEME	
MMAPNAME_FAIL	108

ERROR_STARTLAUNCHER_SHAREME	
INIT_FAIL	109
ERROR_STARTLAUNCHER_WAITEVE	
NT_FAIL	109
ERROR_SUCCESS56, 59, 61, 66, 68, 76,	120, 122, 123, 124, 150
Executable file Encryption Program...	102
Extended Server Interface (AntiCpX)...	78

H

HackShield Cryptor	101, 160
HS_ERR_ALREADY_INITIALIZED	35
HS_ERR_COMPATIBILITY_MODE_RU	
NNING	34
HS_ERR_DEBUGGER_DETECT	35
HS_ERR_DRV_FILE_CREATE_FAILED	
.....	38, 39
HS_ERR_INIT_DRV_FAILED	35
HS_ERR_INVALID_FILES	34
HS_ERR_INVALID_LICENSE	34
HS_ERR_INVALID_PARAM	34, 43
HS_ERR_NEED_ADMIN_RIGHTS	35
HS_ERR_NOT_INITIALIZED37, 40, 41,	42, 44
HS_ERR_OK	34, 37, 42, 44
HS_ERR_REG_DRV_FILE_FAILED ...	38
HS_ERR_START_DRV_FAILED	39
HS_ERR_START_ENGINE_FAILED ..	38
HS_ERR_UNKNOWN	36
HsCryptLib	117
HsCryptLib.lib	115
HsCryptoUtil 프로그램	115
HSERROR_ENVFILE_NOTREAD	144
HSERROR_ENVFILE_NOTWRITE ...	144
HSERROR_LIB_NOTEDIT_REG	144
HSERROR_NETWORK_CONNECT_FAI	
L	144
HSERROR_NOTFINDFILE	145
HSERROR_PROTECT_LISTLOAD_FAIL	
.....	145
HShield.lib	51, 79, 138, 147, 154
HspLauncher	101
HspLauncherLib	105
HSUpdate.exe	13
HsUserUtil	130
HsUserUtil Data	128
HsUserUtil.lib	128
HSUSERUTIL_ERR_ADDNEWACE_FAI	
L	135
HSUSERUTIL_ERR_ADDNEWACETON	
EWDACL_FAIL	135
HSUSERUTIL_ERR_ADDNEWDACL_F	
AIL	135
HSUSERUTIL_ERR_ADDSHADOWACN	
T_FAIL	133
HSUSERUTIL_ERR_DELHIDEIDINFO_F	

AIL 133
 HSUSERUTIL_ERR_DELSHADOWACN
 T_FAIL 133
 HSUSERUTIL_ERR_DELSHADOWACN
 TINFO_FAIL 133
 HSUSERUTIL_ERR_GETGROUPSID_F
 AIL 135
 HSUSERUTIL_ERR_GETSECINFO_FAI
 L 135
 HSUSERUTIL_ERR_NOT_ADMIN... 132,
 134
 HSUSERUTIL_ERR_NOT_NT 132

HSUSERUTIL_ERR_OK..... 132, 134
 HSUSERUTIL_ERR_SETFLDRPERMIS
 SION_FAIL 134

P

PFN_AhnHS_Callback..... 46

V

V3Pro32s.dll 13