

AhnLab HackShield

프로그래밍 가이드

Version 5.2

AhnLab HackShield for Online Game

시작하기 전에

Copyright (C) AhnLab, Inc. 1988-2009. All rights reserved.

이 프로그래밍 가이드의 내용과 프로그램은 저작권법과 컴퓨터 프로그램 보호법에 의해 보호받고 있습니다.

문서 개요

이 문서는 SDK 형태로 제공되는 AhnLab HackShield for Online Game, AhnLab HackShield for Online Game(이하 HackShield)의 전체 구조와 기능 및 API 함수 사용법을 설명한 프로그래밍 가이드입니다.

고객 지원

㈜안철수연구소 고객만족센터	
주소	150-869 서울시 영등포구 여의도동 12번지 CCMM빌딩 6층 (주)안철수연구소
홈페이지	http://www.ahnlab.com
메일 주소	hs@ahnlab.com
전화	02-2186-3082 (기업고객전용 핫라인) 02-2186-3000 (고객만족센터)
팩스	02-2186-6100 (대표) 02-2186-3001 (고객 지원 센터)

목차

1. HackShield 소개	9
1.1. 기능	9
1.2. 필요한 시스템 환경	10
2. 기본 기능	11
2.1. 개요	11
기능	11
특징	12
시스템 구조(System Architecture)	13
2.2. Application Programming	15
프로그래밍 순서	15
2.3. 프로그래밍 준비	17
2.3.1. HackShield 관련 파일	17
2.3.2. 적용	18
핵셀드 모니터링 함수 작성	18
핵셀드 업데이트 함수 작성	18
핵셀드 라이선스 키 발급받기	18
핵셀드 초기화 함수 작성	19
핵셀드 서비스 시작 함수 작성	21
핵셀드 Callback 함수 작성	22
핵셀드 서비스 정지 함수 작성	23
핵셀드 서비스 종료 함수 작성	23
2.4. Application Programming Interface	25
_AhnHS_Initialize	25
_AhnHS_Callback	34
_AhnHS_StartService	38
_AhnHS_StopService	42
_AhnHS_Uninitialize	43
_AhnHS_PauseService	44
_AhnHS_ResumeService	46
3. 핵셀드 업데이트 기능	48
3.1. 개요	48
기능	48
특징	48
시스템 구조(System Architecture)	49
3.2. Application Programming	50
프로그래밍 적용 방법	50
3.3. Application Programming Interface	58
_AhnHS_HSUpdateEx	58
_AhnHS_HSUpdate	62
4. 확장 서버 연동 크랙 방지 기능	66
4.1. 개요	66
기능	66
특징	67
시스템 구조(System Architecture)	67
4.2. Application Programming	71
프로그래밍 적용 방법	71

4.3. Application Programming Interface	74
_AhnHS_CreateServerObject	74
_AhnHS_CloseServerHandle	76
_AhnHS_CreateClientObject	77
_AhnHS_CloseClientHandle	79
_AhnHS_MakeRequest	80
_AhnHS_VerifyResponseEx	83
_AhnHS_VerifyResponse	85
_AhnHS_MakeResponse	91
5. 서버 연동 크랙 방지 기능	95
5.1. 개요	95
기능	95
특징	96
시스템 구조(System Architecture)	96
5.2. Application Programming	99
프로그래밍 적용 방법	99
5.3. Application Programming Interface	102
_AntiCpSvr_Initialize	102
_AntiCpSvr_Finalize	104
_AntiCpSvr_MakeGuidReqMsg	105
_AntiCpSvr_AnalyzeGuidAckMsg	107
_AntiCpSvr_MakeReqMsg	112
_AntiCpSvr_AnalyzeAckMsg	115
_AhnHS_MakeGuidAckMsg	119
_AhnHS_MakeAckMsg	122
_AhnHS_SaveFuncAddress	124
6. 모니터링 서비스 기능	126
6.1. 개요	126
기능	126
특징	126
시스템 구조(System Architecture)	127
6.2. Application Programming	128
프로그래밍 적용 방법	128
6.3. Application Programming Interface	130
_AhnHS_StartMonitor	130
_AhnHS_SetUserId	132
7. LMP 기능	133
7.1. 개요	133
기능	133
특징	134
시스템 구조(System Architecture)	134
7.2. Application Programming	135
프로그래밍 적용 방법	136
_AhnHS_IsModuleSecure	138
_AhnHS_IsModuleSecure	139
8. 부가 기능	141
8.1. 데이터 파일/메시지 암호화 기능	141
8.1.1. 개요	141
기능	141
특징	141

시스템 구조(System Architecture)	141
8.1.2. Application Programming.....	143
프로그래밍 순서.....	143
프로그래밍 준비.....	143
HsCryptLib 파일.....	143
_HsCrypt_InitCrypt.....	144
_HsCrypt_GetEncMsg.....	144
_HsCrypt_GetDecMsg.....	145
_HsCrypt_FRead.....	145
8.1.3. Application Programming Interface	147
_HsCrypt_InitCrypt.....	147
_HsCrypt_GetEncMsg.....	149
_HsCrypt_GetDecMsg.....	150
_HsCrypt_FRead.....	151
8.2. User 권한 실행 지원 기능	153
8.2.1. 개요.....	153
기능.....	153
특징.....	153
시스템 구조(System Architecture)	154
8.2.2. Application Programming.....	155
프로그래밍 순서.....	155
프로그래밍 준비.....	156
_AhnHsUserUtil_CreateUser.....	156
_AhnHsUserUtil_SetFolderPermission.....	157
_AhnHsUserUtil_DeleteUser.....	157
_AhnHsUserUtil_IsEnableHSAAdminRights.....	158
_AhnHsUserUtil_CheckHSShadowAccount.....	158
_AhnHsUserUtil_IsAdmin.....	158
8.2.3. Application Programming Interface	159
_AhnHsUserUtil_CreateUser.....	159
_AhnHsUserUtil_SetFolderPermission	161
_AhnHsUserUtil_DeleteUser.....	163
_AhnHsUserUtil_IsEnableHSAAdminRights	164
_AhnHsUserUtil_CheckHSShadowAccount.....	165
_AhnHsUserUtil_IsAdmin	166
9. 툴 사용 방법	167
9.1. AntiCpSvr 툴	167
기능.....	167
9.2. AntiCpSvr 툴 사용 방법	168
UI 수동 설정 기반의 CRC 정보 파일 생성.....	168
자동 CRC 정보 파일 생성.....	169
9.3. HSBGen 툴(HackShield 전용, 4.2 이후 버전).....	170
기능.....	170
9.4. HSBGen 툴 사용 방법	171
UI 수동 설정 기반의 HSB 정보 파일 생성.....	171
자동 HSB 정보 파일 생성.....	173
HSBGen.ini 설명.....	174
9.5. HSUpSetEnv 툴(HackShield 전용, 5.1 이후 버전).....	179
기능.....	179
9.6. HSUpSetEnv 툴 사용 방법	180
HSUpdate.env 파일 생성.....	180
9.7. CSInspector 툴.....	182

기능.....	182
대상 파일을 보호 대상으로 지정.....	182
9.8. SetServerList 툴(HackShield 전용, 5.1 이후 버전).....	183
기능.....	183
9.9. SetServerList 툴 사용 방법.....	184
afs.dat 파일 생성.....	184
afs.dat 파일 배포.....	185
9.10. HSBHelper 툴 사용 방법	186
기능.....	186
10. 부록.....	188
10.1. FAQ.....	188
10.2. 색인.....	192
10.3. 변경 사항.....	195

표 목차

표 2-1 HackShield 관련 파일	17
표 2-2 게임 자체 업데이트 서버에 설치되어야 하는 파일 목록	18
표 2-3 [AHNHS_CHKOPT_SELF_DESTRUCTION 적용시 처리 콜백 코드].....	28
표 3-1 업데이트 관련 파일	51
표 3-2 업데이트 서버에 설치되어야 하는 파일 목록.....	51
표 3-3 핵셴드 업데이트 이후 자동 생성 파일.....	52
표 4-1 서버연동 버전관리	68
표 4-2 AntiCpXSvr 관련 파일.....	71
표 5-1 서버연동 버전 관리	97
표 5-2 AntiCPSvr 관련 파일.....	99
표 6-1 모니터링 관련 파일	128
표 7-1 LMP 관련 파일	136
표 7-2 LMP 지원 패커	140
표 8-1 HsCryptLib 파일.....	143
표 8-2 HsUserUtil 파일	156

그림 목차

그림 2-1 HackShield 구조도	13
그림 2-2 어플리케이션 프로그램 순서	15
그림 3-1 핵셴드 업데이트 동작원리	50
그림 3-2 디폴트 핵셴드 업데이트 이미지	56
그림 4-1 AntiCpX의 동작 원리	69
그림 5-1 AntiCpSvr의 동작원리	98
그림 6-1 모니터링 서비스의 동작원리	127
그림 7-1 Local Memory Protection 동작 구조	135
그림 8-1 HsCryptLib의 전체 구조 및 동작 원리	142
그림 8-2 HsUserUtil의 전체 구조 및 동작 원리	154
그림 8-3 HsUserUtil 프로그래밍 순서	155
그림 9-1 HackShield CRC 정보 생성 툴	168
그림 9-2 Command-line 방식의 AntiCpSvrTool.exe	169
그림 9-3 HSB File Generator	171
그림 9-4 Command-line 방식의 HSBGen.exe (일반 파일의 경우)	173
그림 9-5 Command-line 방식의 HSBGen.exe (패킹된 파일의 경우)	174
그림 9-6 Command-line 방식의 HSBGen.exe(패킹된 파일을 실행 하는 경우)	174
그림 9-7 Command-line 방식의 HSBGen.exe(패킹된 파일을 실행 시 파라미터가 필요한 경우)	174
그림 9-8 HSUpSetEnv.exe	180
그림 9-9 CSInspector.exe	182
그림 9-10 SetServerList.exe	184
그림 9-11 SetServerList 툴의 주소 입력 창	185
그림 9-12 HSBHelper 툴	186

1. HackShield 소개

HackShield는 (주)안철수연구소에서 개발한 게임 프로그램 해킹 툴 탐지, 해킹 차단, 크랙 방지, 실행 파일 실시간 보호, 데이터 암호화 등의 기능을 제공하는 솔루션입니다.

1.1. 기능

해킹 차단 기능

시그니처 및 메모리 휴리스틱 기반의 해킹 툴 탐지 기능, 메모리 해킹 차단 기능, 스피드 핵 차단 기능, 디버깅 차단, 메시지 후킹 차단, 파일 위/변조 차단, 오토 마우스 차단 등의 기능을 제공합니다.

서버 연동 크랙 방지 기능

서버와 연동하여 실시간 실행 파일 조작 감지, 메모리 조작 감지, 핵설드 동작 상태 확인 등의 기능을 제공합니다.

데이터 파일/메시지 암호화 기능

주요 데이터 파일과 서버, 클라이언트간에 주고 받는 메시지를 암호화하여 데이터 파일이나 메시지가 노출되어도 암호화되어 있어서, 내용을 확인할 수 없도록 안전하게 보호하는 기능을 제공합니다.

User 권한 실행 기능

NT 계열에서 Administrator 권한이 아닌 일반 User 권한이나 Guest 권한으로 실행할 수 있는 기능을 제공합니다.

1.2. 필요한 시스템 환경

HackShield를 설치하거나 동작하는데 필요한 시스템 환경입니다.

Client Side

지원 항목	권장 사양	최소 사양
운영 체제	Windows 98/ME/2000 Professional/XP Home /XP Professional/Server 2003/Vista	Windows 98 이상
CPU	Intel Pentium 500Mhz 이상	Intel Pentium 133MHz 이상 IBM-PC 호환기종
RAM	128MB 이상	32MB
HDD	2MB 이상	2MB

Server Side (구 서버연동 & 확장 서버연동)

OS	Platform
Windows 2K, 2K3	x86
Solaris 8, 10 (32bit)	x86
Fedora Linux 7.1 (32bit)	x86

주의

웹 서버, DB 서버 등 서버 구동용으로 사용하는 Windows NT 계열에서 HackShield를 실행할 경우, 예상하지 못한 성능 저하 등의 문제가 발생할 수 있습니다.

참고

CPU는 Intel x86 계열(x86호환 AMD 계열포함)만 지원합니다. Windows NT를 실행하는 Alpha chip machine과 Windows를 실행하는 NEC pc 8xxx machine 운영 체제에 대해서는 지원하지 않습니다.

2. 기본 기능

HackShield에서 제공하는 해킹 차단 및 해킹 툴 탐지 기능을 구현하는데 필요한 시스템 구조와 프로그래밍 방법을 설명합니다.

참고

이 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

2.1. 개요

기능

엔진기반 (시그니처 & 휴리스틱) 의 해킹 툴 탐지 기능

(주)안철수연구소의 안티 바이러스 기술을 이용하여, 시그니처가 엔진에 등록되어 있는 해킹 툴을 탐지합니다. 해킹 툴이 엔진에 등록되어 있으면 해당 프로세스를 강제로 종료시키고(옵션에 따라 강제로 종료시키지 않을 수도 있습니다), **Callback** 함수로 해당 파일 이름을 게임 클라이언트에게 알려줍니다. 엔진에 등록되어 있지 않은 해킹 툴이 발견되면 새로운 해킹 툴에 대한 시그니처가 추가된 엔진으로 업데이트하여 해킹 툴을 차단할 수 있습니다.

메모리 해킹 차단 기능

게임 프로그램에서 사용하는 Windows API(**OpenProcess**, **Read/WriteProcessMemory...**)를 통한 메모리 접근을 차단합니다. 커널 레벨에서 직접 메모리를 추적하여, 결과 값을 조작하는 등의 해킹을 차단합니다.

주의

해킹 툴은 아니지만 메모리에 직접 접근하는 프로그램의 경우 정상적으로 동작하지 않을 수 있습니다.

스피드 핵 차단 기능

스피드 핵은 크게 시스템에 장착되어 있는 타이머를 조작하는 하드웨어 방식과 운영 체제의 시간 관련 API를 조작하는 소프트웨어 방식이 있습니다. 하드웨어나 소프트웨어 방식으로 실행되는 스피드 핵을 차단하기 위해서는 마이크로 프로세서 레벨에서 시스템의 시간과 운영 체제에서 만들어내는 논리적인 시간의 차이를 수시로 확인합니다. 시스템의 시간과 운영 체제의 시간이 일정 시간 이상 차이가 나면 이를 **Callback** 함수로 게임

클라이언트에게 알려줍니다.

사용자의 시스템이나 운영 체제, 그리고 게임의 성격에 따라 스피드핵 감지에 대한 정도의 차이가 있으므로 별도의 파라미터를 통해서 스피드핵 감지율에 대한 레벨을 5단계로 설정할 수 있습니다.

참고

스피드 핵이란 타이머 처리 마이크로 프로세서나 **Windows** 시스템에서 제공하는 시간 관련 함수를 조작하여 시간 관련 기능이 정상 동작하지 못하도록 하는 프로그램입니다.

디버깅 차단

디버거를 이용하여 게임 프로그램을 분석하고 해킹을 시도할 수 없도록 모든 디버거의 디버거 트레이싱을 차단합니다. 디버거 트레이싱 시도가 감지되면 **Callback** 함수를 통해서 게임 클라이언트에게 알려줍니다. **HackShield**를 초기화할 때에도 **SoftICE**와 같은 디버깅 프로그램이 실행중인지 여부를 먼저 확인하고, 디버깅 프로그램이 실행되고 있으면 에러를 리턴합니다.

오토 마우스 차단

마우스 입력을 자동으로 처리해서 게임 프로그램을 조작하고 게임 서버에 과 부하를 발생시킬 수 없도록 오토 마우스를 차단합니다.

특징

인터페이스 함수(API) 제공

HackShield에서 제공하는 기능을 편리하게 사용하고, 그 실행 결과 값을 받아 볼 수 있도록 인터페이스 **DLL**을 제공합니다. 제공되는 인터페이스 **DLL**을 사용하여 게임 개발사는 고유 정책에 따라서 **HackShield**의 해킹 차단 기능 중 필요한 기능만 실행하도록 프로그래밍할 수 있습니다.

샘플 프로그램 제공

HackShield에서 제공하는 **API**를 사용하여 구현한 테스트용 게임 클라이언트 프로그램을 제공합니다. 게임 개발사는 샘플 프로그램을 참고하여 **HackShield**의 기능 및 성능을 확인하고, 클라이언트 프로그램을 개발하는데 참고할 수 있습니다.

핵실드 업데이트 제공

핵실드 업데이트 기능을 사용하여 해킹 차단 및 해킹 툴 탐지 엔진 등을 수시로 제공합니다. 게임 클라이언트는 지정된 업데이트 서버를 통하여 최신 해킹 차단 및 해킹 툴 탐지 엔진을 다운로드 받을 수 있습니다. 업데이트 서버는 **FTP** 및 **HTTP**를 모두 지원하며, **FTP**의 경우 **anonymous** 로그인과 사용자 로그인을 모두 지원합니다. 업데이트를 진행할 때, 옵션을 선택하면 업데이트 진행 상황을 윈도우로 출력해서 확인할 수 있습니다.

시스템 구조(System Architecture)

HackShield는 독립된 실행 파일 형태가 아닌 SDK를 이용한 라이브러리 형태로 제공합니다. HackShield의 전체 구조 및 동작 원리는 다음과 같습니다.

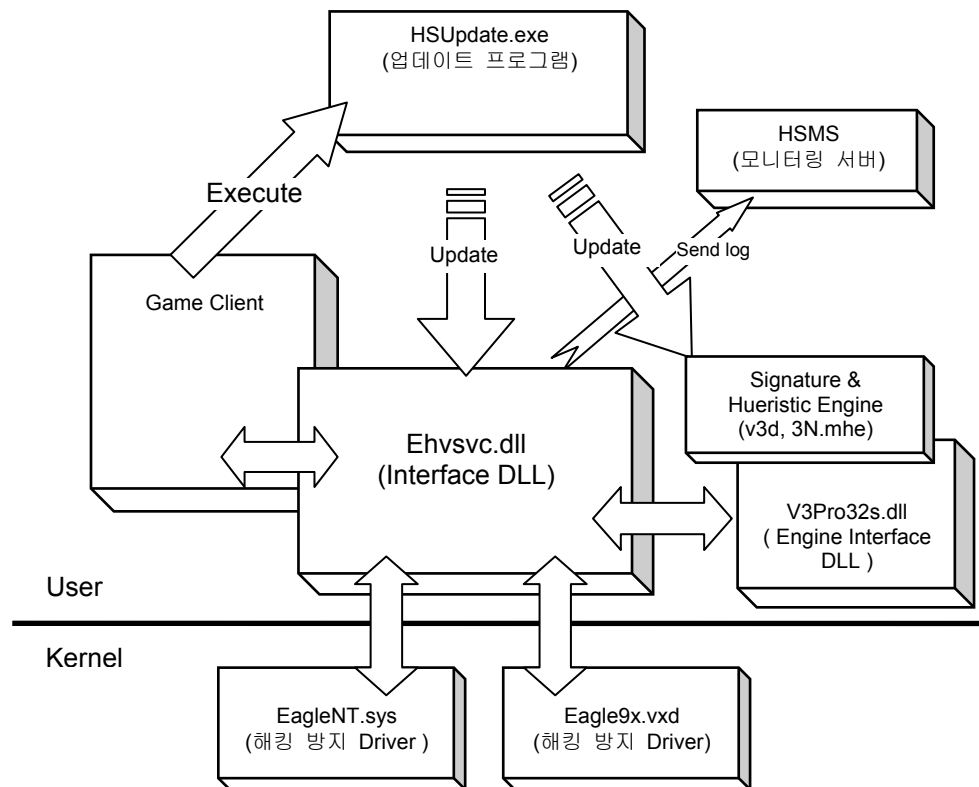


그림 2-1 HackShield 구조도

HSUpdate.exe(업데이트 프로그램)

HackShield 전용 업데이트 프로그램으로서 업데이트 서버로부터 해킹 차단 모듈과 해킹 툴 탐지 모듈을 포함한 엔진 파일을 다운로드받아서 업데이트합니다.

EHSvc.dll(인터페이스 DLL)

인터페이스 파일로서 해킹 차단 모듈과 해킹 툴 탐지 엔진을 동작시킵니다. 해킹 툴 탐지 및 해킹 차단 결과를 Callback 함수를 통하여 게임 클라이언트에게 알려줍니다.

V3Pro32s.dll(엔진 인터페이스 DLL)

(주)안철수연구소의 안티 바이러스 기술을 이용한 해킹 툴 탐지 엔진을 동작시킵니다. 엔진 시그니처 파일에 등록되어 있는 해킹 툴이 실행되면, 즉시

이를 탐지하여 게임 프로세스에게 알려줍니다. 엔진 시그니처 파일에 등록되어 있지 않은 해킹 툴이 발견되면, 우선 엔진 시그니처 파일에 등록시켜 즉시 차단하고 새로운 해킹 툴의 동작 방식을 분석한 엔진 시그니처 파일로 엔진을 업데이트합니다.

Eagle9x.vxd or EagleNT.sys(해킹 차단 드라이버)

커널 모드에서 동작하는 드라이버 파일로서 해킹 차단 기능을 수행합니다. EHSvc.dll(인터페이스 DLL) 파일에 포함되어 있으며, 실행되는 운영 체제에 따라서 Eagle9x.vxd 또는 EagleNT.sys가 시스템에 로드됩니다.

2.2. Application Programming

HackShield에서 제공하는 API를 이용해서 해킹 차단 기능 및 해킹 툴 탐지 기능을 구현하는 방법을 설명합니다.

참고

본 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

프로그래밍 순서

게임 클라이언트 프로그램 개발자는 다음과 같은 순서로 해킹 툴 탐지 및 해킹 차단 기능을 구현합니다.

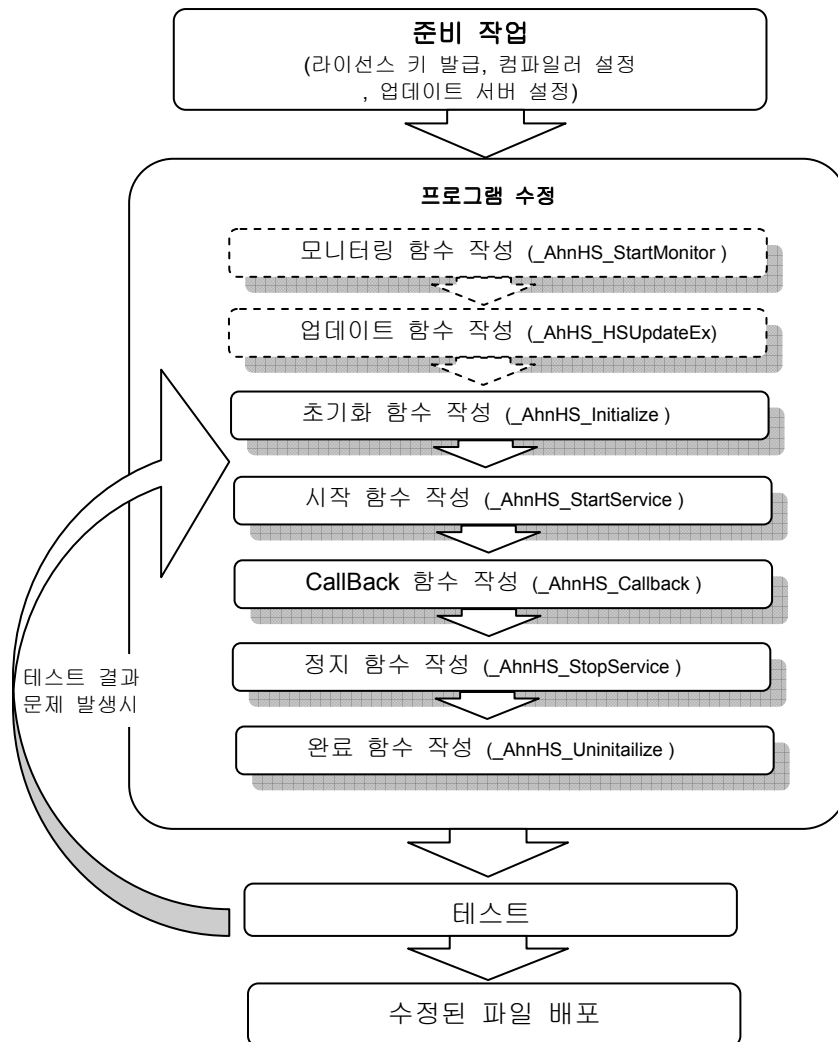


그림 2-2 어플리케이션 프로그램 순서

1. **준비 작업:** 제공 받은 HackShield 파일 목록을 확인하고, 필요한 파일을 복사합니다. 프로그래밍을 하기 위해서는 반드시 라이선스 키를 발급 받아야 합니다.
2. **핵실드 모니터링 함수(AhnHS_StartMonitor) 작성:** 핵실드 모니터링 서버 (HSMS) 가 설치된 경우, 서버에 해킹 및 오류 정보를 전송할 수 있도록 함수를 작성합니다.
3. **핵실드 업데이트 함수(AhHS_HSUpdateEx) 작성:** 핵실드 업데이트를 사용하는 경우, 핵실드 모듈이 자동으로 업데이트 될 수 있도록 핵실드 업데이트 함수를 작성합니다.

참고

HackShield의 업데이트 프로그램을 이용하여 업데이트를 하는 경우에는 별도로 제공되는 업데이트 라이브러리(HSUpChk.lib)를 사용해서 업데이트 함수를 호출합니다.

4. **핵실드 초기화 함수(AhnHS_Initialize) 작성:** HackShield의 초기화 함수 호출 코드를 작성하여 파일 위/변조 검사 및 데이터 초기화를 합니다.
5. **핵실드 서비스 시작 함수(AhnHS_StartService) 작성:** 서비스 시작 함수 호출 코드를 작성하여 해킹 차단 및 해킹 툴을 탐지합니다.
6. **핵실드 Callback 함수(AhnHS_Callback) 작성:** 해킹 차단 기능 및 해킹 툴 탐지 기능 실행 결과를 처리하는 Callback 함수를 작성합니다.
7. **핵실드 서비스 중지 함수(AhnHS_StopService) 작성:** 프로그램에 대한 종료 처리 부분에 서비스 중지 함수 호출을 추가함으로써 해킹 차단 기능 및 해킹 툴 탐지 기능을 정지시킵니다.
8. **핵실드 완료 함수(AhnHS_Uninitialize) 작성:** 서비스 중지 함수 호출 다음에 완료 함수를 호출하도록 코드를 작성합니다.

주의

게임이 비정상적으로 종료되는 경우에도, 핵실드의 종료루틴이 반드시 실행되어야 합니다.

비정상 종료 상황에 대비하여 게임의 최초 시작 위치에 아래의 코드를 추가하면 예외로 인한 게임 종료 시 핵실드 종료 루틴을 수행할 수 있습니다.

```
void Game_UnhandledExceptionHandler ()
{
    _AhnHS_StopService();
    _AhnHS_Uninitialize();
}

::SetUnhandledExceptionFilter ( Game_UnhandledExceptionHandler );
```


9. 작성된 소스 코드가 정상적으로 동작하는지 테스트합니다.
10. 게임 클라이언트를 사용자에게 배포합니다.

2.3. 프로그래밍 준비

HackShield를 사용하여 프로그래밍을 시작하기 전에 설치 디렉터리에서 확인해야 하는 HackShield 파일 목록과 컴파일러 설정하는 방법, 업데이트 모듈 사용하는 방법, 업데이트 서버 설정하는 방법, 라이선스 키 발급하는 방법을 설명합니다.

2.3.1. HackShield 관련 파일

설치 디렉터리

게임 개발사는 HackShield 파일들을 게임 클라이언트가 설치되어 있는 폴더의 하위에 HackShield 폴더를 별도로 생성하여 설치합니다.

<Game Directory>\HShield\

HackShield 관련 파일

표 2-1 HackShield 관련 파일

파일 이름	설치 폴더	설명
3N.mhe	[HackShield 폴더]	휴리스틱 엔진 파일
EhSvc.dll	[HackShield 폴더]	인터페이스 DLL
HShield.dat	[HackShield 폴더]	핵실드 관련 dat 파일
psapi.dll	[HackShield 폴더]	Process Status Helper DLL
v3warpds.v3d	[HackShield 폴더]	해킹 차단 엔진 패턴 파일
v3warpns.v3d	[HackShield 폴더]	해킹 차단 엔진 패턴 파일
v3pro32s.dll	[HackShield 폴더]	해킹 톨 탐지 엔진 인터페이스 DLL
HShield.h	[게임 소스 폴더]	헤더 파일
HShield.lib	[게임 소스 폴더]	라이브러리 파일

컴파일러 설정

HackShield를 사용하는 게임 클라이언트 프로그램의 프로젝트 파일에는 HShield.lib 파일을 라이브러리나 소스 코드 목록에 포함해야 합니다.

2.3.2. 적용

핵심드 모니터링 함수 작성

모니터링 서버는 AhnLab_HSMS_Install_Guide 와 AhnLab_HSMS_Operator_Guide 를 통하여 설치 및 운영할 수 있도록 합니다.

클라이언트 부분은 [4. 핵심드 모니터링 서비스 기능](#) 참고하여 모니터링 기능을 적용 합니다.

핵심드 업데이트 함수 작성

[Case1. 핵심드 업데이트 모듈 사용]

[3. 핵심드 업데이트 기능](#) 참고하여 업데이트 서버 설정 및, 클라이언트 적용을 합니다.

클라이언트 적용시 HackShield의 인터페이스 DLL인 EhSvc.dll 파일이 로드되기 전에 업데이트 라이브러리 함수를 호출해서 핵심드 업데이트를 실행해야 합니다.

[Case2. 게임 개발사 자체 패치 모듈 사용]

게임 개발사 자체 패치 모듈을 사용해서 HackShield 관련 파일을 업데이트할 경우에는 별도의 업데이트 모듈을 배포할 필요는 없습니다. 기존 게임 패치 모듈과 함께 HackShield 관련 파일들만 포함시켜 배포해야 합니다.

게임사 자체 패치 서버를 사용할 경우 핵심드 관련 패치 파일은 다음과 같습니다.

표 2-2 게임 자체 업데이트 서버에 설치되어야 하는 파일 목록

파일 이름	설명
3N.mhe	휴리스틱 엔진 파일
Ehsvc.dll	HackShield 인터페이스 DLL
HShield.dat	핵심드 관련 dat 파일
psapi.dll	Process Status Helper DLL
V3pro32s.dll	해킹 툴 탐지 엔진 인터페이스 DLL
V3warpds.v3d	해킹 툴 패턴 엔진 파일
V3warpns.v3d	해킹 툴 패턴 엔진 파일

핵심드 라이선스 키 발급받기

핵심드를 적용하려면 반드시 안철수연구소로부터 부여받은 라이선스가 필요합니다.

라이선스 키를 발급받는 방법은 다음과 같습니다.

1. EhSvc.dll을 사용하는 실행 파일의 이름, 퍼블리셔 이름(지역), 게임 개발사 이름, 게임 프로그램 이름을 ㈜안철수연구소에 전달하여 라이선스 키 발급을 요청합니다.

2. 고유의 게임 코드(4자리 숫자)와 24자리 문자열로 구성된 라이선스 키가 발급 완료됩니다.
3. 발급 받은 게임 코드와 라이선스 키 값을 초기화 함수 `_AhnHS_Initialize`의 파라미터로 전달합니다.

주의

초기화 함수 `_AhnHS_Initialize`의 파라미터로 잘못된 값을 전달하면, 초기화 함수가 정상적으로 호출되지 못하고 에러(`HS_ERR_INVALID_LICENSE`)를 리턴합니다.

핵심드 초기화 함수 작성

프로그래밍을 위한 준비 작업이 끝났으면 가장 먼저 초기화 함수 `_AhnHS_Initialize`를 호출합니다. 초기화 함수 호출이 성공적으로 완료되어야 `HackShield`의 해킹 차단 기능 및 해킹 툴 탐지 기능을 실행할 수 있습니다.

초기화 함수는 게임 클라이언트 프로그램의 인스턴스를 초기화하거나 메인 윈도우를 초기화하는 루틴에서 호출합니다.

초기화 함수를 호출한 예제는 다음과 같으며, 아래 예제에서는 초기화 함수의 마지막 파라미터에 모든 옵션을 사용하도록 `AHNHS_CHKOPT_ALL`을 사용하였습니다.

(옵션 중에는 선택하지 않아도 자동으로 적용되는 코어 옵션¹이 있습니다. 코어 옵션은 사용자가 적용하지 않아도 내부적으로 자동적으로 적용됩니다.)

예제

```

BOOL HS_Init()
{
    int          nRet = 0;
    TCHAR szFullFilePath[MAX_PATH];
    TCHAR szMsg[MAX_PATH];
    DWORD dwOption = 0;

    // ① 핵심드 폴더의 EhSvc.dll 위치를 지정합니다.
    lstrcat ( szFullFilePath, _T( "\\HShield\\EhSvc.dll" ) );

    // ② _AhnHS_Initialize 함수 호출에 쓰일 옵션 플래그를 정의합니다
    dwOption = AHNHS_CHKOPT_ALL;

    // ③ _AhnHS_Initialize 함수를 호출하여 핵심드 서비스를 초기화 합니다.

    nRet = _AhnHS_Initialize ( szFullFilePath,
                               HS_CallbackProc, // 콜백함수

```

¹ 코어 옵션: `AHNHS_CKOPT_SPPEDHACK` | `AHNHS_CHKOPT_READWRITEPROCESSMEMORY` | `AHNHS_CHKOPT_KDTRACER` | `AHNHS_CHKOPT_OPENPROCESS` | `AHNHS_CHKOPT_AUTOMOUSE` | `AHNHS_CHKOPT_PROCESSCAN`

```

1000, // 게임 코드
"B228F291B7D7FAD361D7A4B7", // 라이선스 키
dwOption, // 옵션 플래그
AHNHS_SPEEDHACK_SENSING_RATIO_NORMAL );

// ④ _AhnHS_Initialize 함수의 리턴 값을 검사하여 에러 처리합니다.
if ( nRet != HS_ERR_OK )
{
    switch( nRet )
    {
        case HS_ERR_COMPATIBILITY_MODE_RUNNING:
        case HS_ERR_NEED_ADMIN_RIGHTS:
        case HS_ERR_INVALID_FILES:
        case HS_ERR_INIT_DRV_FAILED:
        case HS_ERR_DEBUGGER_DETECT:
        case HS_ERR_NOT_INITIALIZED:
        default:
            wsprintf( szMsg, "해킹방지 기능에 문제가 발생하였습니다. (%x)", nRet );
            break;
    }
    MessageBox( NULL, szMsg, szTitle, MB_OK );
    return FALSE;
}
return TRUE;
}

```

주의

초기화 함수 _AhnHS_Initialize는 한 프로세스당 한 번만 호출할 수 있습니다. 초기화 함수를 중복 호출하면 오류가 발생할 수 있습니다.

EhSvc.dll 파일이 위/변조되거나 버전이 맞지 않을 경우에는 초기화 함수를 호출하면 에러 값(HS_ERR_INVALID_FILES)을 리턴합니다.

참고

HackShield 인터페이스 DLL 파일 EhSvc.dll이 아닌 게임 클라이언트 프로그램에서 HackShield 함수를 호출하는 부분이 위/변조될 가능성도 있습니다. 따라서 게임 클라이언트 프로그램에서도 파일 위/변조를 차단할 수 있도록 제공된 Packer 프로그램을 사용하여 게임 클라이언트 파일을 암호화, 압축하여 배포할 것을 권장합니다.

그러나 HackShield에서는 서버 연동 클라이언트 크랙 감시 솔루션을 이용해서 크랙한 실행 파일로 게임을 실행하는 것을 원천 차단할 수 있는 기능을 제공합니다.

일부 사용자나 해커는 Windows XP에서 제공하는 하위 호환성 모드(Compatibility Mode)로 프로그램을 실행함으로써 프로그램이 오동작하도록 시도합니다. 하위 호환성 모드로 실행될 경우 현재 시스템이 Windows XP이더라도 Windows 98이나 ME, Windows 2000 등의 운영 체제인 것처럼 프로그램이 판단하게 되므로 예기치 못한 결과를 초래할 수 있습니다. 만약

호환성 모드에서 프로그램이 실행 중인 경우 HackShield 모듈의 초기화 함수를 호출하면 에러 값(HS_ERR_COMPATIBILITY_MODE_RUNNING)을 리턴합니다.

초기화 함수를 호출할 때 기본 기능 옵션 이외에 부가 기능 옵션을 선택할 수 있습니다.

핵실드 서비스 시작 함수 작성

HackShield의 해킹 차단 및 해킹 툴 탐지 기능을 실행하기 위해서는 서비스 시작 함수 `_AhnHS_StartService`를 다음 예제와 같이 호출합니다.

```
예제
BOOL HS_StartService()
{
    int          nRet = 0;
    TCHAR szMsg[MAX_PATH];

    // ① _AhnHS_StartService 함수를 호출하여 핵실드 서비스를 시작합니다.
    nRet = _AhnHS_StartService();

    // ② _AhnHS_StartService 함수의 리턴 값을 검사하여 에러 처리합니다.
    if ( nRet != HS_ERR_OK )
    {
        switch ( nRet )
        {
            case HS_ERR_START_ENGINE_FAILED:
            case HS_ERR_DRV_FILE_CREATE_FAILED:
            case HS_ERR_REG_DRV_FILE_FAILED:
            case HS_ERR_START_DRV_FAILED:
            default:
                wsprintf ( szMsg, "해킹 방지 기능에 문제가 발생
하였습니다.(%x)", nRet );
                break;
        }

        MessageBox( NULL, szMsg, szTitle, MB_OK );
        return FALSE;
    }
    return TRUE;
}
```

서비스 시작 함수 `_AhnHS_StartService`는 초기화 함수 `_AhnHS_Initialize`가 정상적으로 호출 완료된 다음 호출해야 합니다.

주의

게임 클라이언트 프로그램을 보다 철저히 보호하기 위해서는 초기화 함수 호출과 거의 동시에 서비스 시작 함수를 호출하는 것이 좋습니다. 서비스 시작 함수를 늦게 호출할수록 해킹 툴이 게임 클라이언트에 침입할 가능성이 높아집니다.

핵심드 Callback 함수 작성

서비스 시작 함수를 호출하면, 현재 서비스 상태를 실시간으로 이벤트로 알려줍니다. **Callback** 함수에 각 이벤트를 처리하는 방법을 아래 예제와 같이 작성합니다.

예제

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

    // ① 각 경우에 대하여 알맞은 에러 메시지를 출력합니다
    switch ( lCode )
    {
        // ② Engine Callback
        case AHNHS_ENGINE_DETECT_GAME_HACK:
            wsprintf( szMsg, "다음의 프로그램과 게임이 함께 실행될 수
없습니다. (%x) \n [%s]", lCode, (LPTSTR)pParam );
            MessageBox( NULL, szMsg, szTitle, MB_OK );
            break;

        // ③ 오토메크로 탐지
        case AHNHS_ACTAPC_DETECT_AUTOMACRO:
            wsprintf(szMsg, _T("매크로 기능으로 의심되는 동작이
감지되었습니다. (Code = %x)", lCode);
            MessageBox(NULL, szMsg, szTitle, MB_OK);
            break;

        // ④ Speed 관련
        case AHNHS_ACTAPC_DETECT_SPEEDHACK:
            wsprintf( szMsg, "스피드해킹으로 의심되는 동작이
감지되었습니다. (%x)", lCode );
            MessageBox( NULL, szMsg, szTitle, MB_OK );
            break;

        //⑤ 디버깅 방지
        case AHNHS_ACTAPC_DETECT_KDTRACE:
        case AHNHS_ACTAPC_DETECT_KDTRACE_CHANGED:
            wsprintf( szMsg, "게임에 대하여 디버깅 시도가
감지되었습니다. (%x)", lCode );
            MessageBox( NULL, szMsg, szTitle, MB_OK );
            break;

        // ⑥그 외 해킹 방지 기능 이상
        case AHNHS_ACTAPC_DETECT_DRIVERFAILED:
        case AHNHS_ACTAPC_DETECT_HOOKFUNCTION:
        case AHNHS_ACTAPC_DETECT_MODULE_CHANGE:
        case AHNHS_ACTAPC_DETECT_LMP_FAILED:
        case AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP:
        case AHNHS_AHNHS_ACTAPC_DETECT_ENGINEFAILED:
        case AHNHS_ACTAPC_DETECT_CODEMISMATCH:
        case AHNHS_ACTAPC_DETECT_ANTIFREESERVER:
        case AHNHS_ACTAPC_DETECT_ABNORMAL_HACKSHIELD_STATUS:
```

```

        wsprintf( szMsg, "해킹 방어 기능에 이상이 발생하였습니다.
(%x)", lCode );
        MessageBox( NULL, szMsg, szTitle, MB_OK );
        break;

    return 1;
}

```

해킹드 서비스 정지 함수 작성

게임 클라이언트 프로그램을 종료하기 전에 먼저 서비스 정지 함수 `_AhnHS_StopService`를 호출하여, 해킹 차단 기능 및 해킹 툴 탐지 기능을 정지합니다.

게임 클라이언트 프로그램을 종료하지 않고 HackShield 서비스만 잠시 중지할 경우에는, 서비스 정지 함수 `_AhnHS_StopService`를 호출한 뒤에 서비스 시작 함수 `_AhnHS_StartService`를 다시 호출합니다.

```

예제
BOOL HS_StopService()
{
    int      nRet = 0;

    // ① _AhnHS_StopService 함수를 호출하여 해킹드 서비스를 정지합니다.
    nRet = _AhnHS_StopService();

    if ( nRet != HS_ERR_OK )
    {
        return FALSE;
    }
    return TRUE;
}

```

해킹드 서비스 종료 함수 작성

게임 클라이언트 프로그램을 완전히 종료하려면, 다음 예제와 같이 서비스 종료 함수 `_AhnHS_Uninitialize`를 호출하여 프로그램에 사용된 메모리를 해제시킵니다.

```

예제
BOOL HS_UnInit()
{
    int      nRet = 0;

    // ① _AhnHS_Uninitialize 함수를 호출하여 해킹드 서비스를 종료합니다.
    nRet = _AhnHS_Uninitialize();

    if ( nRet != HS_ERR_OK )
    {
        return FALSE;
    }
    return TRUE;
}

```

```
}
```

서비스 종료 함수 `_AhnHS_Uninitialize`가 호출되면 더 이상 게임 클라이언트가 Hackshield에 의해서 보호받을 수 없습니다. 따라서, 서비스 종료 함수는 게임 클라이언트가 종료되는 마지막 시점에 호출합니다.

주의

게임 클라이언트 프로그램이 비정상적으로 종료되면 HackShield의 해킹 차단 드라이버를 언로드하지 못할 수 있습니다. 이 경우 초기화 함수 `_AhnHS_Initialize`를 다시 호출하면, 사용자의 시스템에 HackShield의 해킹 차단 드라이버가 이미 로드되어 있으므로 에러(`HS_ERR_INIT_DRV_FAILED`)를 리턴합니다. 이 경우에는 시스템을 리부팅하여 게임 클라이언트 프로그램을 처음부터 다시 실행시켜야 합니다.

2.4. Application Programming Interface

AhnHS_Initialize

DESCRIPTION

HackShield를 초기화하고, 옵션을 설정합니다. 프로그램이 초기화될 때 한번만 호출할 수 있습니다. 다른 게임 프로그램에서 HackShield를 사용하고 있거나 서비스가 비정상적으로 종료되었을 경우 오류가 발생할 수 있습니다.

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		EHSvc.dll의 전체 경로
Pfn_Callback		Callback 함수의 포인터
nGameCode	4자리 숫자	각 게임에 해당하는 고유 ID 코드
szLicenseKe	24자리 문자열	각 게임에 해당하는 라이선스 키
dwOption		초기화 옵션 설정
unSHackSensingRatio		스피드 핵 감지율 레벨

OPTIONS

주의

아래 옵션은 코어 옵션으로 함수 호출 시 옵션으로 적용하지 않아도 자동으로 적용이 됩니다.

AHNHS_CHKOPT_SPEEDHACK
AHNHS_CHKOPT_READWRITEPROCESSMEMORY
AHNHS_CHKOPT_KDTRACER , AHNHS_CHKOPT_OPENPROCESS
AHNHS_CHKOPT_AUTOMOUSE , AHNHS_CHKOPT_PROCESSCAN

AHNHS_CHKOPT_SPEEDHACK (Core Option)

스피드 해킹 차단 기능을 사용합니다. 하드웨어 제어 또는 소프트웨어 방식의 스피드 해킹이 발견되면, 스피드 해킹 동작 여부를 **Callback** 함수를 통하여 알려줍니다. 스피드 해킹을 진단하는 감지율 레벨 설정에 따라 감지 민감도가 달라집니다. SPEEDHACK SENSING RATIO를 참고하십시오.

AHNHS_CHKOPT_READWRITEPROCESSMEMORY (Core Option)

HackShield DLL을 사용하고 있는 현재 프로세스의 메모리를 외부 프로세스에서 읽고 쓰기할 수 없도록 차단합니다.

AHNHS_CHKOPT_KDTRACER (Core Option)

커널 모드 디버거 실행을 감지하여 게임 프로세스에게 알려줍니다.

AHNHS_CHKOPT_OPENPROCESS (Core Option)

HackShield DLL을 사용하고 있는 현재 프로세스에 대한 정보를 얻을 수 있는 `OpenProcess` API 함수 호출을 차단합니다.

AHNHS_CHKOPT_AUTOMOUSE (Core Option)

오토 마우스 종류의 프로그램이 현재 프로세스에 아무런 영향을 미칠 수 없도록 차단합니다.

AHNHS_CHKOPT_PROCESSSCAN (Core Option)

주기적으로 프로세스 목록을 검사하여 해킹 툴 실행 여부를 검사합니다.

AHNHS_CHKOPT_ALL

위에서 설명한 옵션을 모두 포함하는 옵션입니다.

AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION

지정한 보호 대상 파일의 메모리 영역을 보호하는 옵션입니다.

AHNHS_CHKOPT_ANTIFREESERVER

게임 서버가 아닌 다른 주소로의 접속을 방지하는 옵션입니다.

AHNHS_USE_LOG_FILE

해킹 시도 동작과 관련된 로그를 기록합니다. 로그 파일은 `EhSvc.dll`이 위치한 디렉터리에 `hshield.log`라는 이름으로 만들어집니다. 로그 파일은 문제가 발생할 경우 원인을 분석하기 위해 사용됩니다. 로그 파일은 암호화되어 있기 때문에 별도의 프로그램을 통해서만 내용을 확인할 수 있습니다.

AHNHS_ALLOW_SVCHOST_OPENPROCESS

NT 계열의 운영 체제에서 구동되는 서비스 프로그램인 `SvcHost`에서 게임 프로세스를 오픈할 수 있게 해줍니다. Windows XP에서는 새로 추가된 Windows Audio 서비스를 통하여 사운드 출력을 하며 이 서비스에서

클라이언트 프로세스를 오픈하는 동작을 하기 때문에 **OpenProcess** 차단 옵션을 사용하면 사운드 출력이 정상적으로 출력되지 않을 수 있습니다. 따라서 Windows XP에서 DirectMusic또는 DirectSound을 통한 사운드 출력이 정상적으로 동작하지 않는 경우 이 옵션을 추가하여 사용합니다.

AHNHS_ALLOW_LSASS_OPENPROCESS

NT 계열의 운영 체제에서 구동되는 서비스 프로그램인 **LSASS.exe**에서 게임 프로세스를 오픈할 수 있게 해줍니다. 게임 클라이언트 내부에서 직접 신용카드 결제 컨트롤을 포함하는 경우 **LSASS.exe**에서 게임 프로세스에 접근을 하게 되므로 이 옵션을 사용하여 프로세스 오픈을 허용합니다.

AHNHS_ALLOW_CSRSS_OPENPROCESS

NT 계열의 운영 체제에서 구동되는 서비스 프로그램인 **CSRSS.exe**에서 게임 프로세스를 오픈할 수 있게 해줍니다. 게임 클라이언트 내부에서 직접 신용카드 결제 컨트롤을 포함하는 경우 **CSRSS.exe**에서 게임 프로세스에 접근을 하게 되므로 이 옵션을 사용하여 프로세스 오픈을 허용합니다.

AHNHS_DONOT_TERMINATE_PROCESS

프로세스 목록을 검사하여 해킹 툴을 탐지하는 경우 해당 해킹 툴 프로세스를 강제 종료시키는 것을 기본 옵션으로 하고 있으나 해킹 툴 프로세스를 강제 종료시키지 않고자 하는 경우 이 옵션을 추가적으로 사용합니다. **AHNHS_CHKOPT_PROCESSCAN** 옵션을 사용할 경우 이 옵션과 같이 사용할 것을 권장합니다.

AHNHS_DISPLAY HACKSHIELD LOGO

초기화 시 HackShield 로고를 화면에 보여줍니다. 별도의 스레드로 동작하며 2초 후에 자동으로 로고가 사라집니다. HackShield가 동작하고 있다는 것을 보여주기 위해 사용합니다.

AHNHS_ALLOW_SWITCH_WINDOW

전체화면으로 실행되는 게임에 대하여 강제로 창 모드로 실행시키는 해킹 툴을 탐지 하는 기능을 사용하면서 게임 중 다른 어플리케이션과의 화면 전환을 가능하게 하는 옵션입니다. 창모드로 실행이 불가능한 게임에 대해서만 적용되는 사항이므로 전체 화면 모드와 창 모드를 모두 지원하는 게임에서는 사용할 수 없습니다.

AHNHS_CHKOPT_STANDALONE

핵쉴드의 단독 실행 옵션입니다. 핵쉴드는 기본적으로 멀티 로딩을 지원하지만, 이 옵션을 사용하면 동일한 게임 코드를 사용하는 게임에 대하여 중복 실행이 불가능하도록 핵쉴드에서 오류 코드를 리턴합니다.

AHNHS_CHKOPT_PROTECT_D3DX

DirectX 모듈의 VTable을 후킹 하는 해킹 툴을 차단하는 기능을 활성화 하는 옵션입니다. 이 옵션은 Windows NT 이상의 OS에서만 동작 됩니다.

해당 옵션을 사용하는 경우에는 콜백함수(HS_CallbackProc)에 AHNHS_ACTA PC DETECT ABNORMAL FUNCTION CALL 대한 case 구문 처리를 추가

해주어야 합니다.

참고

특히, FPS 게임일 경우 DirectX 관련 해킹툴이 90% 이상을 차지하기 때문에 해당 옵션을 사용하는 것을 권장합니다.

AHNHS_CHKOPT_SELF_DESTRUCTION

핵실드에서 해킹 행위를 감지하여 Callback 함수 호출 한 이후에도 게임 프로세스가 종료 되지 않은 경우, 일정 시간 이후 (1분 후) 핵실드 자체적으로 프로세스를 종료시키는 기능입니다.

Callback 함수를 수정하여 핵실드 핵 감지 기능을 무력화 시키는 해킹툴에 대응하기 위한 기능입니다.

핵실드 자체 종료는 비정상적인 종료를 유발하므로, 해당 기능을 사용할 때는 반드시 Callback 함수에서 핵실드 에러가 리턴된 이후에 1분 이내에 게임 프로세스를 정상적으로 종료 시켜야 합니다.

해당 옵션을 사용할 때는 반드시 아래 표를 참고하여 콜백 함수에 콜백 코드에 대하여 적절한 처리를 한 이후 프로세스를 종료 시켜야 합니다. 만약 콜백함수에 아래에 해당 하는 콜백 코드가 단 하나라도 처리가 안된 다면, 핵실드에 의해 비정상적으로 게임이 종료될 수 있습니다.

표 2-3 [AHNHS_CHKOPT_SELF_DESTRUCTION 적용시 처리 콜백 코드]

적용 옵션	Callback 함수에 적용해야 하는 콜백 코드
필수 적용 콜백 코드	AHNHS_ACTAPC_DETECT_AUTOMACRO
	AHNHS_ACTAPC_DETECT_SPEEDHACK
	AHNHS_ENGINE_DETECT_GAME_HACK
	AHNHS_ACTAPC_DETECT_ABNORMAL_MEMORY_ACCESS
	AHNHS_ACTAPC_DETECT_ENGINEFAILED
	AHNHS_ACTAPC_DETECT_HOOKFUNCTION
	AHNHS_ACTAPC_DETECT_KDTRACE
	AHNHS_ACTAPC_DETECT_CODEMISMATCH
	AHNHS_ACTAPC_DETECT_ABNORMAL_HACKSHIELD_STATUS
	AHNHS_ACTAPC_DETECT_DRIVERFAILED
AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION	AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP
	AHNHS_ACTAPC_DETECT_LMP_FAILED
AHNHS_CHKOPT_ANITFREES EVER	AHNHS_ACTAPC_DETECT_ANTIFREESERVER
AHNHS_CHKOPT_PROTECTSCREENEX	AHNHS_ACTAPC_DETECT_PROTECTSCREENFAILED

예제

```
int stdcall AhnHS Callback(long lCode, long lParamSize, void* pParam)
{
    switch(lCode)
    {
        //Engine Callback
        case AHNHS_ENGINE_DETECT_GAME_HACK:
        case AHNHS_ACTAPC_DETECT_SPEEDHACK:
        case AHNHS_ACTAPC_DETECT_HOOKFUNCTION:
        case AHNHS_ACTAPC_DETECT_KDTRACE:
```

```

        case AHNHS_ACTAPC_DETECT_ABNORMAL_MEMORY_ACCESS:
        case AHNHS_ACTAPC_DETECT_ENGINEFAILED:
        case AHNHS_ACTAPC_DETECT_AUTOMACRO:
        case AHNHS_ACTAPC_DETECT_CODEMISMATCH:
        case AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP:
        case AHNHS_ACTAPC_DETECT_ANTIREESEVER:
        case AHNHS_ACTAPC_DETECT_ABNORMAL HACKSHIELD STATUS:
            MessageBox ( );
            ExitGame ( );
            break;

    {

        return 1;
    }

```

AHNHS_DISPLAY_HACKSHIELD_TRAYICON

핵싧드가 싧행 되면 시스템 트레이에 핵싧드 트레이 아이콘을 보여줍니다.

AHNHS_CHKOPT_DETECT_VIRTUAL_MACHINE

핵싧드가 가상 머신이나 에뮬레이터에서 싧행 되는 것을 방지 합니다.

이 옵션을 적용하면 이미 정의되어 있는 가상 머신이나 에뮬레이터에서 게임이 싧행 되는 것을 방지할 수 있습니다.

이 기능은 가상 머신의 기능 변경에 의하여 감지 로직이 추가, 또는 변경 될 수 있습니다.

참고

AHNHS_CHKOPT_DETECT_VIRTUAL_MACHINE 옵션으로 감지 가능한 가상 머신 프로그램은 다음과 같습니다.

Virtual, VMWare Workstation, Virtual Box

(단, Virtual Box의 경우 9X 계열의 Guest Machine에 대한 감지는 하지 않습니다.)

SPEEDHACK SENSING RATIO

AHNHS_SPEEDHACK_SENSING_RATIO_HIGHEST

가장 민감하게 동작하는 레벨로서 기준치는 14.5%와 -12.5%입니다. 이는 A Speeder 스피드 핵 프로그램을 기준으로 0.8706배 이하로 속도를 느리게 하거나 1.1487배 이상으로 속도를 빠르게 할 때 스피드핵으로 진단합니다.

AHNHS_SPEEDHACK_SENSING_RATIO_HIGH

두 번째로 민감하게 동작하는 레벨로서 기준치는 17.5%와 -15.5%입니다. 이는 A Speeder 스피드 핵 프로그램을 기준으로 0.8409배 이하로 속도를 느리게 하거나 1.1892배 이상으로 속도를 빠르게 할 때 스피드핵으로 진단합니다.

AHNHS_SPEEDHACK_SENSING_RATIO_NORMAL

일반적인 경우에 동작하는 레벨로서 기준치는 17.5%와 -21%입니다. 이는 A Speeder 스피드 핵 프로그램을 기준으로 0.7846배 이하로 속도를 느리게 하거나 1.2746배 이상으로 속도를 빠르게 할 때 스피드핵으로 진단합니다.

AHNHS_SPEEDHACK_SENSING_RATIO_LOW

두 번째로 둔감하게 동작하는 레벨로서 기준치는 31.5%와 -23.5%입니다. 이는 A Speeder 스피드 해킹 프로그램을 기준으로 0.7579배 이하로 속도를 느리게 하거나 1.3195배 이상으로 속도를 빠르게 할 때 스피드해킹으로 진단합니다.

AHNHS_SPEEDHACK_SENSING_RATIO_LOWEST

가장 둔감하게 동작하는 레벨로서 기준치는 39.0%와 -28.5%입니다. 이는 A Speeder 스피드 해킹 프로그램을 기준으로 0.7071배 이하로 속도를 느리게 하거나 1.4142배 이상으로 속도를 빠르게 할 때 스피드해킹으로 진단합니다.

RETURN VALUE

HS_ERR_OK (Value = 0x0000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_INVALID_PARAM(Value = 0x002)

- 설 명 : 파라미터가 잘못 되었습니다.
- 원 인 : Callback 함수의 포인터가 잘못 설정되었거나 라이선스 키 값이 NULL인 경우에 발생하는 오류입니다.
- 확인사항 : 개발 과정에서만 발생할 수 있으므로, 별도의 처리는 하지 않아도 됩니다.

HS_ERR_NOT_INITIALIZED(Value = 0x003)

- 설 명 : 초기화에 실패하였습니다.
- 원 인 : 해킹드 동작 시 사용하는 시스템 라이브러리가 정상적으로 로드 되지 않아 발생하는 오류입니다.
- 확인사항 : 지속적으로 발생한다면 (주)안철수연구소에 문의하시기 바랍니다.

HS_ERR_COMPATIBILITY_MODE_RUNNING (Value = 0x004)

- 설 명 : 게임이 호환성 모드로 실행되었습니다.
- 원 인 : 게임 클라이언트가 Windows XP 계열에서 제공하는 호환성 모드(Compatibility Mode)로 실행된 경우 발생하는 오류입니다.
- 확인사항 : 호환성 모드로 게임을 실행하였다면 악의적인 목적을 가진 사용자라고 판단되므로 프로그램을 강제 종료하고 다시 시작하도록 해야 합니다.

HS_ERR_INVALID_LICENSE (Value = 0x100)

- 설 명 : 라이선스가 잘못 입력되었습니다.
- 원 인 : 초기화 함수의 파라미터로 설정한 게임 코드와 라인선스 키 값이 실제 값과 일치하지 않을 때 발생하는 오류입니다.
- 확인사항 : 라이선스값이 정확히 입력이 되었는지 확인합니다.

HS_ERR_INVALID_FILES (Value = 0x101)

- 설 명 : 핵설드 파일이 잘못 되었습니다.
- 원 인 : 인터페이스 DLL 파일(EhSvc.dll)이 위/변조되었거나, 버전이 같지 않을 경우 발생하는 오류입니다. 인터페이스 DLL 파일이 위/변조되면 해킹 차단 기능이 무력화될 수 있습니다.
- 확인사항 : 핵설드 파일이 과거 버전인거나 핵설드 폴더에 파일이 있는지 확인해야 합니다.

HS_ERR_INIT_DRV_FAILED (Value = 0x102)

- 설 명 : 핵설드 드라이버 시작이 실패하였습니다.
- 원 인 : 해킹 차단 기능 드라이버 초기화에 실패했을 때 발생하는 오류입니다. 드라이버 초기화에 실패하면 해킹 차단 기능이 정상 동작할 수 없으므로 프로그램을 강제 종료하고 다시 시작해야 합니다.
- 확인사항 :
 - ① 시스템 폴더에 **EagleNT.sys** 파일이 권한이 변경되어 접근할수 없는 형태로 되어 있는지 확인합니다.
 - ② **EagleNT.sys** 파일이 언로드 되지 않고 떠 있는지 확인합니다.

HS_ERR_ALREADY_INITIALIZED (Value = 0x104)

- 설 명 : 핵설드가 이미 초기화 되어 있습니다.
- 원 인 : **_AhnHS_Initialize**를 호출하여 시스템이 이미 초기화되어 있는 상태일 때 발생하는 오류입니다.
- 확인사항 : **_AhnHS_Initialize**는 프로그램 초기화 시점에 한 번만 호출할 수 있으며 중복으로 호출할 수 없습니다.
내부적으로 위 함수를 여러 번 호출하지 않았는지 확인합니다.

HS_ERR_DEBUGGER_DETECT (Value = 0x105)

- 설 명 : 디버거가 감지 되었습니다.
- 원 인 : 시스템에서 디버거가 실행 중일 경우 발생하는 오류 입니다.

- 확인사항 :
 - ① 개발 과정에 있다면 릴리즈 버전이 아닌 **Dev** 버전을 사용하여 디버깅을 하시기 바랍니다. (**Remark** 참조)
 - ② 사용자에게 해당 에러가 발생한다면 해킹 침입 위험이 있으므로 게임을 종료하고 디버거를 중지한 다음 다시 실행하도록 합니다.

만약, 디버거가 없는데 해당 에러가 발생한다면 (주)안철수연구소에 문의하시기 바랍니다.

HS_ERR_NEED_ADMIN_RIGHTS (Value = 0x107)

- 설명 : 어드민 계정이 필요합니다.
- 원인 : 게임 클라이언트가 **Windows NT** 계열의 시스템에서 관리자 계정이 아닌 일반 사용자 계정의 권한으로 실행된 경우 발생하는 오류입니다.
- 확인사항 : 유저 계정에서 실행 하였다면, 윈도우 계정이 생성되었는지 확인합니다. 일반 사용자 계정 권한으로 핵쉴드가 제공하는 기능을 사용하기 위해서는 핵쉴드 윈도우 계정이 먼저 생성되어 있어야 합니다. 자세한 내용은 '**8.2. User 권한 실행 지원 기능**'을 참고하십시오.

HS_ERR_UNKNOWN (Value = 0x001)

- 설명 : 알 수 없는 에러가 발생하였습니다.
- 원인 : 위에서 정의하지 않은 알 수 없는 문제가 발생했을 때 발생하는 오류 입니다.
- 확인사항 : 지속적으로 발생할 경우 (주)안철수연구소에 문의하시기 바랍니다.

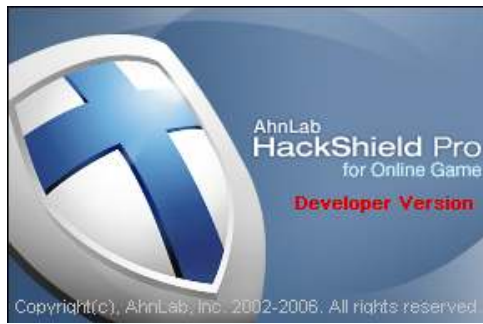
REMARKS

개발 과정이나, 테스트 과정에서 게임 클라이언트를 디버깅 하기 위해서는 개발자용 HShield.lib, Ehvsc.dll, HShield.dat 을 사용하셔야 합니다.

(Ehvsc.dll 과 HShield.dat 파일이 릴리즈 버전과 함께 사용하면 서버 연동시 문제가 발생할 수 있으니 유의하시기 바랍니다.)

- 경 로 : \Developer\

개발자용 모듈을 사용하게 되면 핵쉴드 실행시 아래와 같은 로고가 생깁니다. (클릭하면 사라집니다.)



AhnHS_Callback

DESCRIPTION

해킹 차단 및 해킹 툴 탐지 결과에 대한 정보를 전달하는 Callback 함수이며 다음과 같은 이벤트를 전달합니다.

해킹 차단 기능과 관련된 이벤트

해킹 툴 탐지 기능과 관련된 이벤트

STRUCTURE

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

PARAMETERS

Parameter	Value	Description
lCode	Long	이벤트 코드
lParamSize	Long	이벤트 파라미터의 크기
pParam	Void*	이벤트 파라미터

EVENTS

AHNHS_ENGINE_DETECT_GAME_HACK

시스템 상에서 실행되고 있는 게임 전용 해킹 툴을 탐지한 경우, Callback 함수로 전달 되는 이벤트입니다. 전달된 해킹 툴 정보를 사용하여, 게임 개발사는 해킹 툴의 실행 파일을 강제로 삭제하는 등의 조치를 취할 수 있습니다.

pParam: 탐지된 게임 전용 해킹 툴의 실행 파일 이름(파일 경로 포함)

lParamSize: 탐지된 해킹 툴 실행 파일 이름의 문자 길이

AHNHS_ENGINE_DETECT_WINDOWED_HACK

전체화면으로 실행되는 게임에 대하여 강제로 창모드로 실행시키는 해킹 툴을 탐지한 경우, Callback 함수로 전달되는 이벤트 입니다. 이것은 창모드로 실행이 불가능한 게임에 대해서만 적용되는 사항이므로 전체화면 모드와 창모드를 모두 지원하는 게임에서는 사용하면 안됩니다.

pParam: NULL

lParamSize: 0

AHNHS_ACTAPC_DETECT_ALREADYHOOKED

해킹 차단 기능이 실행 중일 때, 해킹 차단 기능에서 보호하고자 하는 API 함수가 이미 후킹되어 있을 경우 전달되는 이벤트입니다. 해킹 툴이 아닌 정상 프로그램에서도 기능 구현을 위해서 후킹을 할 수 있으므로, 해킹 툴 인지에 대한 판단은 게임 개발사 내부 규칙에 의해서 결정해야 합니다.

pParam: ACTAPCPARAM_DETECT_HOOKFUNCTION*

IParamSize: ACTAPCPARAM_DETECT_HOOKFUNCTION 구조체의 길이

AHNHS_ACTAPC_DETECT_HOOKFUNCTION

해킹 차단 기능이 동작 중일 때 Win32 함수나 보호 함수가 후킹되었을 경우에 전달되는 이벤트입니다. 이 이벤트가 발생하면, 해킹 툴에 의한 침입일 위험이 높으므로 게임 프로그램을 강제 종료해야 합니다.

pParam : ACTAPCPARAM_DETECT_HOOKFUNCTION*

IParamSize : ACTAPCPARAM_DETECT_HOOKFUNCTION 구조체의 길이

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

szFunctionName: 후킹된 함수 이름

szModuleName: 후킹한 모듈 이름

AHNHS_ACTAPC_DETECT_AUTOMOUSE

오토 마우스 종류의 프로그램을 사용해서 키보드나 마우스를 조정하여 자동으로 데이터를 입력할 경우 발생하는 이벤트입니다. 다음과 같은 구조체 포인터가 전달됩니다.

pParam : ACTAPCPARAM_DETECT_AUTOMOUSE

IParamSize : ACTAPCPARAM_DETECT_AUTOMOUSE의 크기.

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

byDetectType의 값과 의미는 다음과 같으며, dwPID, szProcessName, szAPIName는 현재 사용되지 않습니다.

EAGLE_AUTOMOUSE_APCTYPE_SHAREDMEMORY_ALTERATION

(3): 오토마우스 차단 기능을 하는 라이브러리의 내부 데이터를 변경합니다. HackShield가 해킹을 차단하는데 사용하는 내부 정보를 해킹 프로그램이 수정을 한 경우에 차단 기능이 동작할 수 없으므로, 게임 프로그램을 강제로 종료시켜야 합니다.

EAGLE_AUTOMOUSE_APCTYPE_API_CALLED: 키보드, 마우스 관련 API 호출

EAGLE_AUTOMOUSE_APCTYPE_API_ALTERATION: API 후킹 변조

AHNHS_ACTAPC_DETECT_AUTOMACRO

오토 마우스 종류의 프로그램을 사용해서 키보드나 마우스를 조정하여 자동으로 데이터를 입력할 경우 발생하는 이벤트입니다. 정상적이지 않은 입력 값이 반복해서 나타나는 현상이므로 게임 프로그램을 강제 종료시켜야 합니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_DRIVERFAILED

해킹 차단 드라이버가 시스템에 로드되지 않을 경우 발생하는 이벤트입니다. 해킹 차단 드라이버가 시스템에 로드되지 않으면 해킹 차단 기능이 정상 동작하지 않으므로 즉시 게임 프로그램을 강제 종료시킵니다. 또한, 비정상적으로 드라이버가 제거된 경우에는, 시스템이 불안정해질 수 있으므로 시스템을 다시 시작합니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_SPEEDHACK

시스템의 시간 변화 속도가 비정상적일 경우 발생하는 이벤트입니다. 하드웨어 또는 소프트웨어 방식의 스피드 핵이 실행 중일 가능성이 매우 높으므로, 시간에 민감한 게임 프로그램은 강제 종료시킵니다. 하드웨어 방식의 스피드 핵은 시스템의 시간을 제어하는 하드웨어를 직접 조작하여 시스템 전체 시간에 영향을 미칩니다. 이러한 방식의 하드웨어 스피드 핵은 Windows 버전에 따라 운영 체제 자체에서 차단하기도 합니다.

pParam: (double *) 최근 5초 동안의 시간 데이터 정보

IParamSize: pParam으로 전달된 시간 데이터의 개수

AHNHS_ACTAPC_DETECT_KDTRACE

커널 레벨 혹은 어플리케이션 레벨의 디버거에 의해 디버거 트레이스가 발생하면 전달되는 이벤트입니다. 이 이벤트가 발생하면 해커와 같은 악의적인 사용자가 게임 프로그램에 대한 디버깅 작업을 진행 중일 가능성이 높으므로 게임 프로그램을 종료시키는 것이 좋습니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_ABNORMAL_MEMORY_ACCESS

허가되지 않은 프로세스가 게임 프로세스의 메모리에 접근할 때 발생하는 이벤트입니다. 메모리 접근을 시도한 프로세스의 경로와 이름을 알고 있으므로

어떤 프로세스가 접근했는지 확인할 수 있습니다.

pParam: 탐지된 게임 전용 해킹 툴의 실행 파일 이름(파일 경로 포함)

IParamSize: 탐지된 해킹 툴 실행 파일 이름의 문자 길이

AHNHS_ACTAPC_DETECT_KDTRACE_CHANGED

디버거 트레이스 차단 루틴이 변경된 경우 나타나는 이벤트입니다. 이 이벤트가 발생하면, 게임 프로그램에 대한 디버거 트레이스 차단 루틴이 수행되고 있는 동안 커널 레벨의 디버거 프로그램이 활성화되었음을 의미하므로 게임 프로그램을 강제 종료시켜야 합니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_MODULE_CHANGE

HackShield 모듈에 대한 위/변조가 감지되었을 경우 나타나는 이벤트입니다. 이 이벤트가 발생하면 HackShield가 정상적으로 동작하지 않을 수 있으므로 게임 프로그램을 강제로 종료시켜야 합니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_ENGINEFAILED

HackShield의 휴리스틱 엔진(3N.mhe) 파일이 삭제되었거나 정상적이지 않아서 로드가 안되었을 경우 나타나는 이벤트입니다. 이 이벤트가 발생하면 HackShield의 ProcessScan 기능이 정상적으로 동작하지 않을 수 있으므로 게임을 강제 종료시켜야 합니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_CODEMISMATCH

HackShield의 인터페이스 모듈인 Ehsvc.dll 모듈의 코드영역이 조작 되었을 경우 나타나는 이벤트입니다. 이 이벤트가 발생하면 HackShield의 기능이 정상적으로 동작하지 않을 수 있으므로 게임을 강제 종료 시켜야 합니다.

pParam: NULL

IParamSize: 0

AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP

보호 대상으로 지정한 파일의 메모리 영역이 조작되었을 경우 나타나는 이벤트입니다. 이 이벤트가 발생하면 해당 모듈이 정상적으로 동작하지 않을 수 있으므로 게임 프로그램을 강제로 종료시켜야 합니다.

pParam: 조작된 모듈 이름(조작된 페이지 시작주소)

IParamSize: 0

AHNHS_ACTAPC_DETECT_LMP_FAILED

LMP 보호 기능이 외부요인(해킹 및 시스템 이상) 으로 인해 정상동작하지 않을경우 발생하는 이벤트입니다. 게임 재실행 및 시스템을 재부팅합니다.

IPParamSize: 0

AHNHS_ACTAPC_DETECT_ANTIFREESERVER

게임 프로그램에서 정상적이지 않은 주소의 서버로 접속하려 할 때 발생하는 이벤트입니다. 이 경우 정상적인 게임 실행이 아니므로 게임을 강제 종료 시켜야 합니다. pParam은 접속하려는 주소 값입니다. 이 주소 값은 내부 확인용으로만 사용하고 사용자에게 보여주지 않는 것이 좋습니다.

pParam: 접속하려는 IP 주소

IPParamSize: 0

AHNHS_ACTAPC_DETECT_ABNORMAL_HACKSHIELD_STATUS

HackShield 동작이 정상적으로 이루어지지 않았을 경우 나타나는 이벤트입니다. 이 이벤트가 발생하면 HackShield가 정상적으로 동작하지 않고 있으므로 게임 프로그램을 강제로 종료시켜야 합니다.

pParam: NULL

IPParamSize: 0

AHNHS_ACTAPC_DETECT_PROTECTSCREENFAILED

가상 데스크탑을 사용하여 게임을 보호하는 기능이 공격을 당하여 동작하지 않는 경우 발생합니다. 해당 이벤트가 발생하면 더 이상 가상 데스크탑 기능이 동작하지 않으므로 가상 데스크탑 기능을 사용하는 게임 프로그램은 강제로 종료 시켜야 합니다.

pParam: NULL

IPParamSize: 0

AhnHS_StartService

DESCRIPTION

해킹 툴 탐지 기능과 해킹 차단 기능을 동작시킵니다. _AhnHS_Initialize 함수를 호출한 다음에 호출해야 하며, 중복 호출할 수 없습니다. _AhnHS_StopService 함수를 호출하여 서비스를 중지한 경우에는, 이 함수를 다시 호출하여 서비스를 다시 시작시킬 수 있습니다.

SYNTAX

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

PARAMETERS

없음.

RETURN VALUE

HS_ERR_OK (Value = 0x000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- 설 명 : 핵셴드가 초기화 되지 않았습니다.
- 원 인 : `_AhnHS_Initialize` 함수를 호출하지 않았거나 함수 호출에 실패하여 **HackShield**를 초기화하지 않은 상태에서 이 함수를 호출했을 때 발생하는 오류입니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_START_ENGINE_FAILED (Value = 0x200).

- 설 명 : 엔진을 로드하는데 실패하였습니다.
- 원 인 : 초기화 함수를 호출할 때 해킹 툴 프로세스 감지 옵션 (`AHNHS_CHKOPT_PROCESSSCAN`)을 설정했으나, 해킹 툴 패턴 엔진 초기화에 실패한 경우 발생하는 오류입니다. 해킹 툴 패턴 엔진 관련 파일이 제대로 설치되지 않거나, 패턴 엔진 관련 파일(`v3pro32s.dll`, `v3warpds.v3d`, `v3warpns.v3d`)이 핵셴드 폴더에 제대로 설치되지 않았을 경우에 발생합니다.
- 확인사항 : 이 오류가 발생하면 프로그램을 강제 종료하고, 다시 시작하거나 다시 설치해야 합니다.

HS_ERR_ALREADY_SERVICE_RUNNING (Value = 0x201)

- 설 명 : 핵셴드가 이미 구동중입니다.
- 원 인 : `_AhnHS_StartService` 함수를 이미 호출한 상태에서 다시 호출한 경우 발생하는 오류입니다.
- 확인사항 : 이 함수를 다시 호출하려면 반드시 `_AhnHS_StopService` 를 호출해서 서비스를 종료해야 합니다. 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_DRV_FILE_CREATE_FAILED (Value = 0x202)

- 설명 : 핵섀드 드라이브 파일 생성이 실패하였습니다.
- 원인 : 해킹 차단을 위한 드라이버 파일 생성에 실패했을 때 발생하는 오류입니다. **HackShield** 프로그램 내부에서는 해킹 차단 처리를 위한 드라이버를 게임이 시작하는 시점에 생성하여 로드시켜줍니다. 게임을 시작할 때 드라이버 파일 생성에 문제가 발생한 경우 발생하는 오류입니다.
- 확인사항 :
 - ① 현재 세션이 시스템 폴더에 대하여 쓰기 권한이 있는지 확인해 보셔야 합니다.
 - ② 시스템 폴더에 핵섀드 드라이버 파일 (**EagleNT.sys**) 이 속성이 읽기 전용 또는 액세스 가능하지 않은 속성으로 되어있는지 확인 해야 합니다. 만약 위와 같은 상황이라면 해당 파일을 삭제하고 핵섀드를 다시 시작합니다.
(지속적으로 발생한다면 (주)안철수연구소에 문의하시기 바랍니다.)

HS_ERR_REG_DRV_FILE_FAILED (Value = 0x203)

- 설명 : 핵섀드 드라이브 파일 등록이 실패하였습니다.
- 원인 : 해킹 차단을 위한 드라이버 파일을 시스템에 등록하는데 실패했을 때 발생하는 오류입니다.
- 확인사항 : 이 오류가 발생하면 **HackShield**가 정상적으로 동작할 수 없으므로 프로그램을 강제 종료하고, 다시 시작하거나 다시 설치합니다.

HS_ERR_START_DRV_FAILED (Value = 0x204)

- 설명 : 핵섀드 드라이브 구동이 실패하였습니다.
- 원인 : 게임 클라이언트가 여러 가지 원인으로 인하여 비정상적으로 종료되면서 `_AhnHS_StopService` 또는 `_AhnHS_Uninitialize` 함수를 호출하지 못하여 기존에 로드했던 드라이버가 정지되거나 로드하지 않아야 하는 드라이버가 계속 로드되어 있는 상태에서 다시 게임을 실행시키면 발생하는 오류입니다. 이 오류가 발생하면 시스템 자체가 불안하거나 문제가 있을 수 있으므로, 프로그램을 종료하고 시스템을 다시 시작해야 합니다.
- 확인사항 :
 - ① 현재 세션이 시스템 폴더에 대하여 쓰기 권한이 있는지 확인해 보셔야 합니다.
 - ② 시스템 폴더에 핵섀드 드라이버 파일 (**EagleNT.sys**) 이 속성이 읽기 전용 또는 액세스 가능하지 않은 속성으로 되어있는지 확인 해야 합니다. 만약 위와 같은 상황이라면 해당 파일을 삭제하고 핵섀드를 다시 시작합니다.
(지속적으로 발생한다면 (주)안철수연구소에 문의하시기 바랍니다.)

HS_ERR_ALREADY_GAME_STARTED (Value = 0x206)

- 설명 : 게임이 이미 실행 중입니다.
- 원인 : 초기화 함수를 호출할 때 핵실효 단독 실행 옵션(AHNHS_CHKOPT_STANDALONE)을 적용한 경우에만 발생하는 오류입니다. 동일한 게임코드로 게임이 이미 실행 중인 경우 발생합니다.
- 확인사항 : 초기화 함수를 호출할 때 핵실효 단독 실행 옵션(AHNHS_CHKOPT_STANDALONE)을 적용하였다면, 게임 프로세스는 하나만 실행할 수 있습니다. 이 오류가 발생하면 프로그램은 종료되어야 합니다.)

HS_ERR_VIRTUAL_MACHINE_DETECT (Value = 0x207)

- 설명 : 게임이 가상 머신 또는 에뮬레이터 상에서 실행 중입니다.
- 원인 : 초기화 함수를 호출할 때 가상 머신에서의 실행을 방지하는 옵션(AHNHS_CHKOPT_DETECT_VIRTUAL_MACHINE)을 적용한 경우에만 발생하는 오류입니다.
- 확인사항 : 초기화 함수를 호출할 때 가상 머신에서의 실행을 방지하는 옵션(AHNHS_CHKOPT_DETECT_VIRTUAL_MACHINE)을 적용하였다면, 핵실효에서 제어하는 가상 머신 상에서의 실행을 방지합니다.

DESCRIPTION

해킹 차단 기능과 해킹 툴 탐지 기능을 정지시킵니다.

SYNTAX

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

PARAMETERS

없음.

RETURN VALUE

HS_ERR_OK (Value = 0x0000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- 설 명 : 핵셴드가 초기화 되지 않았습니다.
- 원 인 : _AhnHS_Initialize 함수를 호출하지 않았거나 함수 호출에 실패하여 HackShield를 초기화하지 않은 상태에서 이 함수를 호출했을 때 발생하는 오류입니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_SERVICE_NOT_RUNNING (Value = 0x301)

- 설 명 : 핵셴드가 시작되지 않았습니다.
- 원 인 : _AhnHS_StartService 함수를 호출하여 HackShield를 시작하지 않은 상태에서 호출한 경우 발생한 오류입니다. 개발 과정에서만 발생할 수 있는 오류로 별도의 처리는 하지 않아도 됩니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

AhnHS_Uninitialize

DESCRIPTION

시스템 내부적으로 사용되었던 메모리를 해제하고 변수를 초기화합니다.

SYNTAX

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

PARAMETERS

없음.

RETURN VALUE

HS_ERR_OK (Value = 0x0000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_SERVICE_STILL_RUNNING (Value = 0x302)

- 설 명 : 핵셴드가 실행중입니다.
- 원 인 : _AhnHS_StopService 함수를 호출하여 HackShield를 종료하지 않은 상태에서 호출하면 발생하는 오류입니다. 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.
- 확인사항 :

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- 설 명 : 핵셴드가 초기화 되지 않았습니다.
- 원 인 : _AhnHS_Initialize 함수를 호출하지 않았거나 함수 호출에 실패하여 HackShield를 초기화하지 않은 상태에서 이 함수를 호출했을 때 발생하는 오류입니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

DESCRIPTION

HackShield가 제공하는 기능 중 일부분의 기능을 잠시 중단합니다. 메시지 후킹 방어 기능 중 키보드 방어에 대해서만 처리합니다. 게임 실행 중에 결재 등을 하기 위해 Microsoft Internet Explorer의 웹페이지 방식으로 사용자 입력을 받는 경우에 키보드 방어 기능으로 사용자 입력이 작동되지 않는 것을 일시 중지시키는데 사용합니다. 일시 중지 후 반드시 _AhnHS_ResumeService를 사용하여 다시 활성화해야 합니다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
dwPauseOption	DWORD	현재는 AHNHS_CHKOPT_MESSAGEHOOK 옵션만 사용할 수 있습니다. 그 외의 옵션이 전달되면 HS_ERR_INVALID_PARAM 에러가 리턴됩니다.

RETURN VALUE

HS_ERR_OK (Value = 0x000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- 설 명 : 핵실드가 초기화 되지 않았습니다.
- 원 인 : _AhnHS_Initialize 함수를 호출하지 않았거나 함수 호출에 실패하여 HackShield를 초기화하지 않은 상태에서 이 함수를 호출했을 때 발생하는 오류입니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_SERVICE_NOT_RUNNING (Value = 0x301)

- 설명 : 핵셴드가 시작되지 않았습니다.
- 원인 : `_AhnHS_StartService` 함수를 호출하여 `HackShield`를 시작하지 않은 상태에서 호출한 경우 발생한 오류입니다. 개발 과정에서만 발생할 수 있는 오류로 별도의 처리는 하지 않아도 됩니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_INVALID_PARAM (Value = 0x002)

- 설명 : 파라미터가 잘 못 되었습니다.
- 원인 : `dwPauseOption` 값이 `AHNHS_CHKOPT_MESSAGEHOOK`이 아닐 경우에 발생하는 오류입니다.
- 확인사항 :

AhnHS_ResumeService

DESCRIPTION

AhnHs_Pause service를 호출하여 잠시 중단했던 HackShield 기능을 다시 활성화합니다. 메시지 후킹 방어 기능중 키보드 방어에 대해서만 처리합니다. 이 기능의 필요성에 대해서는 _AhnHS_PauseService의 설명을 참고하십시오.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
dwResumeOption	DWORD	현재는 AHNHS_CHKOPT_MESSAGEHOOK 옵션만 사용할 수 있습니다. 그 외의 옵션이 전달되면 HS_ERR_INVALID_PARAM 에러가 리턴됩니다.

RETURN VALUE

HS_ERR_OK (Value = 0x000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_NOT_INITIALIZED (Value = 0x003)

- 설 명 : 핵셴드가 초기화 되지 않았습니다.
- 원 인 : _AhnHS_Initialize 함수를 호출하지 않았거나 함수 호출에 실패하여 HackShield를 초기화하지 않은 상태에서 이 함수를 호출했을 때 발생하는 오류입니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_SERVICE_NOT_RUNNING (Value = 0x301)

- 설 명 : 핵셴드가 시작되지 않았습니다.

- 원 인 : `_AhnHS_StartService` 함수를 호출하여 **HackShield**를 시작하지 않은 상태에서 호출한 경우 발생한 오류입니다. 개발 과정에서만 발생할 수 있는 오류로 별도의 처리는 하지 않아도 됩니다.
- 확인사항 : 이 오류는 개발 과정에서만 발생할 수 있으므로 별도의 처리는 하지 않아도 됩니다.

HS_ERR_INVALID_PARAM (Value = 0x002)

- 설 명 : 파라미터가 잘 못 되었습니다.
- 원 인 : `dwPauseOption` 값이 `AHNHS_CHKOPT_MESSAGEHOOK`이 아닐 경우에 발생하는 오류입니다.
- 확인사항 :

3. 핵실드 업데이트 기능

3.1. 개요

핵실드 업데이트 기능은 기존에 핵실드 패치가 이루어 졌을 때 게임사 패치 기간에 빌드하고 게임사 런처를 통해 업데이트 받는 방식을 개선하기 위해서 개발이 이루어 졌습니다. 핵실드 전용의 업데이트를 통해 빠르게 핵실드 모듈 / 엔진 패치가 이루어질 수 있고 신속하게 해킹툴을 대응 할 수 있습니다.

기능

핵실드 모듈 업데이트 기능

핵실드 업데이트는 핵실드 패치가 필요한 경우 별도의 작업을 거치지 않고 손쉽게 핵실드 모듈을 업데이트 하기 위해서 제공하는 기능입니다..

엔진 업데이트 기능

새로운 해킹툴에 대해 빠르게 대처하기 위하여 **Signature, Heuristic** 엔진을 업데이트 할 수 있도록 합니다.

핵실드 업데이트 ON/OFF 기능

핵실드 업데이트 기능 사용 중 부득이하게 핵실드 업데이트 기능을 사용하지 않아야 하는 경우에 대비하기 위하여 패치셋 설정에 따라 핵실드 업데이트에 대한 ON/OFF가 가능합니다.

스플래쉬 이미지 기능

핵실드 업데이트 진행 중 화면 우측 하단과 중앙에 사용자가 임의로 지정한 이미지를 보여줍니다.

특징

인터페이스 함수(API) 제공

HSUpChk.lib 에서 제공되는 API를 사용하여 게임의 런처나 실행파일에서 제일 처음 호출하여 핵실드 모듈 및 엔진을 업데이트 받을 수 있도록 합니다.

업데이트 환경 파일 생성 프로그램 제공

업데이트 서버 환경 파일을 제공하여 업데이트 받고자 하는 서버를 임의대로 설정할 수 있고 다중 URL을 지원하여 여러대의 업데이트 서버를 구축할 수 있습니다.

테스트 프로그램 제공

HSUpChk.lib에서 제공하는 API를 사용하여 구현한 테스트용 프로그램인

Amazon.exe를 제공합니다. Amazon.exe는 기존의 핵셴드 테스트 기능과 핵셴드 업데이트의 테스트 기능을 제공합니다.

시스템 구조(System Architecture)

핵셴드 업데이트는 HSupChk.lib을 제공하여 클라이언트에서 API를 호출하여 업데이트를 수행할 수 있게 합니다. 업데이트의 전체 구조 및 동작 원리는 다음과 같습니다.

HSUpChk.lib (핵셴드 업데이트 라이브러리)

클라이언트에서 사용되며 업데이트 함수가 호출되면 HsUpdate.exe 가 호출되어 필요한 핵셴드 모듈을 업데이트 할수 있게 합니다.

HSUpdate.env(업데이트 환경 파일)

HSUpSetEnv.exe 툴에서 생성되는 파일로서, 업데이트 서버 환경설정 정보를 담고 있는 파일입니다.

HSUpSetEnv.exe (업데이트 환경 파일 생성 프로그램)

HSUpdate.env 업데이트 환경 설정 파일을 생성하는 툴로서 업데이트 서버에 대한 환경 설정을 할 수 있습니다.

noupdate.ui (업데이트 비활성화(OFF) 설정 파일)

noupdate.ui 파일은 업데이트를 일시적으로 비활성화(OFF) 시키기 위하여 사용되는 파일입니다.

업데이트를 사용하던 중 부득이하게 업데이트 기능을 비활성화 해야 하는 경우, 서비스 중인 업데이트 서버에서 핵셴드 패치셋 하위(ahn.ui, autoup.exe 등과 동일한 위치)에 noupdate.ui 파일을 위치시키면 핵셴드 업데이트 기능이 비활성화(OFF) 됩니다.

업데이트 기능을 정상적으로 활성화(ON) 시키려면 서비스 중인 업데이트 서버에서 핵셴드 패치셋 폴더 하위(ahn.ui, autoup.exe 등과 동일한 위치)에 위치시킨 noupdate.ui 파일을 삭제 하시면 됩니다.

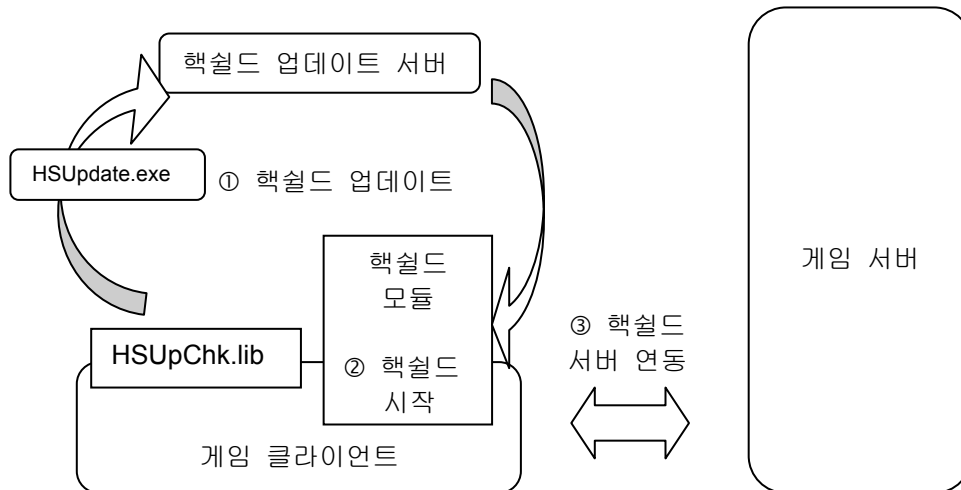


그림 3-1 해설드 업데이트 동작원리

- ① HSUpchk.lib 에서 제공되는 API_AhnHS_HSUpdate 를 호출하여 해설드 업데이트 서버에 접속을 하여 해설드 모듈 및 엔진을 업데이트를 받습니다.
- ② HShield.lib 에서 제공되는 API_AhnHS_Initialize, _AhnHS_Start 함수를 호출하여 해설드를 실행시킵니다.
- ③ 게임 서버에 접속하여 서버연동을 시작합니다.

3.2. Application Programming

HSUpChk.lib 에서 제공하는 API를 이용해서 해설드 업데이트를 구현하는 방법을 설명합니다.

참고

이 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

프로그래밍 적용 방법

HSUpChk.lib 를 사용하여 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

3.2.1.1.업데이트 관련 파일

업데이트 관련 파일

표 3-1 업데이트 관련 파일

파일 이름	설치 폴더	설명
HSUpChk.lib	[프로그램 소스 폴더]	서버에서 사용할 헤더 파일
HSUpChk.h	[프로그램 소스 폴더]	서버에서 사용할 DLL 파일
AhnUpCtl.dll	[Game]\HShield	업데이트 dll
AhnUpGS.dll	[Game]\HShield	업데이트 dll
AspINet.dll	[Game]\HShield	업데이트 dll
Bz32Ex.dll	[Game]\HShield	압축 dll
HSInst.dll	[Game]\HShield	UI dll
HSUpdate.env	[Game]\HShield	업데이트 환경 설정 파일
HSUpdate.exe	[Game]\HShield	업데이트 실행 파일
mspatcha.dll	[Game]\HShield	MS patch dll
V3Hunt.dll	[Game]\HShield	업데이트 dll

표 3-2 업데이트 서버에 설치되어야 하는 파일 목록

PatchSet 경로 : [HackShield SDK] \PatchSet\

파일 이름	설명
ahn.ui	업데이트 정보 파일 (엔진 파일 버전 정보)
ahn.uic	업데이트 정보 파일 (엔진 파일 버전 정보)
ahni2.dll	업데이트 파일
ahnupctl.dll	업데이트 파일
autoup.exe	업데이트 파일
v3bz32.dll	업데이트 파일
patch\39\3n.mh-	휴리스틱 엔진 파일 (압축된 파일)
patch\39\ahn.ui	업데이트 정보 파일 (패치 파일 버전 정보)
patch\39\ahn.uic	업데이트 정보 파일 (패치 파일 버전 정보)
patch\39\ahnupctl.dl-	업데이트 파일 (압축된 파일)
patch\39\ahnupgs.dl-	업데이트 파일 (압축된 파일)
patch\39\aspinet.dl-	업데이트 파일 (압축된 파일)
patch\39\bz32ex.dl-	업데이트 파일 (압축된 파일)
patch\39\ehsvc.dl-	HackShield 인터페이스 DLL (압축된 파일)
patch\39\hshield.da-	해킹 관련 dat 파일 (압축된 파일)
patch\39\hsinst.dl-	업데이트 파일 (압축된 파일)
patch\39\hsupdate.ex-	업데이트 실행 파일 (압축된 파일)
patch\39\mspacha.dl-	업데이트 파일 (압축된 파일)
patch\39\psapi.dl-	Process Status Helper DLL (압축된 파일)
patch\39\v3hunt.dl-	업데이트 파일 (압축된 파일)
patch\39\v3inetgs.dl-	업데이트 파일 (압축된 파일)
win\elb\v3_echo_sl\v3pr	해킹 툴 탐지 엔진 인터페이스 DLL (압축된 파일)

o32s.dl-	
win\elb\b_sign_s\v3warp ds.v3-	해킹 툴 패턴 엔진 파일 (압축된 파일)
win\elb\b_sign_s\v3warp ns.v3-	해킹 툴 패턴 엔진 파일 (압축된 파일)

표 3-3 업데이트에 사용 되는 데이터 파일

Data 파일 경로 : [HackShield SDK] \ Data\

파일 이름	설명
noupdate.ui	업데이트 기능을 일시적으로 비활성화(OFF) 시키기 위해 필요한 설정 파일

업데이트 이후 자동으로 생성되는 파일은 아래 표와 같습니다.

표 3-3 핵실드 업데이트 이후 자동 생성 파일

파일 이름	설치 폴더	설명
ahn.ui	[HackShield 폴더] /Update/	업데이트 정보 파일
ahn.uic	[HackShield 폴더] /Update/	업데이트 정보 파일
ahni2.dll	[HackShield 폴더] /Update/	업데이트 DLL
ahnupctl.dll	[HackShield 폴더] /Update/	업데이트 DLL
autoup.exe	[HackShield 폴더] /Update/	업데이트 실행 파일 (파일 패치 처리)
v3bz32.dll	[HackShield 폴더] /Update/	업데이트 DLL (파일 압축 처리)

3.2.1.2. 적용 방법

서버 운영 방법

핵실드 업데이트 서버는 퍼블리셔(지역별)측에 준비 및 셋팅, 운영이 필요합니다.

1. IIS 설정 등을 통하여 FTP나 HTTP로 외부에서 접근 가능한 업데이트 서버를 구성합니다.

참고

FTP의 경우, 별도의 계정을 설정하거나 anonymous 접근이 가능하도록 설정합니다.

FTP의 경우, **passive mode** 로 접속 가능하도록 설정해야 합니다. **passive mode** 접속 여부를 확인하기 위해서 IE를 통해 FTP 접근 후 파일 목록이 보이는지 확인해주시기 바랍니다.

2. [HackShield SDK] \PatchSet 폴더에 있는 전체 파일을 서버에 업로드 합니다.

파일 목록은 표 3-2 와 같습니다.

주의

패치셋은 서버에 root 폴더 아래 하위 폴더를 하나 생성하여 패치셋을 업로드해야 합니다.

주의

업데이트 주소설정시 다음 사항을 유의해 주십시오.

업데이트 도메인주소란에 ahnlab이라는 명칭이 들어가지 않게 설정 부탁드립니다. 특정 웜바이러스가 도메인주소란에 ahnlab이 포함되어 있으면 접속을 제한하는 웜이 리포트 된 적이 있습니다.

예 : 업데이트 주소를 ahnlabGame.co.kr/real/이라고 표시한 경우, 주소란에 ahnlab이 있음으로 해당 HTTP사이트로 접속불가.

클라이언트 적용 방법

1. HSupChk.lib 라이브러리 파일을 작업할 프로젝트에 포함시킵니다.
2. 제공되는 HSupChk.h 파일을 작업할 소스 파일에 포함시킵니다.

참고

핵실드 업데이트 API 는 두가지 입니다. 두개의 API 중 하나를 선택해서 적용하실 수 있습니다.

AhnHSUpdate : 핵실드 기본 업데이트 기능

AhnHSUpdateEx : 핵실드 기본 업데이트 기능 + Env 파일 교체 방지 기능 + Host File Check 기능

3. 업데이트 API (_AhnHS_HSupDateEx 또는 _AhnHS_HSupDate) 를 사용합니다.

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

// 핵실드 폴더 위치를 지정합니다.
_tcscat ( szFullFilePath, _T( "\\HShield" ) );

// AhnHS_HSupDate 함수 호출
```

```

dwRet = _AhnHS_HSUpdateEx( szFullFilePath, // 핵실드 폴더 경로
                           1000 * 600,    // 업데이트 전체 타임 아웃
                           1234,          // 게임 코드
                           AHNHSUPDATE_CHKOPT_HOSTFILE|
                           AHNHSUPDATE_CHKOPT_GAMECODE,
                           1000* 20 ); // 서버 연결 타임아웃

// Ex 함수를 이용하실때는 반드시 HSUpSetEnv.exe 설정 툴로 env 파일에
// 게임 코드를 입력하셔야 합니다.
if ( dwRet != ERROR_SUCCESS)
{
    // 에러 처리
    switch ( dwRet )
    {
        case HSERROR_ENVFILE_NOTREAD:
            ExitClient();
            break;
        case HSERROR_ENVFILE_NOTWRITE:
            ExitClient();
            break;
        case HSERROR_NETWORK_CONNECT_FAIL:
            ExitClient();
            break;
        case HSERROR_HSUPDATE_TIMEOUT:
            ExitClient();
            break;
        case HSERROR_MISMATCH_ENVFILE:
            ExitClient();
            break;
        case HSERROR_HOSTFILE_MODIFICATION:
            ExitClient();
            break;

        ...
    }
}

```

```

DWORD dwRet = 0;
TCHAR szFullFilePath[MAX_PATH];

// 핵실드 폴더 위치를 지정합니다.
_tcscat ( szFullFilePath, _T( "\\HShield" ) );

// _AhnHS_HSUpdate 함수 호출
dwRet = _AhnHS_HSUpdate ( szFullFilePath, // 핵실드 폴더 경로
                           1000 * 600,    // 업데이트 전체 타임아웃
                           1000* 20 );    // 서버 연결 타임아웃
if ( dwRet != ERROR_SUCCESS)
{
    // 에러 처리
    switch ( dwRet )
    {
        case HSERROR_ENVFILE_NOTREAD:
            ExitClient();
            break;
        case HSERROR_ENVFILE_NOTWRITE:

```

```

        ExitClient();
        break;
    case HSERROR_NETWORK_CONNECT_FAIL:
        ExitClient();
        break;
    case HSERROR_HSUPDATE_TIMEOUT:
        ExitClient();
        break;
    ...
}
}

```

4. [HackShield SDK]\Bin\Update 폴더의 파일 전부를 배포할 클라이언트의 [Game Directory]\HShield 폴더로 복사합니다.
5. [Game Directory]\HShield\ 폴더 내에 Update 폴더를 생성하고 [HackShield SDK]\Bin\Update 폴더에 있는 ahn.ui, ahn.uic, ahni2.dll, ahnupctl.dll, autoup.exe, v3bz32.dll 파일들을 복사합니다.

복사 후 폴더 파일 구조는 다음과 같습니다.

예제 - 핵실드 업데이트 모듈 사용 시 설치 디렉터리 구조

```

[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
v3warpsd.v3d
v3warpsns.v3d

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

```

6. [HackShield SDK]\Util\HSUpSetEnv.exe 파일을 실행합니다.

주의

HSUpSetEnv.exe 는 업데이트 설정파일을 생성하는 툴입니다. 이 툴은 배포되어선 안됩니다

7. [9.5 HSUpSetEnv 툴 사용방법](#)을 참고하여 HSUpdate.env 파일을 생성합니다.

8. HSUpdate.env 파일을 [GameDirecotry]\HShield 폴더로 복사합니다.

주의

업데이트가 정상적으로 실행되기 위해서는 _AhnHS_HSUpdate 함수에 첫번째로 전달되는 파라미터([Game Directory]\HShield)에 [HackShield SDK]\WBin\WUpdate 에 있는 업데이트 모듈이 모두 있어야 합니다.

9. **Splash** 이미지를 사용할 경우에는 사용할 이미지 파일명을 splash.jpg 로 변경한 후에 [Game Directory]\HShield 폴더에 복사해 둡니다.

10. 업데이트시 화면 우측 하단에 보이는 **HShield Update** 이미지를 변경할 경우에는 사용할 이미지 파일명을 hsupdate.jpg 로 변경한 후 [Game Directory]\HShield 폴더에 복사해 둡니다.

디폴트 업데이트 이미지 사이즈: **200 * 149**이며 이미지 하단에 프로그레스 바가 생기므로 업데이트 이미지 생성시 참고하도록 합니다.



그림 3-2 디폴트 핵셴드 업데이트 이미지

참고

1. 업데이트 이미지

— 파일명: hsupdate.jpg (not case sensitive)

— 사이즈: 200 * 149

— 복사할 위치: [Game Directory]\HShield

— 설명: 핵셴드 폴더내에 hsupdate.jpg 가 존재한다면 해당 그림파일을 로드하여 업데이트 이미지로 사용하게 됩니다. 만약 존재하지 않다면, 디폴트 이미지를 사용합니다.

2. 스플래쉬 이미지

— 파일명: splash.jpg (not case sensitive)

— 사이즈: 290 * 190

— 복사할 위치: [Game Directory]\HShield

— 설명: 핵셴드 폴더내에 splash.jpg 가 존재한다면 해당 그림파일을 로드하여 화면 정 가운데에 스플래쉬 이미지로 사용하게 됩니다. 만약

존재하지 않다면, 아무것도 보여주지 않습니다.

3.3. Application Programming Interface

AhnHS_HSUpdateEx

DESCRIPTION

HackShield 파일들을 업데이트합니다. HsUpdate.env 파일 과 호스트 파일 체크 기능이 있습니다.

SYNTAX

```
DWORD __stdcall  
_AhnHS_HSUpdateEx(  
    LPCTSTR szUpdateDir,  
    DWORD dwTimeout,  
    INT64 i64GameCode,  
    DWORD dwOption,  
    DWORD dwTimeoutPerConnection = 0  
);
```

PARAMETERS

Parameter	Value	Description
szUpdateDir	LPCTSTR	업데이트 파일이 설치되어 있는 폴더
dwTimeout	DWORD (milliseconds)	업데이트 시 타임 아웃 시간. 0으로 설정 시 INFINITE으로 설정 * 네트워크 사정에 따라 600000(10분) 내외의 값으로 설정하는 것이 좋으며 가급적이면 0으로는 설정하지 않는 것이 좋습니다.
INT64	i64GameCode	HSUpSetEnv.exe 톨로 HSUpdate.env 파 일에 Game Code 를 저장한 값을 넣어 줍니다.
DWORD	dwOption	업데이트 추가 기능 옵션 AHNHSUPDATE_CHKOPT_HOSTFILE AHNHSUPDATE_CHKOPT_GAMECODE
dwTimeoutPer Connection	DWORD (milliseconds)	서버 연결 시 타임 아웃 시간. * 네트워크 사정에 따라 적절한 값으로 설정하도록 한다. * 0으로 설정 시, 서버 연결 시 타임 아웃 기능이 동작하지 않습니다.

szUpdateDir

[in] szUpdateDir 변수에는 배포되는 핵실드 관련 파일들이 존재하는 절대 경로를 설정합니다. 반드시 해당 폴더에는 업데이트 모듈이 전부 있어야 합니다. 올바른 절대 경로의 설정을 위해서 GetModuleFileName 함수를 이용하는 것이 좋습니다. GetCurrentDirectory 함수는 원하는 경로를 얻지 못할 수도 있습니다.

dwTimeOut

[in] 함수 호출 후 대기 시간을 의미합니다. 두 번째 인자는 millisecond 단위로 되어있으며, 이 대기 시간 동안 응답이 없으면 함수 호출 실패로 간주합니다. 해외와 같이 네트워크 속도가 느린 경우에 대비하여 10분 정도의 대기 시간을 주는 것이 좋습니다. 업데이트 실패 시에는 게임사 정책에 따라 게임 실행 여부를 판단해주시기 바랍니다.

i64GameCode

[in] 업데이트 환경 설정 파일을 임의로 바꾸지 못하도록 (HSUpdate.env) 업데이트 모듈과 맞추주기 위해서 넣어주는 파라미터 입니다. 업데이트 환경 파일에 저장한 게임코드를 넣어주어야 하며, 환경 파일에 저장한 게임코드 값과 다르거나, 환경 파일에 게임코드 값이 없을 경우 HERROR_MISMATCH_ENVFILE 에러를 리턴하게 됩니다.

dwOption

[in] _AhnHS_HSUpdateEx 함수에서 수행하는 기능을 정의해 주는 옵션입니다.

AHNHSUPDATE_CHKOPT_HOSTFILE : Host File 을 검사하여 업데이트 URL 서버 리스트 중 하나라도 IP 지정이 되어 있다면 HERROR_HOSTFILE_MODIFICATION 에러를 리턴합니다. Host file 에 업데이트 URL 정보를 입력하여 업데이트를 우회하는 것을 차단합니다.

AHNHSUPDATE_CHKOPT_GAMECODE : HSUpdate.env 파일에 저장된 게임코드와 parameter 로 전달된 i64GameCode 를 비교하여 다른 HERROR_MISMATCH_ENVFILE 에러를 리턴합니다.

dwTimeOutPerConnection

[in] 서버에 연결 시 대기 시간을 의미합니다. 세 번째 인자는 milliseconds 단위로 되어 있으며, 이 대기 시간 동안 응답이 없으면 현재 접속하려는 서버로의 연결은 실패로 처리됩니다.

HSUpSetEnv.exe 를 이용하여 HSUpdate.env 파일에 한 개 이상의 서버를 설정한 경우, 각각의 서버들에 대해서 dwTimeOutPerConnection 값이 동일하게 적용됩니다.

'0'으로 설정 시에는 서버에 연결 시 대기 시간을 체크하지 않습니다.

RETURN VALUE

HACKSHIELD_ERROR_SUCESS (Value = 0x00000000)

- 설 명 : 업데이트 성공
- 원 인 : 정상적인 경우 발생합니다.
- 확인사항 :

HSERROR_ENVFILE_NOTREAD (Value = 0x30000010)

- 설 명 : HSUpdate.env 파일을 읽을 수 없습니다
- 원 인 : env 환경파일이 없거나 HSUpSetEnv.exe 툴 버전이 맞지 않은 것을 사용하면 발생합니다.
- 확인사항 :
 - ① 해당 위치에 env 파일이 존재하는지 확인해 봅니다.
 - ② HSUpSetEnv.exe 툴이 sdk 내에 존재하는 최신 버전인지 확인해 봅니다.

HSERROR_ENVFILE_NOTWRITE (Value = 0x30000020)

- 설 명 : HSUpdate.env 파일을 쓸 수 없습니다.
- 원 인 : HSUpdate.env 파일을 생성할 때 읽기 속성 또는 접근 권한이 없는 경우 발생합니다.
- 확인사항 :
 - ① env 환경 파일의 속성이 접근 권한이 있는지 확인해 봅니다.
 - ② 환경 파일을 삭제하여 새로 생성했을 경우 위와 같은 문제가 재발생 하는지 확인해 봅니다.

HSERROR_NETWORK_CONNECT_FAIL (Value = 0x30000030)

- 설 명 : 해당 서버에 연결할 수 없습니다.
- 원 인 : 업데이트 서버 (ftp/http) 에 접속할 수 없는 경우 발생합니다
- 확인사항 :
 - ① 업데이트 서버가 정상적으로 접근되는지 ftp 툴 또는 IE 를 통하여 접속을 확인해 봅니다.
 - ② Env 환경파일에 업데이트 주소 및 ID&PW 입력이 잘되었는지 확인해 봅니다.

HSERROR_LIB_NOTEDIT_REG (Value = 0x30000050)

- 설 명 : 네트워크 결과 입력 중에 오류가 발생하였습니다.
- 원 인 :
- 확인사항 :

HSERROR_NOTFINDFILE (Value = 0x30000060)

- 설 명 : HackShield 업데이트 프로그램 관련 파일을 찾을 수 없습니다.
- 원 인 :
- 확인사항 :

HSERROR_PROTECT_LISTLOAD_FAIL (Value = 0x30000070)

- 설명 : HSUpdate.pt 인증 파일을 찾을 수 없습니다.
- 원인 :
- 확인사항 :

HSERROR_HSUPDATE_TIMEOUT (Value = 0x30000090)

- 설명 : HackShield 업데이트 프로그램이 업데이트 전송 시간을 초과하였습니다.
- 원인 : _AhnHS_HSUpdate 함수를 호출할 경우 타임 아웃을 인자로 설정해 줄 수 있습니다. 이 시간안에 업데이트가 이루어지지 않은 경우 에러가 발생합니다. 네트워크 상황이 안좋은 경우 발생할 수 있습니다.
- 확인사항 :

HSERROR_MISMATCH_ENVFILE (Value = 0x300000C0)

- 설명 : Game Client 와 업데이트 환경 설정 파일이 짝이 맞지 않습니다.
- 원인 : _AhnHS_HSUpdateEx 함수를 호출할 때 전달한 i64GameCode 값과 HSUpdate.env 파일에 저장된 GameCode 값과 다르거나, HSUpdate.env 파일에 게임코드가 저장되어 있지 않다면, 발생합니다.
- 확인사항 : 파라미터로 전달된 i64GameCode 와 HSUpdate.env 파일에 저장된 게임코드가 동일한지 확인합니다.
AHNHSUPDATE_CHKOPT_GAMECODE 옵션으로 해당 기능을 ON/OFF 할 수 있습니다.

HSERROR_HOSTFILE_MODIFICATION (Value = 0x300000D0)

- 설명 : Hosts File 에 핵샐드 업데이트 URL 이 지정되어 있습니다.
- 원인 : HSUpdate.env 파일에 저장된 서버 주소 리스트 중에 하나라도 hosts 파일에 IP 지정이 되어 있다면, 감지하게 됩니다.
- 확인사항 : hosts 파일에 핵샐드 업데이트 URL 이 IP 지정이 되어 있는지 확인합니다. AHNHSUPDATE_CHKOPT_HOSTFILE 옵션으로 해당 기능을 ON/OFF 할 수 있습니다.

HSERROR_AUTOUPDATE_FAIL (Value = 0x300000E0)

- 설명 : HackShield 자동 업데이트 프로세스 중 문제가 발생하였습니다.
- 원인 : 업데이트 내부 에러로 인하여 발생할 수 있습니다.
안철수연구소 기술지원에 문의하시기 바랍니다.
- 확인사항 :

REMARKS

업데이트 함수는 반드시 초기화 함수(_AhnHS_Initialize)를 호출하기 전에 호출해야 합니다. 다른 HackShield 함수와 별도로 게임 프로세스를 실행시키는 런처와 같은 실행 파일에서 호출해서 사용할 수 있습니다.

AhnHS_HSUpdate

DESCRIPTION

HackShield 파일들을 업데이트합니다.

SYNTAX

```
DWORD __stdcall  
_AhnHS_HSUpdate(  
    LPCTSTR szUpdateDir,  
    DWORD dwTimeOut,  
    DWORD dwTimeOutPerConnection = 0  
);
```

PARAMETERS

Parameter	Value	Description
szUpdateDir	LPCTSTR	업데이트 파일이 설치되어 있는 폴더
dwTimeOut	DWORD (milliseconds)	업데이트 시 타임 아웃 시간. 0으로 설정 시 INFINITE으로 설정 * 네트워크 사정에 따라 600000(10분) 내외의 값으로 설정하는 것이 좋으며 가급적이면 0으로는 설정하지 않는 것이 좋습니다.
dwTimeOutPer Connection	DWORD (milliseconds)	서버 연결 시 타임 아웃 시간. * 네트워크 사정에 따라 적절한 값으로 설정하도록 한다. * 0으로 설정 시, 서버 연결 시 타임 아웃 기능이 동작하지 않습니다.

szUpdateDir

[in] szUpdateDir 변수에는 배포되는 핵실드 관련 파일들이 존재하는 절대 경로를 설정합니다. 반드시 해당 폴더에는 업데이트 모듈이 전부 있어야 합니다. 올바른 절대 경로의 설정을 위해서 GetModuleFileName 함수를 이용하는 것이 좋습니다. GetCurrentDirectory 함수는 원하는 경로를 얻지 못할 수도 있습니다.

dwTimeOut

[in] 함수 호출 후 대기 시간을 의미합니다. 두 번째 인자는 millisecond

단위로 되어있으며, 이 대기 시간 동안 응답이 없으면 함수 호출 실패로 간주합니다. 해외와 같이 네트워크 속도가 느린 경우에 대비하여 10분 정도의 대기 시간을 주는 것이 좋습니다. 업데이트 실패 시에는 게임사 정책에 따라 게임 실행 여부를 판단해주시기 바랍니다.

`dwTimeOutPerConnection`

[in] 서버에 연결 시 대기 시간을 의미합니다. 세 번째 인자는 `milliseconds` 단위로 되어 있으며, 이 대기 시간 동안 응답이 없으면 현재 접속하려는 서버로의 연결은 실패로 처리됩니다.

`HSUpSetEnv.exe` 를 이용하여 `HSUpdate.env` 파일에 한 개 이상의 서버를 설정한 경우, 각각의 서버들에 대해서 `dwTimeOutPerConnection` 값이 동일하게 적용됩니다.

'0'으로 설정 시에는 서버에 연결 시 대기 시간을 체크하지 않습니다.

RETURN VALUE

HACKSHIELD_ERROR_SUCESS (Value = 0x00000000)

- 설 명 : 업데이트 성공
- 원 인 : 정상적인 경우 발생합니다.
- 확인사항 :

HSERROR_ENVFILE_NOTREAD (Value = 0x30000010)

- 설 명 : `HSUpdate.env` 파일을 읽을 수 없습니다
- 원 인 : `env` 환경파일이 없거나 `HSUpSetEnv.exe` 툴 버전이 맞지 않은 것을 사용하면 발생합니다.
- 확인사항 :
 - ③ 해당 위치에 `env` 파일이 존재하는지 확인해 봅니다.
 - ④ `HSUpSetEnv.exe` 툴이 `sdk` 내에 존재하는 최신 버전인지 확인해 봅니다.

HSERROR_ENVFILE_NOTWRITE (Value = 0x30000020)

- 설 명 : `HSUpdate.env` 파일을 쓸 수 없습니다.
- 원 인 : `HSUpdate.env` 파일을 생성할 때 쓰기 속성 또는 접근 권한이 없는 경우 발생합니다.
- 확인사항 :
 - ③ `env` 환경 파일의 속성이 접근 권한이 있는지 확인해 봅니다.
 - ④ 환경 파일을 삭제하여 새로 생성했을 경우 위와 같은 문제가 재발생 하는지 확인해 봅니다.

HSERROR_NETWORK_CONNECT_FAIL (Value = 0x30000030)

- 설 명 : 해당 서버에 연결할 수 없습니다.
- 원 인 : 업데이트 서버 (ftp/http) 에 접속할 수 없는 경우 발생합니다
- 확인사항 :
 - ③ 업데이트 서버가 정상적으로 접근되는지 ftp 툴 또는 IE 를 통하여 접속을 확인해 봅니다.
 - ④ Env 환경파일에 업데이트 주소 및 ID&PW 입력이 잘되었는지 확인해 봅니다.

HSERROR_LIB_NOTEDIT_REG (Value = 0x30000050)

- 설 명 : 네트워크 결과 입력 중에 오류가 발생하였습니다.
- 원 인 :
- 확인사항 :

HSERROR_NOTFINDFILE (Value = 0x30000060)

- 설 명 : HackShield 업데이트 프로그램 관련 파일을 찾을 수 없습니다.
- 원 인 :
- 확인사항 :

HSERROR_PROTECT_LISTLOAD_FAIL (Value = 0x30000070)

- 설 명 : HSUpdate.pt 인증 파일을 찾을 수 없습니다.
- 원 인 :
- 확인사항 :

HSERROR_HSUPDATE_TIMEOUT (Value = 0x30000090)

- 설 명 : HackShield 업데이트 프로그램이 업데이트 전송 시간을 초과하였습니다.
- 원 인 : _AhnHS_HSUpdate 함수를 호출할 경우 타임 아웃을 인자로 설정해 줄 수 있습니다. 이 시간안에 업데이트가 이루어지지 않은 경우 에러가 발생합니다. 네트워크 상황이 안좋은 경우 발생할 수 있습니다.
- 확인사항 :

HSERROR_AUTOUPDATE_FAIL (Value = 0x300000E0)

- 설 명 : HackShield 자동 업데이트 프로세스 중 문제가 발생하였습니다.
- 원 인 : 업데이트 내부 에러로 인하여 발생할 수 있습니다.

안철수연구소 기술지원에 문의하시기 바랍니다.

- 확인사항 :

REMARKS

업데이트 함수는 반드시 초기화 함수(`_AhnHS_Initialize`)를 호출하기 전에 호출해야 합니다. 다른 **HackShield** 함수와 별도로 게임 프로세스를 실행시키는 런처와 같은 실행 파일에서 호출해서 사용할 수 있습니다.

4. 확장 서버 연동 크랙 방지 기능

4.1. 개요

확장 서버 연동(AntiCpX)은 파일/메모리 조작 감지 기능을 제공한 (구) 서버 연동 SDK(Software Development Kit)를 한단계 발전시킨 SDK입니다. 가장 크게 향상된 점은 (구) 서버 연동에서 함수 단위로 메모리 상에 로드된 코드를 보호했던 것에 반해서 확장 서버 연동에서는 코드 영역 전체를 보호합니다.

기능

확장 서버 연동(AntiCpX)은 (구) 서버 연동 기능을 포함하며 ‘메모리 무결성 기능’이 강화되었습니다.

게임 클라이언트 파일 무결성 검증

서버에서 AntiCpXSvr.dll과 AntiCpXSvr.h를 적용하고, 클라이언트에서는 핵셸드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 실시간으로 게임 실행 파일의 조작 여부를 확인할 수 있습니다.

주의

게임클라이언트가 바이러스에 감염되는 경우에도 확장서버연동에 의하여 감지될 수 있습니다. 이에 게임사의 정책에 맞게 해당콜백에 대한 종료처리를 부탁드립니다.

패킷 무결성 기능

패킷을 캡처하고 특정 상황에 맞추어 패킷을 생성해서 정상적으로 동작하게 하는 해킹을 막기 위해, 패킷의 무결성을 보장하는 모듈이 내부적으로 동작합니다. 네트워크에서 패킷을 캡처하고 생성하는 것과 같은 방법을 원천적으로 차단할 수 있습니다. 단, 이 기능은 서버 연동 크랙 방지와 관련된 메시지들에 한하여 보호합니다.

핵셸드 정상동작 확인

서버에서 AntiCPXSvr.dll과 AntiCpXSvr.h를 적용하고, 클라이언트에서는 핵셸드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 간단한 방법으로 핵셸드의 동작 상태를 확인할 수 있습니다. 따라서 서버는 핵셸드가 동작한 다음에 쿼리 메시지를 전송해야 합니다.

메모리 무결성 기능 (확장서버연동 강화 기능)

서버에서 AntiCpXSvr.dll과 AntiCpXSvr.h를 적용하고, 클라이언트에서는 핵셸드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 실시간으로 게임 프로세스의 메모리 조작 여부를 검증할 수 있습니다. 확장

서버 연동에서는 기능이 강화되어 (구) 서버 연동에서 함수 단위로 메모리 상의 코드를 보호했던 것에 반해서 코드 영역 전체를 보호합니다.

핵심드 엔진 무결성 확인

서버에서 AntiCPXSvr.dll과 AntiCpXSvr.h를 적용하고, 클라이언트에서는 핵심드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 휴리스틱 엔진 (3N.mhe) 파일의 무결성을 검증할 수 있습니다.

특징

인터페이스 함수(API) 제공

확장 서버 연동(AntiCpX)에서 제공하는 기능을 편리하게 사용하고, 그 실행 결과 값을 받아 볼 수 있도록 인터페이스 DLL과 라이브러리를 제공합니다. 제공되는 인터페이스 DLL과 라이브러리를 사용하여 개발자는 고유 정책에 따라서 게임 파일의 무결성 및 핵심드의 정상 동작 유무를 확인할 수 있습니다.

HSB데이터 파일정보 생성 프로그램 제공

확장 서버 연동(AntiCpX)에서 제공하는 HSBGen.exe는 클라이언트와 서버가 통신하여 클라이언트 프로그램의 파일/메모리 무결성 여부를 판단하는 기준이 되는 각종 파일 정보와 메모리 정보를 생성하고 저장합니다. 게임 개발사의 특성에 맞게 서버에서 파일/메모리 정보를 저장할 수 있으며, 이 데이터를 이용하여 서버 연동을 통한 파일/메모리 크랙 방지 기능이 동작하게 됩니다. 자세한 사용 방법은 '9.3 HSBGen 툴(HackShield 전용, 4.2 이후 버전)'에서 설명합니다.

시스템 구조(System Architecture)

확장 서버 연동(AntiCpX)은 AntiCpXSvr.dll, HShield.lib, EHSvc.dll을 제공하여 서버와 클라이언트에 DLL과 라이브러리 형태로 적용됩니다. 확장 서버 연동(AntiCpX)의 전체 구조 및 동작 원리는 다음과 같습니다.

AntiCPXSvr.dll (인터페이스 dll)

서버에서 사용되며 서버에서 요청 메시지를 생성하고 클라이언트에서 받은 메시지를 이용하여 클라이언트 파일 및 메모리가 정상 동작하는지 확인할 수 있게 하는 API를 제공합니다.

HShield.lib (핵심드 라이브러리)

클라이언트에서 사용되며 서버에서 전송한 요청 메시지를 받아 요구한 내용을 수행하고 결과로 얻은 각종 정보를 암호화하여 서버에 전송할 데이터를 만드는 API를 제공합니다.

HSBGen.exe (파일 정보 생성 프로그램)

서버에서 사용할 파일 정보 및 메모리 정보 파일을 생성하여 게임 파일의 무결성 여부와 실행되는 게임의 메모리 무결성 여부를 판단할 수 있게 합니다. 파일 정보 데이터는 생성할 때 마다 매번 다른 값을 저장하므로, 게임 서버가

여러 개이면 서버마다 다르게 적용시켜야 높은 보안을 유지할 수 있습니다.

AntiCpx.hsb (클라이언트 CRC 파일)

HSBGen.exe 툴에서 생성되는 파일로서, 클라이언트 프로그램에 대해 무결성을 보장할 수 있도록 합니다.

HSPub.key (서버 연동 인증 키 파일)

접속되는 클라이언트의 해쉬드를 인증하는 용도로 사용되는 파일로서, AntiCpx.hsb 파일과 동일한 위치에 있어야 합니다.

3N.mhe (휴리스틱 엔진 파일)

클라이언트에서 사용하는 엔진파일로서, 서버연동 사용 시 이전 버전의 3n.mhe 엔진을 사용하는 클라이언트의 접속을 끊을 수 있도록 서버의 AntiCpX.hsb 파일과 동일한 위치에 놓아야 합니다.

HShield.dat (해쉬드 버전 파일)

클라이언트에서 사용하는 데이터파일로서, 서버연동 사용 시 이전 버전의 hshield.dat파일을 사용하는 클라이언트의 접속을 끊을 수 있도록 서버의 AntiCpX.hsb 파일과 동일한 위치에 놓아야 합니다.

참고

서버 hsb 파일 경로에 3N.mhe 또는 HShield.dat 파일을 놓아두면 이전 버전의 해쉬드로 접속하는 클라이언트에 대해서 접속을 끊을 수 있습니다. (3N.mhe & HShield.dat 파일을 올려 놓을 때는 서버의 재기동이 필요없습니다.)

아래 표와 같이 “접속 끊음”으로 표시되는 경우에 한해서 _AhnHS_VerifyResponseEx 함수에서 ANTICPX_RECOMMEND_CLOSE_SESSION (세부 예러: ERROR_ANTICPXSVR_INVALID_ENGINE_VERSION/ERROR_ANTICPXSVR_INVALID_HACKSHIELD_VERSION) 예러가 리턴됩니다.

해당 예러가 리턴될 경우 클라이언트 접속을 끊으시면 됩니다.

표 4-1 서버연동 버전관리

파일 명	버전 비교	접속상태
3N.mhe	Client > Server	접속 끊음
	Client ≤ Server	접속 유지
HShield.dat	Client > Server	접속 끊음
	Client ≤ Server	접속 유지

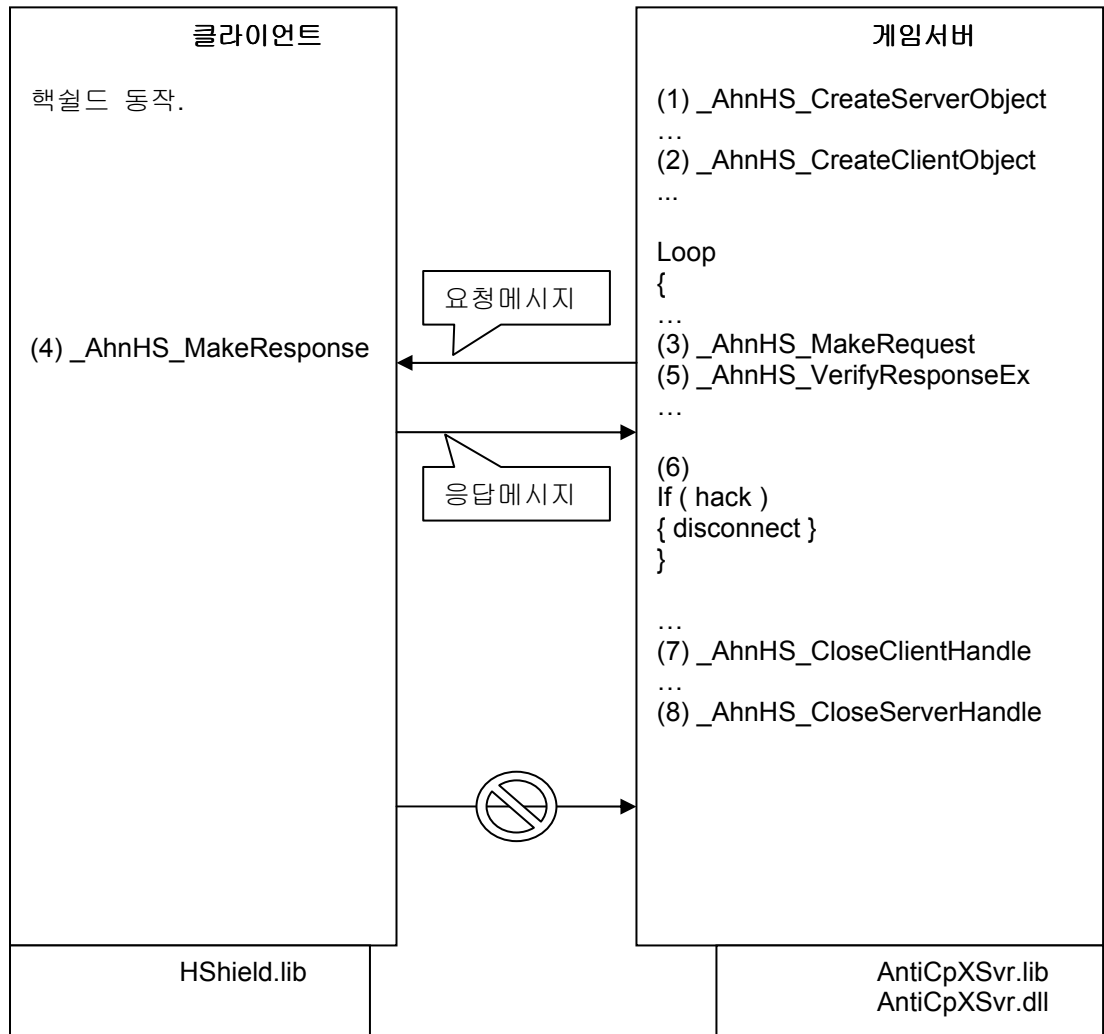


그림 4-1 AntiCpX의 동작 원리

게임 서버가 최초 실행될 때 `_AhnHS_CreateServerObject` 함수를 통해 `AHNHS_SERVER_HANDLE`를 생성합니다. 이후 각 클라이언트가 접속 될 때 마다 `_AhnHS_CreateClientObject` 함수와 서버 핸들을 파라미터로 취하여 `AHNHS_CLIENT_HANDLE`를 생성합니다. 클라이언트의 위변조 여부를 감시하기 위해 `_AhnHS_MakeRequest` 함수를 사용하여 요청 메시지를 주기적으로 생성하여 전송합니다.

클라이언트는 서버의 요청 메시지에 적절한 응답 메시지를 생성합니다. 응답 메시지는 `_AhnHS_MakeResponse` 함수를 통해 생성합니다. 클라이언트의 응답 메시지가 유효한지를 `_AhnHS_VerifyResponseEx` 함수를 통해 검사합니다.

주의

`_AhnHS_MakeResponse` 함수에 의해 생성된 응답메세지가 서버측으로 전달되어 `_AhnHS_VerifyResponseEx` 함수가 호출되기 전까지

_AhnHS_MakeRequest 함수가 호출되지 않도록 구성합니다.

4.2. Application Programming

확장 서버 연동(AntiCpX)에서 제공하는 API를 이용해서 파일 및 메모리의 무결성 확인을 구현하는 방법을 설명합니다.

참고

이 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

프로그래밍 적용 방법

확장 서버 연동(AntiCpX)을 사용하여 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

4.2.1.1.AntiCpXSvr 관련 파일

AntiCpXSvr 관련 파일

표 4-2 AntiCpXSvr 관련 파일

파일 이름	설치 폴더	설명
AntiCpXSvr.h	[프로그램 소스 폴더]	서버에서 사용할 헤더 파일
AntiCpXSvr.dll	[프로그램 소스 폴더]	서버에서 사용할 DLL 파일
AntiCpXSvr.lib	[프로그램 소스 폴더]	AntiCPXSvr.dll의 임포트 라이브러리
AntiCpx.hsb	[프로그램 실행 폴더]	클라이언트 무결성 검증 파일
HSPub.key	[프로그램 실행 폴더]	서버 연동 인증 키 파일
3N.mhe	[HSB파일과 동일한 폴더]	핵셴드 휴리스틱 엔진 버전 관리 파일
HShield.dat	[HSB파일과 동일한 폴더]	핵셴드 버전 관리 파일
HShield.h	[프로그램 소스 폴더]	클라이언트에서 사용할 헤더 파일, 핵셴드 기능도 포함되어 있음.
HShield.lib	[프로그램 소스 폴더]	클라이언트에서 사용할 라이브러리 파일, 핵셴드 기능도 포함되어 있음.
EHSvc.dll	[프로그램 소스 폴더]	클라이언트에서 사용할 DLL 파일, 핵셴드 기능도 포함되어 있음.

4.2.1.2.적용 방법

서버 적용 방법

1. HSBGen.exe 를 이용하여 AntiCpx.hsb 를 생성합니다.

(10.5.HSBGen 툴 참조)

2. HSPub.key 파일을 AntiCpx.hsb 파일이 존재하는 동일한 폴더에 복사해 둡니다.
3. 3N.mhe 와 HShield.dat 파일을 hsb 파일이 존재하는 폴더에 복사해 둡니다. (이 과정은 필수적인 것은 아니지만 핵셴드 버전관리를 위하여 권장하고 있습니다.)

참고

서버에 hsb가 존재하는 위치에 클라이언트에서 사용하는 3N.mhe 와 HShield.dat 파일을 올려둠으로써 3N.mhe 와 HShield.dat 버전이 올바르게 않은 클라이언트를 접속하지 못하게 관리할 수 있습니다.

4. 게임 서버가 최초 실행될 때 _AhnHS_CreateServerObject 함수를 통해 AHNHS_SERVER_HANDLE을 생성합니다. 이 핸들은 게임 서버가 종료되기 전까지 유지되어야 합니다. (참고: 서버 핸들은 다음 2단계의 클라이언트 핸들을 만드는데 사용됩니다)
5. 각 클라이언트가 접속 될 때 마다 _AhnHS_CreateClientObject 함수와 서버 핸들을 파라미터로 받아 AHNHS_CLIENT_HANDLE을 생성합니다. 이 핸들은 클라이언트 네트워크 접속이 유지하는 세션 동안 유지합니다. (참고: 클라이언트 핸들은 다음 3단계의 요청 메시지를 만드는데 사용됩니다.)
6. 클라이언트의 위변조 여부를 감시하기 위해 요청 메시지를 생성하여 전송합니다. 요청 메시지는 _AhnHS_MakeRequest 함수를 사용하여 생성하며 클라이언트 핸들을 파라미터로 사용합니다.
7. 클라이언트는 서버의 요청메시지에 적절한 응답 메시지를 생성합니다. 응답 메시지는 _AhnHS_MakeResponse 함수를 통해 생성합니다.
8. 클라이언트의 응답 메시지가 유효한지 검사합니다. 응답 메시지의 유효성 검사는 _AhnHS_VerifyResponseEx 함수를 통해 검사합니다.
9. _AhnHS_VerifyResponseEx 함수에서 ANTICPX_RECOMMAND_CLOSE_SESSION 값이 발생했을 경우 인자로 전달받은 에러 값에 따라 적절하게 에러 처리를 수행한 후 게임 클라이언트의 접속을 중단합니다.

주의

(_AhnHS_VerifyResponseEx 와 _AhnHS_VerifyResponse 2개의 함수 중 적용하면 됩니다.)

10. 클라이언트 접속을 끊을 때 (세션이 종료될 때) 2단계에서 생성한 클라이언트 핸들을 닫습니다.
11. 서버 프로세스가 종료될 때 1단계에서 생성한 서버 핸들을 닫습니다. (이때 클라이언트 핸들이 모두 Close된 상태여야 합니다.)

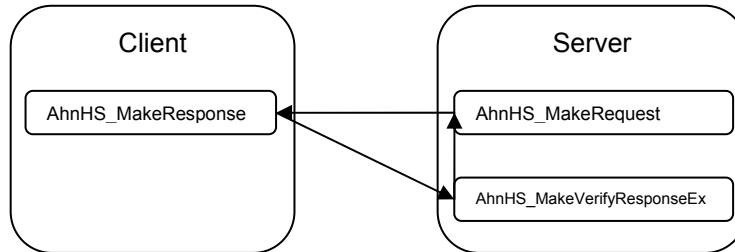
클라이언트 적용 방법

1. HShield.lib 파일을 작업할 프로젝트에 포함시킵니다.
2. 제공되는 HShield.h 파일을 작업할 소스 파일에 포함시킵니다.
3. 서버로부터 암호화된 버전 요청 메시지를 받습니다.
4. _AhnHS_MakeResponse를 호출하여 서버로 전달한 암호화된 버전 응답 메시지를 생성합니다.

5. 암호화된 응답 메시지를 서버에 전송합니다.

서버 연동 주기

주기 설명: 아래 그림과 같이 서버에서 요청 메시지를 보내고 클라이언트에서 응답 메시지를 생성한 후 서버에서 받는 것을 하나의 주기로 설정합니다.



주기 시간: 1주기에 1분이내로 설정하는 것을 권장합니다.

(게임의 특성에 맞추어서 주기 설정이 가능하지만, 서버 연동의 주기가 길어지면 해킹 감지 시간이 길어질 수 있습니다.)

4.3. Application Programming Interface

AhnHS_CreateServerObject

DESCRIPTION

HSBGen.exe를 통해 생성한 .hsb 파일을 로드 하여 서버 핸들(Server Handle)을 생성합니다. 보통 하나의 게임을 서비스하는 서버 프로세스에서 서버 핸들은 한 개를 생성하여 게임 서버 프로세스가 종료할 때까지 유지합니다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
pszFilePath	const char *	HackShield Briefcase (.hsb) 파일 Full Path

RETURN VALUE

서버 핸들을 올바르게 생성하지 못했을 경우 NULL 값을 리턴 합니다. 다음 과 같은 상황에서 발생할 수 있습니다.

- ① HackShield Briefcase (.hsb) 파일 경로가 올바르지 않거나 파일이 없는 경우
- ② HSPub.key 파일이 해당 경로에 존재하지 않을 경우,
- ③ 시스템 리소스(메모리)가 부족할 경우.

ERROR_ANTICPXSVR_FILE_ACCESS_DENIED (Value = 0xE9040001)

- 설 명 : HSB 파일에 접근 중에 오류가 발생하였습니다.
- 원 인 : HSB 파일에 접근 권한이 없습니다.
- 확인사항 : HSB 파일에 접근할 수 있는지 파일 권한(Permission) 혹은 파일 속성(Attributes)을 확인하시기 바랍니다.

ERROR_ANTICPXSVR_FILE_NOT_FOUND (Value = 0xE9040002)

- 설명 : HSB 파일에 접근 중에 오류가 발생하였습니다.
- 원인 : HSB 파일이 `_AhnHS_CreateServerObject` 함수 호출 시 전달받은 경로 내에 존재하지 않습니다.
- 확인사항 : HSB 파일이 존재한 위치 경로를 확인하시기 바랍니다.
`_AhnHS_CreateServerObject` 함수 호출 시 전달한 경로 내에 HSB 파일이 존재해야 합니다.
(Linux/Unix인 경우에는 파일명에 대해서 대소문자를 구분하므로 HSB 파일명을 정확히 기입하시기 바랍니다)

ERROR_ANTICPXSVR_CREATE_SVROBJ_EXCEPTION

(Value = 0xE9040013)

- 설명 : `_AhnHS_CreateServerObject` 함수 수행 중 예외(Exception)가 발생하였습니다.

(본 에러는 Windows용 모듈에서만 반환됩니다)

- 원인 : 여러 가지 원인에 의해서 예외가 발생할 수 있습니다.
- 확인사항 : 예외가 발생한 경우, 내부적으로 MiniDump 파일을 남기도록 되어 있습니다.
'CREATESERVEROBJECT_년_월_일 시-분-초.dmp' 의 파일명 형식으로 덤프 파일이 생성되므로 해당 파일을 (주)안철수연구소 로 전달해 주시기 바랍니다.

Example

`_AhnHS_CreateServerObject` 함수를 호출한 예제는 다음과 같습니다.

```
예제

// 파일 경로는 \'\' 아닌 \'\' 으로 표현해야 합니다.
// 파일 이름까지 경로를 표현해줍니다.

strcpy(g_szHsbFilePath,
       "C:\\GameServer\\HShield\\anticpx.hsb" );

hServer = _AhnHS_CreateServerObject (g_szHsbFilePath);

if ( hServer == ANTICPX_INVALID_HANDLE_VALUE )
{
    // 에러 처리
}
```

AhnHS_CloseServerHandle

DESCRIPTION

서버 핸들(Server Handle)을 닫습니다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
pszFilePath	AHNHS_SERVER_HANDLE	_AhnHS_CreateServerObject 함수를 통해 생성한 핸들

RETURN VALUE

없음.

Example

_AhnHS_CloseServerHandle 함수를 호출한 예제는 다음과 같습니다.

예제

```
_AhnHS_CloseServerHandle ( hServer );
```

AhnHS_CreateClientObject

DESCRIPTION

서버 핸들을 입력 받아 클라이언트 핸들을 생성합니다. 클라이언트 핸들은 클라이언트가 접속할 때 마다 생성하며, 세션이 유지되는 동안 핸들을 닫지 않고 재사용합니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
hServer	AHNHS_SERVER_HANDLE	서버 핸들(Server Handle)

RETURN VALUE

클라이언트 핸들(Client Handle)

ERROR_ANTICPXSVR_CREATE_CLIENT_OBJECT_EXCEPTION (Value = 0xE9040017)

- 설 명 : _AhnHS_CreateClientObject 함수 수행 중 예외(Exception)가 발생하였습니다.

(본 에러는 Windows용 모듈에서만 반환됩니다)

- 원 인 : 여러 가지 원인에 의해서 예외가 발생할 수 있습니다.
- 확인사항 : 예외가 발생한 경우, 내부적으로 MiniDump 파일을 남기도록 되어 있습니다.
'CREATECLIENTOBJECT_년_월_일 시-분-초.dmp' 의 파일명 형식으로 덤프 파일이 생성되므로 해당 파일을 (주)안철수연구소 로 전달해 주시기 바랍니다.

Example

_AhnHS_CreateClientObject 함수를 호출한 예제는 다음과 같습니다.

```
예제  
  
hClient = _AhnHS_CreateClientObject ( hServer );  
  
if ( hClient == ANTICPX_INVALID_HANDLE_VALUE )  
{
```

```
// 에러 처리
```

```
}
```

AhnHS_CloseClientHandle

DESCRIPTION

생성한 클라이언트 핸들은 클라이언트 세션이 종료될 때 해제되어야 합니다. 이때 클라이언트 핸들 생성에 사용되었던 메모리나 시스템 자원을 해제하게 됩니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
hClient	AHNHS_CLIENT_HANDLE	클라이언트 핸들(Client Handle)

RETURN VALUE

없음.

Example

_AhnHS_CloseClientHandle 함수를 호출한 예제는 다음과 같습니다.

예제

```
_AhnHS_CloseClientHandle ( hClient );
```

AhnHS_MakeRequest

DESCRIPTION

현재 세션에 맞는 클라이언트 핸들을 입력하여 요청 메시지를 생성합니다. 요청 메시지는 `AHNHS_TRANS_BUFFER` 구조체 형태로 출력되며, 멤버 변수 값은 다음과 같습니다.

```
typedef struct _AHNHS_TRANS_BUFFER
{
    unsigned short nLength;
    unsigned char byBuffer[ANTICPX_TRANS_BUFFER_MAX]; // 송수신
    패킷의 최대 버퍼 크기
} AHNHS_TRANS_BUFFER, *PAHNHS_TRANS_BUFFER;
```

`nLength` ; 요청 메시지 생성에 사용된 버퍼 길이
`byBuffer` ; 요청 메시지 생성에 사용될 수 있는 최대 바이트 버퍼

주의

`byBuffer`는 요청 메시지 생성에 사용될 수 있는 최대 버퍼 크기이므로 네트워크로 전송할 때 `nLength` 만큼만 전송해야 합니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
<code>hClient</code>	<code>AHNHS_CLIENT_HANDLE</code>	클라이언트 핸들
<code>pRequestBuffer</code>	<code>PAHNHS_TRANS_BUFFER</code>	보낼 데이터 버퍼/길이

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 함수 호출을 성공했을 때 리턴 하는 값입니다.
- 원 인 : 정상적인 경우 입니다.
- 확인사항 :

ERROR_ANTICPXSVR_INVALID_PARAMETER (Value = 0xE9040003)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : hClient, pRequestBuffer 값이 NULL 인 경우 발생합니다.
- 확인사항 : hClient, pRequestBuffer 값이 NULL 이 아닌지 확인합니다.

ERROR_ANTICPXSVR_BAD_FORMAT (Value = 0xE9040004)

- 설 명 : HSB 파일을 읽는데 문제가 발생하였습니다.
- 원 인 : HSB 파일이 정상적으로 생성이 되지 않은 경우 발생합니다.
- 확인사항 : HSBGen.exe Tool 을 최신 버전으로 사용하였는지 확인해 보고 다시 HSB 파일을 생성해 봅니다.

ERROR_ANTICPXSVR_NOT_YET_RECEIVED_RESPONSE (Value = 0xE9040005)

- 설 명 : 이전 요청 메시지에 대한 응답을 받지 않았습니다
- 원 인 : _AhnHS_MakeRequest 함수를 호출하면 클라이언트로부터 Ack 메시지를 받아서 _AhnHS_VerifyResponse 함수를 거쳐서 다시 호출 되어야 합니다. 이 동기화 과정이 정상적으로 이루어지지 않으면 발생 합니다.
- 확인사항 : 클라이언트 세션 관리상의 동기화 문제가 발생하지 않았는지 확인해 봅니다.

ERROR_ANTICPXSVR_NOT_ENOUGH_MEMORY (Value = 0xE9040007)

- 설 명 : 메모리가 부족합니다.
- 원 인 : 서버의 메모리가 부족하여 발생된 에러입니다.
- 확인사항 : 서버에서 메모리 leak 현상이 발생하는지 확인해 보아야 합니다.

ERROR_ANTICPXSVR_BAD_MESSAGE (Value = 0xE9040008)

- 설 명 : 버퍼 암호화에 실패하였습니다.
- 원 인 : 요청 보낼 버퍼를 암호화 하는데 실패한 경우 발생합니다.
- 확인사항 : 위 에러가 발생한다면 구조적인 문제일 수 있으므로 안철수 연구소 기술지원에 문의하시기 바랍니다.

ERROR_ANTICPXSVR_MAKEREQ_EXCEPTION (Value = 0xE9040014)

- 설명 : _AhnHS_MakeRequest 함수 수행 중 예외(Exception)가 발생하였습니다.

(본 에러는 Windwos용 모듈에서만 반환됩니다)

- 원인 : 여러 가지 원인에 의해서 예외가 발생할 수 있습니다.
- 확인사항 : 예외가 발생한 경우, 내부적으로 MiniDump 파일을 남기도록 되어 있습니다.
'MAKEREQUEST_년_월_일 시-분-초.dmp' 의 파일명 형식으로 덤프 파일이 생성되므로 해당 파일을 (주)안철수연구소 로 전달해 주시기 바랍니다.

Example

_AhnHS_MakeRequest 함수를 호출한 예제는 다음과 같습니다.

예제

```
AHNHS_TRANS_BUFFER stReqTransBuf;

ulRet = _AhnHS_MakeRequest ( hClient, &stReqTransBuf );

if ( ulRet != ERROR_SUCCESS )
    return ulRet;

bytesSent = send( ConnectSocket, stReqTransBuf.byBuffer,
stReqTransBuf.nLength, 0 );

...
```

주의

_AhnHS_MakeRequest 함수의 호출은 게임 서버에 게임 사용자가 접속한 이후에 바로 호출되도록 함으로써 해킹 감지가 이루어지도록 하는 게 좋습니다.

AhnHS_VerifyResponseEx

DESCRIPTION

_AhnHS_MakeRequest 함수를 통한 요청(메시지)에 대한 클라이언트의 응답이 올바른지 검사하는 함수입니다.
(내부적으로 _AhnHS_VerifyResponse()함수를 호출하도록 되어 있습니다)

SYNTAX

```
unsigned long __stdcall  
_AhnHS_VerifyResponseEx (  
    IN AHNHS_CLIENT_HANDLE hClient,  
    IN unsigned char *pbyResponse,  
    IN unsigned long nResponseLength,  
    OUT unsigned long *pnErrorCode  
);
```

PARAMETERS

Parameter	Type	Description
hClient	AHNHS_CLIENT_HANDLE	클라이언트 핸들
pbyResponse	char *	클라이언트로 부터 받은 데이터 버퍼
nResponseLength	unsigned long	클라이언트로 부터 받은 데이터 길이
pnErrorCode	Unsigned long	_AhnHS_VerifyResponse()함수에서 반환 값

RETURN VALUE

ANTICPX_RECOMMEND_CLOSE_SESSION (Value = 101)

- 설 명 : 클라이언트에서 해킹 행위가 감지되었으므로 게임 서버에서 해당 게임 클라이언트와의 접속을 종료해야 합니다.
(해킹 행위는 주로 _AhnHS_VerifyResponse()함수에서 설명되어진 에러코드 중에 클라이언트와의 접속을 종료하도록 권장한 에러가 반환되는 경우를 의미함)
- 원 인 : 클라이언트에서 해킹 행위가 감지되었습니다.
- 확인사항 : N/A

ANTICPX_RECOMMEND_KEEP_SESSION (Value = 102)

- 설 명 : 클라이언트에서의 응답이 정상적입니다. 게임 서버에서 해당 게임 클라이언트와의 접속을 계속 진행하시기 바랍니다.
- 원 인 : N/A
- 확인사항 : N/A

Example

_AhnHS_VerifyResponse 함수를 호출한 예제는 다음과 같습니다.

```
예제

DWORD dwError = 0;
ulRet = _AhnHS_VerifyResponseEx ( hClient,
                                   stResponseBuf.byBuffer,
                                   stResponseBuf.nLength,
                                   &dwError );

if ( ulRet == ANTICPX_RECOMMAND_CLOSE_SESSION )
{
    Log ("[AVREx] Disconnect the Client: 0x%x", dwError );
    // Client Out -> _AhnHS_CloseClientHandle 를
    // 호출하여 hClient 핸들을 닫아줌
}
```

AhnHS_VerifyResponse

DESCRIPTION

_AhnHS_MakeRequest 함수를 통한 요청(메시지)에 대한 클라이언트의 응답이 올바른지 검사하는 함수입니다.

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	클라이언트 핸들
pbyResponse	char *	클라이언트로 부터 받은 데이터 버퍼
nResponseLength	unsigned long	클라이언트로 부터 받은 데이터 길이

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 경우 발생합니다.
- 확인사항 :

ERROR_ANTICPSVR_INVALID_PARAMETER (Value = 0xE9040003)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : hClient, pbyResponse 값이 NULL 인 경우 발생합니다.
- 확인사항 : hClient, pbyResponse 값이 NULL 이 아닌지 확인해 봅니다.

ERROR_ANTICPSVR_BAD_FORMAT (Value = 0xE9040004)

- 설 명 : 올바른 포맷이 아닙니다.
- 원 인 : 복호화 과정에서 실패가 일어난 경우 발생합니다.
- 확인사항 : 서버에서 보낸 버퍼가 정상적으로 클라이언트의

`_AhnHS_VerifyResponse` 함수에 전달되었는지 확인해
봅니다.

ERROR_ANTICPXSVR_NO_WAITING (Value = 0xE9040006)

- 설명 : 이전 요청 메시지에 대한 응답을 받지 않았습니다.
- 원인 : `_AhnHS_MakeRequest` 함수를 호출하면 클라이언트로부터 Ack 메시지를 받아서 `_AhnHS_VerifyResponse` 함수를 거쳐서 다시 호출 되어야 합니다. 이 동기화 과정이 정상적으로 이루어지지 않으면 발생 합니다.
- 확인 사항:
 - ① 클라이언트 세션 관리상의 동기화 문제가 발생하지 않았는지 확인해 봅니다.
 - ② `_AhnHS_MakeRequest` 함수를 호출하지 않은 상태에서 `_AhnHS_VerifyResponse` 함수가 호출되지 않았는지 확인해 봅니다.

ERROR_ANTICPXSVR_NOT_ENOUGH_MEMORY (Value = 0xE9040007)

- 설명 : 메모리가 부족합니다.
- 원인 : 서버에서 메모리가 부족한 경우에 발생합니다.
- 확인사항 : 서버에 메모리 leak 현상이 발생하지 않은지 확인해 봅니다.

ERROR_ANTICPXSVR_BAD_MESSAGE (Value = 0xE9040008)

- 설명 : 메시지 암복호화에 실패하였습니다.
- 원인 : 검증 받을 버퍼 값이 클라이언트로부터 정확히 받지 않아서 발생한 문제입니다.
- 확인사항 : 클라이언트에서 보낸 버퍼가 정확하게 `pbyResponse` 으로 들어왔는지 확인하시기 바랍니다.

ERROR_ANTICPXSVR_REPLY_ATTACK (Value = 0xE9040009)

- 설명 : 패킷 분석을 위한 재전송 공격이 감지되었습니다.
- 원인 : 위 에러는 핵실드 클라이언트가 동작하지 않은 상태에서 버퍼가 전송되었을 경우 발생합니다.
- 확인사항 : 패킷 분석을 위한 해커의 공격일 수 있습니다.

ERROR_ANTICPXSVR_HSHIELD_FILE_ATTACK (Value = 0xE904000A)

- 설명 : 핵실드 모듈 변조가 감지되었습니다.
- 원인 : 핵실드 파일 `ehsvc.dll` 이 변조된 경우 발생합니다.
- 확인사항 : 클라이언트의 `ehsvc.dll` 파일이 변조되었는지 확인해 봅니다.

ERROR_ANTICPXSVR_CLIENT_FILE_ATTACK (Value = 0xE904000B)

- **설 명** : 클라이언트 파일 변조가 감지되었습니다
- **원 인** : 게임 클라이언트 파일이 변조된 경우 발생합니다.
- **확인사항** : 게임 클라이언트가 정상적으로 업데이트가 되었는지 확인해 봅니다. 바이러스에 의하여 클라이언트가 감염이 되었는지 확인해 봅니다.

ERROR_ANTICPXSVR_MEMORY_ATTACK (Value = 0xE904000C)

- **설 명** : 메모리 변조가 감지되었습니다.
- **원 인** : 클라이언트의 메모리가 변조된 경우 발생합니다.
- **확인사항** : 클라이언트의 메모리를 조작하는 해킹 툴이 있는지 확인해 보아야 합니다.

ERROR_ANTICPXSVR_OLD_VERSION_CLIENT_EXPIRED (Value = 0xE904000D)

- **설 명** : 구 버전의 클라이언트가 접속을 하였습니다.
- **원 인** : 서버를 종료시키지 않고 **hsb** 파일을 업로드 하였을 때 기존에 있던 **hsb** 파일에 대한 클라이언트가 접속을 시도하였을 때 발생합니다.
- **확인사항** : 에러가 발생하는 유저의 클라이언트 버전을 확인해 봅니다. 패치가 정상적으로 이루어지지 않아 게임에서 허용하지 않는 클라이언트 버전일 가능성이 있습니다.

참고

HSB 파일을 생성할 당시 HSBGen.ini 에 있는 GrantOldSession = 0 으로 설정해준 경우 발생할 수 있습니다. 또는 GrantOldSession = 1 이지만 허용할 수 있는 클라이언트 버전 개수가 MaxAllowedNumber 를 초과한 경우 발생합니다.

예)

[VERCT]

GrantOldSession = 1

MaxAllowedNumber = 1

현재 서버에 올라와 있는 HSB 파일에 맞는 클라이언트만 허용하겠다는 의미입니다.

ERROR_ANTICPXSVR_UNKNOWN_CLIENT (Value = 0xE904000E)

- **설 명** : HSB 파일을 생성할 때 지정한 클라이언트와 짝이 맞지 않습니다

- 원인 : HSBGen으로 hsb 를 생성할 때 사용된 클라이언트 버전이 아닌 경우 발생합니다.
- 확인사항 : 개발과정인 경우 hsb 파일을 다시 생성하여 테스트를 진행하시고 서비스 단계에서 특정 유저에게 발생된다면 클라이언트 파일 패치가 정상적으로 이루어지지 않았거나 바이러스에 클라이언트 파일이 감염되었는지 확인해 봐야합니다.

ERROR_ANTICPXSVR_NANOENGINE_FILE_ATTACK
(Value = 0xE9040010)

- 설 명 : 3N.mhe 파일의 변조가 감지되었습니다.
- 원 인 : 3N.mhe 파일이 변조 되었을 경우 발생하는 에러 입니다.
- 확인사항: 3N.mhe 파일이 변조되었는지 확인해 봅니다.

ERROR_ANTICPSVR_INVALID_HACKSHIELD_VERSION
(Value = 0xE9040011)

- 설 명 : 서버에서 지원하지 않는 핵설드 버전입니다..
- 원 인 : 서버에 존재하는 HShield.dat 파일보다 클라이언트에 존재하는 HShield.dat 파일이 더 낮은 버전은 경우 발생합니다.
- 확인사항: 접속하는 클라이언트의 HShield.dat 파일 버전과 서버의 HShield.dat 파일이 동일한지 확인해 보아야 합니다.

ERROR_ANTICPXSVR_INVALID_ENGINE_VERSION
(Value = 0xE9040012)

- 설 명 : 서버에서 지원하지 않는 휴리스틱 엔진 버전입니다..
- 원 인 : 서버에 존재하는 3N.mhe 파일보다 클라이언트에 존재하는 3N.mhe 파일이 더 낮은 버전은 경우 발생합니다
- 확인사항 : 클라이언트의 핵설드 폴더 있는 3N.mhe 파일이 존재하지 않거나 구 버전인지 확인해 보아야 합니다.

ERROR_ANTICPSVR_VERIFY_EXCEPTION
(Value = 0xE9040015)

- 설명 : `_AhnHS_VerifyResponse` 함수 혹은 `_AhnHS_VerifyResponseEx` 함수 수행 중 예외(Exception)가 발생하였습니다.

(본 예러는 Windows용 모듈에서만 반환됩니다)

- 원인 : 여러 가지 원인에 의해서 예외가 발생할 수 있습니다.
- 확인사항 : 예외가 발생한 경우, 내부적으로 **MiniDump** 파일을 남기도록 되어 있습니다.
 ‘**VERIFYRESPONSE** 년 월 일 시-분-초.dmp’ 의 파일명

형식으로 덤프 파일이 생성되므로 해당 파일을
(주)안철수연구소 로 전달해 주시기 바랍니다.

ERROR_ANTICPXSVR_ABNORMAL_HACKSHIELD_STATUS
(Value = 0xE9040018)

- 설 명 : 핵셴드 동작 상태 체크 결과 정상적이지 않습니다.
- 원 인 : 클라이언트에서 동작중인 핵셴드에서 필요한 기능이 정상적으로 동작 중이지 않습니다.
- 확인사항 : 클라이언트에서 동작중인 핵셴드를 공격하는 해킹 툴이 있는지 확인해 봅니다.

ERROR_ANTICPXSVR_DETECT_CALLBACK_IS_NOTIFIED
(Value = 0xE9040019)

- 설 명 : 클라이언트에서 해킹 툴이 감지 되었습니다.
- 원 인 : 클라이언트에서 해킹 툴이 감지 된 경우입니다.
- 확인사항 : 클라이언트에서 동작중인 핵셴드를 공격하는 해킹 툴이 있는지 확인해 봅니다.

ERROR_ANTICPXSVR_UNKNOWN
(Value = 0xE90400FF)

- 설 명 : 정의되지 않은 에러.
- 원 인 : 정상적인 경우에 발생하지 않는 에러로써 위 에러가 발생한다면 추가 확인작업이 필요합니다.
- 확인사항 : (주)안철수연구소 기술지원에 연락하여 주시기 바랍니다.

주의

ERROR_ANTICPXSVR_BAD_MESSAGE,
ERROR_ANTICPXSVR_REPLY_ATTACK,
ERROR_ANTICPXSVR_UNKNOWN_CLIENT,
ERROR_ANTICPXSVR_HSHIELD_FILE_ATTACK,
ERROR_ANTICPXSVR_CLIENT_FILE_ATTACK,
ERROR_ANTICPXSVR_MEMORY_ATTACK,
ERROR_ANTICPXSVR_NANOENGINE_FILE_ATTACK
ERROR_ANTICPXSVR_UNKNOWN_CLIENT
ERROR_ANTICPXSVR_INVALID_HACKSHIELD_VERSION
ERROR_ANTICPXSVR_INVALID_ENGINE_VERSION
ERROR_ANTICPXSVR_ABNORMAL_HACKSHIELD_STATUS
ERROR_ANTICPXSVR_DETECT_CALLBACK_IS_NOTIFIED

위의 에러 코드가 리턴 되었을 경우에는 클라이언트와의 세션을 끊어서 게임을 더 이상 진행할 수 없도록 할 것을 권장합니다. 나머지 메시지의 경우에는 게임사의 정책에 따라 추가할 수 있습니다.

Example

`_AhnHS_VerifyResponse` 함수를 호출한 예제는 다음과 같습니다.

```
예제

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

if ( ulRet == ERROR_ANTICPXSVR_BAD_MESSAGE ||
      ulRet == ERROR_ANTICPXSVR_REPLY_ATTACK ||
      ulRet == ERROR_ANTICPXSVR_HSHIELD_FILE_ATTACK ||
      ulRet == ERROR_ANTICPXSVR_CLIENT_FILE_ATTACK ||
      ulRet == ERROR_ANTICPXSVR_MEMORY_ATTACK ||
      ulRet == ERROR_ANTICPXSVR_NANOENGINE_FILE_ATTACK ||
      ulRet == ERROR_ANTICPXSVR_UNKNOWN_CLIENT ||
      ulRet == ERROR_ANTICPXSVR_INVALID_HACKSHIELD_VERSION ||
      ulRet == ERROR_ANTICPXSVR_INVALID_ENGINE_VERSION ||
      ulRet == ERROR_ANTICPXSVR_ABNORMAL_HACKSHIELD_STATUS ||
      ulRet == ERROR_ANTICPXSVR_DETECT_CALLBACK_IS_NOTIFIED )
{
    // 에러 처리
    // Client Out -> _AhnHS_CloseClientHandle 를
    // 호출하여 hClient 핸들을 닫아줌
}
```

DESCRIPTION

클라이언트에서 사용합니다. 서버에서 전달한 암호화된 버전 요청 메시지를 복호화하고 현재 클라이언트 파일의 버전을 암호화하여 응답 메시지를 만듭니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
pbyRequest	unsigned char *	[IN] 요청 메시지 버퍼
ulRequestLength	unsigned long	[IN] 요청 메시지 길이
pResponseBuffer	PAHNHS_TRANS_BUFFER	[OUT] 응답 메시지 버퍼

RETURN VALUE

ERROR_SUCCESS . (Value = 0x00000000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 경우 입니다.
- 확인사항 :

ERR_ANTICPXCNT_INVALID_PARAMETER . (Value = 0xE4010001)

- 설 명 : 파라미터 값이 올바르지 않습니다.
- 원 인 : pbyRequest, pResponseBuffer 값이 NULL일 경우 발생합니다
- 확인사항: pbyRequest, pResponseBuffer 값이 NULL이 아닌지 확인해봅니다

ERR_ANTICPXCNT_INVALID_ADDRESS (Value = 0xE4010002)

- 설 명 : 잘못된 메모리 주소를 접근했습니다.
- 원 인 : 구조적인 문제일 수 있습니다.
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

ERR_ANTICPXCNT_NOT_ENOUGH_MEMORY (Value = 0xE40100013)

- 설 명 : 메모리가 부족합니다.
- 원 인 : 메모리 할당이 실패를 해서 발생한 것입니다.
- 확인사항 : 서버 메모리에서 leak 현상이 발생하는지 확인해 봅니다.

ERR_ANTICPXCNT_CRC_TABLE_INIT_FAILED (Value = 0xE4010004)

- 설 명 : 초기화하는 데 실패했습니다.
- 원 인 : pbyRequest 버퍼 사이즈가 ANTICPX_TRANS_BUFFER_MAX (400) 을 초과해서 발생한 문제입니다.
- 확인사항 : HShield.h 와 라이브러리를 최신으로 적용해서 빌드하였는지 확인하시기 바랍니다 서버에서 보낸 pbyRequest 버퍼가 정확하게 전달되었는지 확인해 봅니다.

ERR_ANTICPXCNT_BAD_LENGTH (Value = 0xE4010005)

- 설 명 : 메시지 사이즈가 잘못되었습니다.
- 원 인 : 전달될 메시지 버퍼의 사이즈가 정확하지 않아서 발생한 것입니다.
- 확인사항 : HShield.h 와 라이브러리를 최신으로 적용해서 빌드하였는지 확인하시기 바랍니다

ERR_ANTICPXCNT_INSUFFICIENT_BUFFER (Value = 0xE4010006)

- 설 명 : 전달된 버퍼의 사이즈가 잘못되었습니다.
- 원 인 : Ehsvc.dll 과 HShield.lib, Hshield.h 가 버전이 안 맞는 경우 발생할 수 있습니다.
- 확인사항 : HShield.h 와 라이브러리를 최신으로 적용해서 빌드하였는지 확인하시기 바랍니다

ERR_ANTICPXCNT_NOT_SUPPORTED (Value = 0xE4010007)

- 설 명 : 현재 버전에서 지원하지 않는 기능입니다.
- 원 인 : 전달된 메시지 타입이 핵셴드에서 정의되지 않아서 발생한 문제 입니다.
- 확인사항 : 서버에 적용된 핵셴드 버전과 클라이언트에 적용된 핵셴드 버전이 동일한지 확인해 봅니다.

ERR_ANTICPXCNT_FILE_NOT_FOUND (Value = 0xE4010008)

- 설 명 : 클라이언트 파일을 찾을 수 없습니다.
- 원 인 : 클라이언트 파일을 찾을 수 없는 경우 발생합니다.

- 확인사항 : ㈜안철수연구소 에 문의하시기 바랍니다.

ERR_ANTICPXCNT_INVALID_MESSAGE_SIZE (Value = 0xE4010009)

- 설명 : 입력 받은 메시지의 크기가 올바르지 않습니다.
- 원인 : 서버로부터 전달받은 메시지 사이즈가 올바르지 않아서 발생한 문제입니다.
- 확인사항 : 클라이언트 핵셴드 버전과 서버의 핵셴드 버전을 동일하게 맞추어 주었나 확인해 봅니다.

ERR_ANTICPXCNT_BAD_FORMAT (Value = 0xE401000A)

- 설명 : 올바른 포맷이 아닙니다.
- 원인 : 서버로부터 전달받은 메시지 사이즈가 올바르지 않아서 발생한 문제입니다.
- 확인사항 : 클라이언트 핵셴드 버전과 서버의 핵셴드 버전을 동일하게 맞추어 주었나 확인해 봅니다.

ERR_ANTICPXCNT_DEBUGGER_DETECTED (Value = 0xE401000B)

- 설명 : 디버그 상황을 감지했습니다.
- 원인 : 해당 클라이언트가 디버깅 상태인 경우 발생합니다.
- 확인사항 : 디버거 또는 현재 디버깅 중이 아닌지 확인해 봅니다.

ERR_ANTICPXCNT_BAD_HSHIELD_MODULE (Value = 0xE401000C)

- 설명 : 핵셴드 모듈 경로가 잘못되었거나 핵셴드 모듈이 올바르지 않습니다.
- 원인 : 핵셴드 모듈이 존재하지 존재하지 않아서 발생한 문제입니다.
- 확인사항 : 핵셴드 모듈 경로 및 핵셴드 모듈이 올바른지 확인해 봅니다.

ERR_ANTICPXCNT_BAD_CLIENT_FILE (Value = 0xE401000D)

- 설명 : 클라이언트 모듈이 올바르지 않습니다.
- 원인 : 클라이언트 파일에 접근하는데 문제가 발생하였습니다.
- 확인사항 :

ERR_ANTICPXCNT_BAD_REQUEST (Value = 0xE401000E)

- 설명 : 서버로부터 전달 받은 요청 메시지가 올바르지 않습니다
- 원인 : 서버로 받은 버퍼가 정상적으로 받지 못하여 복호화 실패 메시지 발생한 것입니다.
- 확인사항 : 서버로부터 전송된 버퍼의 값과, 길이가 정확하게

pbyRequest 로 들어왔는지 확인해 봐야합니다.

**ERR_ANTICPXCNT_HSHIELD_CORE_ENGINE_NOT_WORKING
(Value = 0xE401000F)**

- 설 명 : 핵싯드 코어 엔진이 정상적으로 동작하지 않습니다.
- 원 인 : 정상적인 경우에 발생하지 않는 에러로써 위 에러가 발생한다면 추가 확인작업이 필요합니다.
- 확인사항 : (주)안철수연구소에 문의해주시기 바랍니다.

ERR_ANTICPXCNT_UNKNOWN . (Value = 0xE40100FF)

- 설 명 : 해킹 시도에 의해 시스템이 오동작 할 경우 발생합니다.
- 원 인 : 정상적인 경우에 발생하지 않는 에러로써 위 에러가 발생한다면 추가 확인작업이 필요합니다.
- 확인사항 : (주)안철수연구소에 문의해주시기 바랍니다.

Example

_AhnHS_MakeResponse를 사용하는 예제는 다음과 같습니다.

```
예제

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

if ( ulRet != ERROR_SUCCESS )
{
    // 에러 처리
}
```

5. 서버 연동 크랙 방지 기능

5.1. 개요

AntiCPSvr은 2005년 (주)안철수연구소에서 개발한 서버 연동을 통한 파일/메모리 조작 감지 SDK(Software Development Kit)입니다. 최근 컴퓨터와 해킹 기술의 발달로 인해 더욱 더 강력한 크랙 방지 기술이 요구되고 있으며, 클라이언트단 뿐만 아니라 서버와 연동한 무결성 검증을 통해 안전하게 프로그램을 실행하는 방법이 대안으로 등장하고 있습니다. AntiCPSvr를 제공함으로써 게임 크랙 및 해설드 정상 동작 유무를 판단하여 사용자들이 공평하게 게임을 즐길 수 있는 환경을 제공합니다.

기능

파일 무결성 검증

서버에서 AntiCpSvr.dll과 AntiCpSvrFunc.h를 적용하고, 클라이언트에서는 해설드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 실시간으로 게임 실행 파일의 조작 여부를 확인할 수 있습니다.

패킷 무결성 기능

패킷을 캡처하고 특정 상황에 맞추어 패킷을 생성해서 정상적으로 동작하게 하는 해킹을 막기 위해 패킷의 무결성을 보장하는 모듈이 내부적으로 동작합니다. 네트워크에서 패킷을 캡처하고 생성하는 등의 방법을 원천적으로 차단할 수 있습니다. 단, 이 기능은 서버 연동 크랙 방지와 관련된 메시지들에 한하여 보호합니다.

해설드 정상동작 확인

서버에서 AntiCPSvr.dll과 AntiCpSvrFunc.h를 적용하고, 클라이언트에서는 해설드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 간단한 방법으로 해설드의 동작 상태를 확인할 수 있습니다. 따라서 서버는 해설드가 동작한 다음에 쿼리 메시지를 전송해야 합니다.

메모리 무결성 기능

서버에서 AntiCpSvr.dll과 AntiCpSvrFunc.h를 적용하고, 클라이언트에서는 해설드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 실시간으로 게임 프로세스의 메모리 조작 여부를 검증할 수 있습니다.

해설드 엔진 무결성 확인

서버에서 AntiCPSvr.dll과 AntiCpSvrFunc.h를 적용하고, 클라이언트에서는 해설드 모듈인 HShield.lib와 HShield.h를 이용하여 서버와 클라이언트간 통신을 통해 엔진파일(v3warpds.v3d, v3warpns.v3d)의 무결성을 검증할 수 있습니다.

특징

인터페이스 함수(API) 제공

AntiCPSvr에서 제공하는 기능을 편리하게 사용하고, 그 실행 결과 값을 받아 볼 수 있도록 인터페이스 DLL과 라이브러리를 제공합니다. 제공되는 인터페이스 DLL과 라이브러리를 사용하여 개발자는 고유 정책에 따라서 게임 파일의 무결성 및 해킹드의 정상 동작 유무를 확인할 수 있습니다.

파일 정보 생성 프로그램 제공

AntiCPSvr에서 제공하는 AntiCPSvrTool.exe는 클라이언트와 서버가 통신하여 클라이언트 프로그램의 파일/메모리 무결성 여부를 판단하는 기준이 되는 각종 파일 정보와 메모리 정보를 생성하고 저장합니다. 게임 개발사 특성에 맞게 서버에서 파일/메모리 정보를 저장할 수 있으며, 이 데이터를 이용하여 서버 연동을 통한 파일/메모리 크랙 방지 기능이 동작하게 됩니다. 자세한 사용 방법은 '9.2. AntiCpSvr 툴 사용 방법'에서 설명합니다.

테스트 프로그램 제공

AntiCPSvr에서 제공하는 API를 사용하여 구현한 테스트용 프로그램인 Amazon.exe를 제공합니다. Amazon.exe는 기존의 해킹드 테스트 기능과 AntiCPSvr의 테스트 기능을 제공합니다.

시스템 구조(System Architecture)

AntiCPSvr는 AntiCpSvr.dll, HShield.lib, EHSvc.dll을 제공하여 서버와 클라이언트에 DLL과 라이브러리 형태로 적용됩니다. AntiCPSvr의 전체 구조 및 동작 원리는 다음과 같습니다.

AntiCPSvr.dll (인터페이스 dll)

서버에서 사용되며 서버에서 요청 메시지를 생성하고 클라이언트에서 받은 메시지를 이용하여 클라이언트 파일 및 메모리가 정상 동작하는지 확인할 수 있게 하는 API를 제공합니다.

HShield.lib (해킹드 라이브러리)

클라이언트에서 사용되며 서버에서 전송한 요청 메시지를 받아 요구한 내용을 수행하고 결과로 얻은 각종 정보를 암호화하여 서버에 전송할 데이터를 만드는 API를 제공합니다.

AntiCpSvrTool.exe (파일 정보 생성 프로그램)

서버에서 사용할 파일 정보 및 메모리 정보 파일을 생성하여 게임 파일의 무결성 여부와 실행되는 게임의 메모리 무결성 여부를 판단할 수 있게 합니다. 파일 정보 데이터는 생성할 때 마다 다르므로 게임 서버가 여러 개이면 서버마다 다르게 적용시켜야 높은 보안을 유지할 수 있습니다. 메모리 정보 데이터는 AntiCpSvrTool.exe를 실행한 상태에서 게임 클라이언트 프로그램을 실행시킬 경우 자동으로 만들어집니다.

HackShield.crc (클라이언트 CRC 파일)

AntiCpSvrTool.exe 툴에서 생성되는 파일로서, 클라이언트 프로그램에 대해 무결성을 보장할 수 있도록 합니다.

HSPub.key (서버 연동 인증 키 파일)

접속되는 클라이언트의 핵셴드를 인증하는 용도로 사용되는 파일로서, HackShield.crc 파일과 동일한 위치에 있어야 합니다.

3N.mhe (휴리스틱 엔진 파일)

클라이언트에서 사용하는 엔진파일로서, 서버연동 사용 시 이전 버전의 3n.mhe 엔진을 사용하는 클라이언트의 접속을 끊을 수 있도록 서버의 HackShield.crc 파일과 동일한 위치에 놓아야 합니다.

HShield.dat (핵셴드 버전 파일)

클라이언트에서 사용하는 데이터파일로서, 서버연동 사용 시 이전 버전의 hshield.dat파일을 사용하는 클라이언트의 접속을 끊을 수 있도록 서버의 HackShield.crc 파일과 동일한 위치에 놓아야 합니다.

참고

서버 HackShield.crc 파일 경로에 3N.mhe 또는 HShield.dat 파일을 놓아두면 이전 버전의 핵셴드로 접속하는 클라이언트에 대해서 접속을 끊을 수 있습니다. (3N.mhe & HShield.dat 파일을 올려 놓을 때는 서버의 재기동이 필요없습니다.)

아래 표와 같이 “접속 끊음”으로 표시되는 경우에 한해서 AntiCpSvr_AnalyzeAckMsg 함수에서 ERROR_ANTICPSVR_ANALGUIDACKMSG_HSHIELDDENIED_NEWSESSION / ERROR_ANTICPSVR_ANALGUIDACKMSG_NANODENIED_NEWSESSION 에러가 리턴됩니다.

해당 에러가 리턴될 경우 클라이언트 접속을 끊으시면 됩니다.

표 5-1 서버연동 버전 관리

파일 명	버전 비교	접속상태
3N.mhe	Client > Server	접속 끊음
	Client ≤ Server	접속 유지
HShield.dat	Client > Server	접속 끊음
	Client ≤ Server	접속 유지

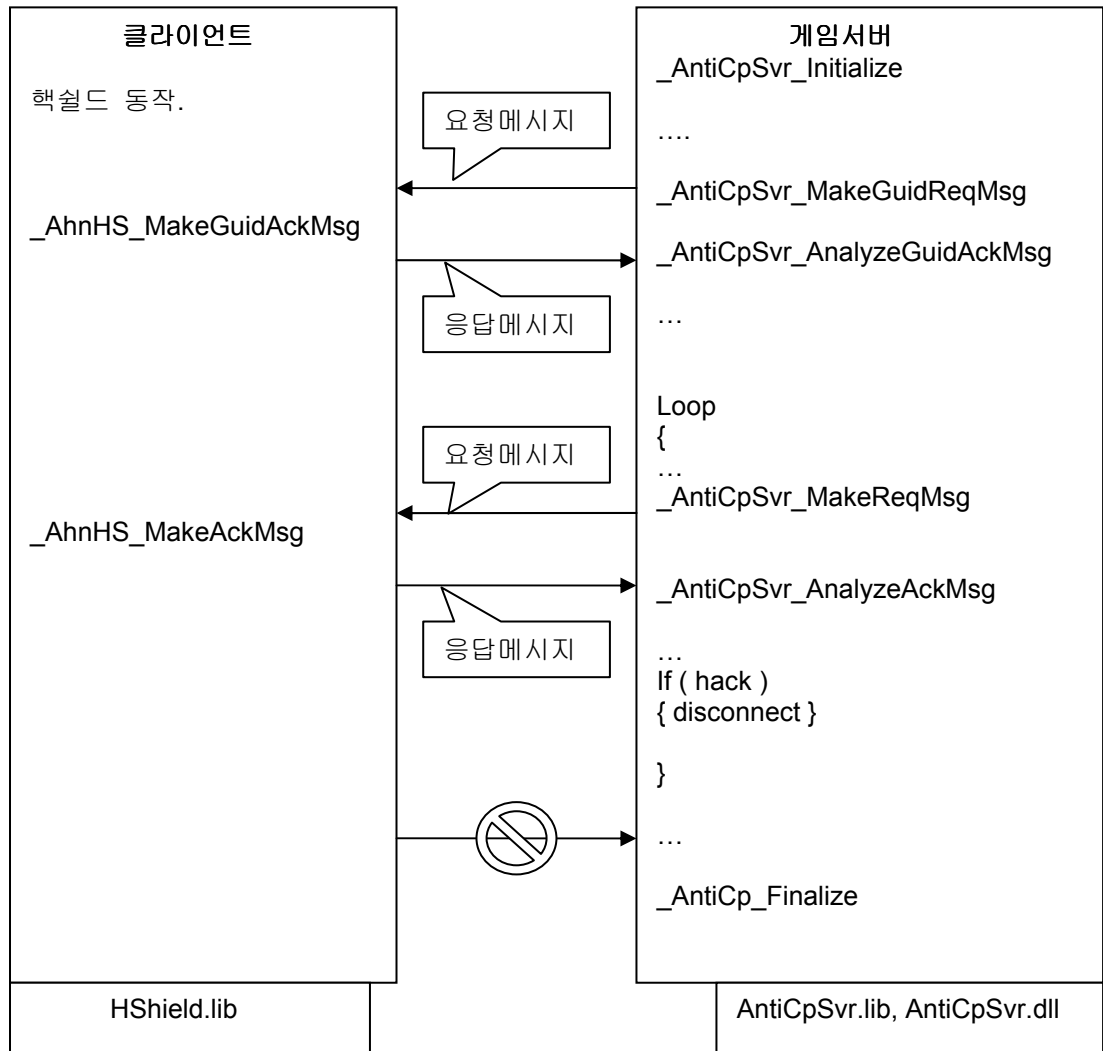


그림 5-1 AntiCpSvr의 동작원리

게임 클라이언트가 서버에 연결되면 해당 클라이언트가 어떤 버전의 실행 파일을 사용하는지 알기 위해 GUID 요청 메시지를 보내고 그에 대한 응답 메시지를 받습니다. 이 때 전달된 버전 정보가 서버에서 허용하는 버전인지 `_AntiCpSvr_AnalyzeGuidAckMsg` API를 통해 알 수 있고 만약 허용하는 버전이라면 연결이 계속 유지되지만 허용하지 않는 버전이라면 연결을 끊음으로써 최신 패치를 받지 않고 게임을 하는 행위를 차단할 수 있습니다.

또한 게임 서버에서는 게임 개발사의 정책에 따라 일정 시간 간격으로 요청 메시지와 응답 메시지를 확인하여 클라이언트의 실행 파일, 메모리, 그리고 해킹드 모듈 및 엔진들에 대한 크랙 유무를 확인할 수 있습니다. 물론 응답 메시지가 게임 서버로 전송이 안되어도 해킹으로 판단합니다.

5.2. Application Programming

AntiCPSvr에서 제공하는 API를 이용해서 파일 및 메모리의 무결성 확인을 구현하는 방법을 설명합니다.

참고

이 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

프로그래밍 적용 방법

AntiCPSvr를 사용하여 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

5.2.1.1.AntiCPSvr 관련 파일

AntiCPSvr 관련 파일

표 5-2 AntiCPSvr 관련 파일

파일 이름	설치 폴더	설명
AntiCPSvrfunc.h	[프로그램 소스 폴더]	서버에서 사용할 헤더 파일
AntiCPSvr.dll	[프로그램 소스 폴더]	서버에서 사용할 DLL 파일
AntiCPSvr.lib	[프로그램 소스 폴더]	AntiCPSvr.dll의 임포트 라이브러리
HackShield.crc	[프로그램 실행 폴더]	클라이언트 무결성 검증 파일
HSPub.key	[프로그램 실행 폴더]	서버 연동 인증 키 파일
3N.mhe	[CRC파일과 동일한 폴더]	핵실드 휴리스틱 엔진 버전 관리 파일
HShield.dat	[CRC파일과 동일한 폴더]	핵실드 버전 관리 파일
HShield.h	[프로그램 소스 폴더]	클라이언트에서 사용할 헤더 파일, 핵실드 기능도 포함되어 있음.
HShield.lib	[프로그램 소스 폴더]	클라이언트에서 사용할 라이브러리 파일, 핵실드 기능도 포함되어 있음.
EHSvc.dll	[프로그램 소스 폴더]	클라이언트에서 사용할 DLL 파일, 핵실드 기능도 포함되어 있음.

5.2.1.2.적용 방법

서버 적용 방법

1. AntiCPSvrTool.exe를 이용하여 HackShield.crc 를 생성합니다.

([10.3.AntiCpSvr](#) 툴 참조)

2. HSPub.key 파일을 HackShield.crc 파일이 존재하는 동일한 폴더에 복사해 둡니다.
3. 3N.mhe 와 HShield.dat 파일을 CRC 파일이 존재하는 폴더에 복사해 둡니다. (이 과정은 필수적인 것은 아니지만 핵셴드 버전관리를 위하여 권장하고 있습니다.)

참고

서버에 crc가 존재하는 위치에 클라이언트에서 사용하는 3N.mhe 와 HShield.dat 파일을 올려둠으로써 3N.mhe 와 HShield.dat 버전이 올바르게 않은 클라이언트를 접속하지 못하게 관리할 수 있습니다.

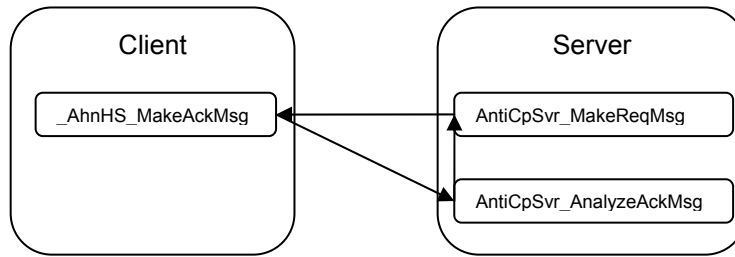
4. AntiCpSvr.dll의 임포트 라이브러리(AntiCpSvr.lib)파일을 프로젝트에 포함시킵니다.
5. 제공되는 AntiCPSvrFunc.h 파일을 작업할 소스 파일에 포함시킵니다.
6. _AntiCpSvr_Initialize를 호출하여 초기화 작업을 수행합니다.
7. 클라이언트 연결 시 _AntiCpSvr_MakeGuidReqMsg를 호출하여 클라이언트로 전달할 암호화된 버전 정보 요청 메시지를 생성합니다.
8. 암호화된 버전 요청 메시지를 클라이언트에게 전송합니다.
9. 클라이언트에서 암호화된 버전 응답 메시지를 받습니다.
10. _AntiCpSvr_AnalyzeGuidAckMsg를 호출하여 클라이언트에서 받은 암호화된 버전 응답 메시지를 분석합니다. ERROR_SUCCESS가 나오지 않으면 접속을 끊습니다.
11. _AntiCpSvr_MakeReqMsg를 호출하여 클라이언트로 전달할 암호화된 요청 메시지를 생성합니다.
12. 암호화된 요청 메시지를 클라이언트에게 전송합니다.
13. 클라이언트에서 암호화된 응답 메시지를 받습니다.
14. _AntiCpSvr_AnalyzeAckMsg를 호출하여 클라이언트에서 받은 암호화된 응답 메시지를 분석합니다. ERROR_SUCCESS가 나오지 않으면 접속을 끊습니다.
15. 서버 프로그램 종료 시 _AntiCpSvr_Finalize 함수를 호출합니다.

클라이언트 적용 방법

1. HShield.lib 파일을 작업할 프로젝트에 포함시킵니다.
2. 제공되는 HShield.h 파일을 작업할 소스 파일에 포함시킵니다.
3. 서버로부터 암호화된 버전 요청 메시지를 받습니다.
4. _AhnHS_MakeGuidAckMsg를 호출하여 서버로 전달한 암호화된 버전 응답 메시지를 생성합니다.
5. 서버로부터 암호화된 요청 메시지를 받습니다.
6. _AhnHS_MakeAckMsg를 호출하여 서버로 전달할 암호화된 응답 메시지를 생성합니다.
7. 암호화된 응답 메시지를 서버에 전송합니다.

서버 연동 주기

주기 설명: 아래 그림과 같이 서버에서 요청 메시지를 보내고 클라이언트에서 응답 메시지를 생성한 후 서버에서 받는 것을 하나의 주기로 설정합니다.



주기 시간: 1주기에 1분이내로 설정하는 것을 권장합니다.
(게임의 특성에 맞추어서 주기 설정이 가능하지만, 서버 연동의 주기가 길어지면 해킹 감지 시간이 길어질 수 있습니다.)

5.3. Application Programming Interface

AntiCpSvr_Initialize

DESCRIPTION

게임 실행 파일, 메모리, 핵셴드 모듈, 엔진 파일에 대한 정보를 가지고 있는 데이터를 로딩하고 기타 초기화 작업을 수행합니다. 파라미터로 전달되는 lpszHashFilePath 경로에 HackShield.crc와 HSPub.key 파일이 반드시 존재해야 합니다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
lpszHashFilePath	const char *	CRC 데이터 파일(HackShield.crc)이 저장된 경로의 전체 경로

RETURN VALUE

ERROR_SUCCESS . (Value = 0x00000000)

- 설 명 : 함수 호출이 성공했을 경우 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPSVR_INIT_INVALIDPARAM (Value = 0x1C001)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : lpszHashFilePath 값이 NULL 일때 발생합니다.
- 확인사항 : lpszHashFilePath 값이 정상적인지 확인합니다.

ERROR_ANTICPSVR_INIT_INSERTCRCDATATOLIST_FAIL (Value = 0x1C002)

- 설 명 : CRC 파일을 추가할 수 없습니다.
- 원 인 : CRC 파일을 읽을 수 없거나 메모리 부족 현상인 경우 발생할 수 있습니다.
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_INIT_READPUBLICKEY_FAIL (Value = 1C004)

- 설 명 : HSPub.key 파일이 없거나 손상이 되었습니다.
- 원 인 : HSPub.key 파일이 해당 경로에 존재하지 않거나 변조되었거나 손상된 경우 발생할 수 있습니다..
- 확인사항 : HSPub.key 파일이 해당 경로에 존재하는지 확인합니다.
(윈도우 서버일 경우에는 폴더 경로 표현을 '/' 아닌 '\'으로 해주어야 합니다.) SDK 에서 HSPub.key 를 새로 복사합니다.

Example

`_AntiCpSvr_Initialize` 함수를 호출한 예제는 다음과 같습니다.

```
예제
dwRet = _AntiCpSvr_Initialize ( m_strFileCrcDataPath // [in]
);
if( dwRet != ERROR_SUCCESS )
{
    // 에러 처리
}
```

AntiCpSvr_Finalize

DESCRIPTION

동적으로 할당된 메모리를 해제하며 내부적으로 사용된 데이터들에 대한 Clean Up 과정을 수행합니다.

SYNTAX

```
void __stdcall  
_AntiCpSvr_Finalize( )
```

PARAMETERS

없음.

RETURN VALUE

없음.

Example

_AntiCpSvr_Finalize 함수를 호출한 예제는 다음과 같습니다.

```
예제  
  
AntiCpSvr_Finalize ();
```


AntiCpSvr_MakeGuidReqMsg

DESCRIPTION

클라이언트로 전달할 암호화된 버전 요청 메시지를 생성합니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
pbyGuidReqMsg	unsigned char *	클라이언트로 전송할 암호화된 버전 요청 메시지로 버퍼의 크기는 AntiCpSvrFunc.h에 정의된 SIZEOF_GUIDREQMSG이어야 합니다.
pbyGuidReqInfo	unsigned char *	_AntiCpSvr_AnalyzeGuidAckMsg 함수에서 분석할 때 필요한 버전 요청 메시지 정보로 버퍼의 크기는 AntiCpSvrFunc.h에 정의된 SIZEOF_GUIDREQINFO이어야 합니다.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPSVR_MAKEGUIDREQMSG_INVALIDPARAM (Value = 0x1C040)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : pbyGuidReqMsg, pbyGuidReqInfo 값이 NULL 일때 발생합니다.
- 확인사항 : pbyGuidReqMsg, pbyGuidReqInfo 값이 NULL 이 아닌지 확인합니다.

ERROR_ANTICPSVR_MAKEGUIDREQMSG_MAKESKEY_FAIL (Value = 0x1C041)

- 설 명 : 메시지 암호화 실패입니다..
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

**ERROR_ANTICPSVR_MAKEGUIDREQMSG_INITCRYPT_FAIL
(Value = 0x1C042)**

- 설 명 : 암복호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

**ERROR_ANTICPSVR_MAKEGUIDREQMSG_ENCRYPT_FAIL
(Value = 0x1C043)**

- 설 명 : 암호화 실패입니다.
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

Example

_AntiCpSvr_MakeGuidReqMsg 함수를 호출한 예제는 다음과 같습니다.

```

예제

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

if( dwRet != ERROR_SUCCESS )
{
    // 에러 처리
}

```

AntiCpSvr_AnalyzeGuidAckMsg

DESCRIPTION

클라이언트에서 전달한 암호화된 버전 응답 메시지를 복호화하여 현재 사용하고 있는 클라이언트 버전이 서버에서 허용하는 버전인지 확인합니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
pbyAckMsg	unsigned char *	클라이언트에서 보낸 암호화된 버전 응답 메시지
pbyGuidReqInfo	unsigned char *	_AntiCpSvr_MakeGuidReqMsg 함수에서 생성한 버전 요청 메시지에 대한 정보
pCrcInfo	PHSHIELD_CLIENT_CONTEXT	해당 클라이언트가 사용할 CRC 정보에 대한 구조체 포인터

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPSVR_ANALGUIDACKMSG_INVALIDPARAM (Value = 0x1c050)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : pbyAckMsg , pbyGuidReqInfo, pCrcInfo 값이 NULL 일때 발생합니다.
- 확인사항 : pbyAckMsg , pbyGuidReqInfo, pCrcInfo 값이 정상인지 확인합니다. (CrcInfo 는 구조체로 선언하여 그 주소 값 (&CrcInfo)을 넘겨야 합니다.)

ERROR_ANTICPSVR_MAKEGUIDREQMSG_MAKESKEY_FAIL
(Value = 0x1C041)

- 설 명 : 메시지 암호화 실패입니다..
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_MAKEGUIDREQMSG_INITCRYPT_FAIL
(Value = 0x1C042)

- 설 명 : 암호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 : 안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_DECRYPT_FAIL
(Value = 0x1c053)

- 설 명 : 메시지 복호화 실패입니다.
- 원 인 : 메시지가 정상적으로 받지 못한 경우 발생할 수 있습니다.
- 확인사항 : 클라이언트에서 보낸 버퍼와 사이즈가 위 함수에 입력 버퍼 (pbyAckMsg) 로 정상적으로 받아졌는지 확인해 봅니다.

클라이언트에서 보낼 때 버퍼를 암호화를 하기 때문에 암호화된 버퍼가 정상적으로 입력버퍼로 들어와야 복호화가 진행됩니다. 버퍼가 비정상적으로 잘려서 들어오면 복호화가 실패되어 해당 에러가 리턴됩니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_PACKET_ERROR
(Value = 0x1c054)

- 설 명 : 패킷 에러가 발생하였습니다.
- 원 인 : 서버에서 MakeGuidReqMsg 호출하고 클라이언트에서 MakeGuidAckMsg 로 응답합니다.
이 동기화가 이루어지지 않는다면 문제가 발생할 수 있습니다.
- 확인사항 : 클라이언트와 서버와 통신은 항상 동기화 되어야 합니다.
동기화 또는 네트워크 문제로 서버에서 요청한 패킷이 손상되었지 확인해 보아야 합니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_DENIED_NEWSESSION
(Value = 0x1c055)

- 설 명 : 이미 연결된 세션에 대해서는 허용을 하지만 해당 버전을 가진 새로운 연결은 허용하지 않습니다.

- 원인 : 서버에서 사용되는 **crc**를 생성한 클라이언트 버전이 아닌 이전 버전의 클라이언트가 접속을 할 경우 리턴되는 에러입니다.
- 확인사항 : 개발 과정이라면 **CRC**를 다시 생성해서 서버를 시작하시기 바랍니다. 서비스 중에 발생한 것이라면 현재 클라이언트가 정상적으로 패치되지 않았거나 바이러스에 의한 감염 문제일 수 있습니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_GETGUIDFROMCRCFILE_ERROR (Value = 0x1c056)

- 설명 : **CRC** 파일을 읽는데 실패하였습니다.
- 원인 : 서버에서 **HackShield.crc** 파일이 없거나 손상된다면 문제가 발생할 수 있습니다.
- 확인사항 : **HackShield.crc** 파일이 존재하는지 확인해 봅니다. 새로운 **crc** 파일을 생성해 보고 위 같은 문제가 발생할 경우 (주)안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_INSERTCRCDATATOLIST_FAIL (Value = 0x1c057)

- 설명 : **CRC** 데이터를 갱신하는데 실패하였습니다.
- 원인 : 서버에 메모리가 부족하면 발생할 수 있습니다.
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_INVALIDGUID (Value = 0x1c058)

- 설명 : 지원하지 않는 버전입니다.
- 원인 : 서버에서 해당 클라이언트 버전을 허용하지 않아서 발생한 문제입니다.
- 확인사항 : 개발 과정이라면 **CRC**를 다시 생성해서 서버를 시작하시기 바랍니다. 서비스 중에 발생한 것이라면 현재 클라이언트가 정상적으로 패치되지 않았거나 바이러스에 의한 감염 문제일 수 있습니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_HSHIELDDENIED_NEWSESSION (Value = 0x1c059)

- 설명 : 서버에서 지원하지 않은 핵셴드 버전입니다.
- 원인 : 접속한 클라이언트 **HShield.dat** 파일 버전이 서버에서 지원하지 않는 **HShield.dat** 파일이 아니어서 발생한 것입니다. 보통 서버의 **HShield.dat** 파일보다 클라이언트 **HShield.dat** 버전이 하위버전인 경우 발생합니다.

- 확인사항 : 클라이언트의 해설드 업데이트가 정상적으로 이루어지지 않았거나 또는 해킹 시도 일 수 있습니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_INSERTHSHIELDRCRDATATOLIST_FAIL (Value = 0x1c05B)

- 설 명 : 해설드 버전을 갱신하는데 실패하였습니다..
- 원 인 : 서버에서 해설드 버전 정보를 갱신하면서 실패한 것입니다.
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_NANODENIED_NEWSESSION (Value = 0x1c05D)

- 설 명 : 서버에서 지원하지않은 휴리스틱 엔진 (3N.mhe) 버전입니다.
- 원 인 : 접속한 클라이언트 3N.mhe 파일 버전이 서버에서 지원하지 않는 3N.mhe 파일 버전인 경우 발생합니다. 보통 서버의 3N.mhe 파일보다 클라이언트 3N.mhe버전이 하위버전인 경우 발생합니다.
- 확인사항 : 클라이언트의 해설드 업데이트가 정상적으로 이루어지지 않았거나 또는 해킹 시도 일 수 있습니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_INSERTNANOCRCRDATATOLIST_FAIL (Value = 0x1c05E)

- 설 명 : 3N.mhe 파일 버전을 갱신하는데 실패하였습니다.
- 원 인 : 서버에서 3N.mhe파일 버전 정보를 갱신하면서 실패한 것입니다.
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_ANALGUIDACKMSG_CLIENTNANOENGINE_OLDVERSION (Value = 0x1c05F)

- 설 명 : 서버에서 지원하지 않는 해설드 버전입니다.
- 원 인 : HShield.dat 버전이 서버에서 지원하지 않는 버전인 경우 발생합니다.
- 확인사항 : 배포된 클라이언트의 HShield.dat 파일이 위변조 되지 않았는지 확인해 봅니다.

주의

위의 에러 코드가 리턴되었을 경우에는 클라이언트와의 세션을 끊어서 게임을 더 이상 진행할 수 없도록 할 것을 권장합니다.

Example

`_AntiCpSvr_AnalyzeGuidAckMsg` 함수를 호출한 예제는 다음과 같습니다.

```
예제

HSHIELD_CLIENT_CONTEXT CrcInfo = {0, };

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

if( dwRet != ERROR_SUCCESS )
{
    // 에러 처리
}
```

OUT 파라미터로 리턴되는 CRC 정보에 대한 구조체(CrcInfo)는 해당 클라이언트가 세션을 끝을 때까지 사용되는 정보이므로 클라이언트별로 보관되어야 하며 `_AntiCpSvr_MakeReqMsg` 함수와 `_AntiCpSvr_AnalyzeAckMsg` 함수에 입력 파라미터로 사용됩니다.

AntiCpSvr_MakeReqMsg

DESCRIPTION

클라이언트로 전달할 암호화된 CRC 요청 메시지를 생성합니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
pCrcInfo	PSHIELD_CLIENT_CONTEXT	_AntiCpSvr_AnalyzeGuidAckMsg 함수에서 리턴된 해당 클라이언트가 사용할 CRC 정보에 대한 구조체 포인터
pbyReqMsg	unsigned char *	클라이언트로 전송할 암호화된 CRC 요청 메시지, 버퍼의 크기는 AntiCpSvrFunc.h에 정의된 SIZEOF_REQMSG이어야 합니다.
pbyReqInfo	unsigned char *	_AntiCpSvr_AnalyzeAckMsg 함수에서 분석할 때 필요한 원본 CRC 요청 메시지 정보로 버퍼의 크기는 AntiCpSvrFunc.h에 정의된 SIZEOF_REQINFO이어야 합니다.
ulOption	unsigned long	_AntiCpSvr_AnalyzeAckMsg 함수에서 분석할 때 어떤 정보들에 대한 Request Message를 만들지에 대한 Flag로 AntiCpSvrFunc.h에 ANTICPSVR_CHECK_GAME_MEMORY, . ANTICPSVR_CHECK_HACKSHIELD_FILE, ANTICPSVR_CHECK_GAME_FILE, ANTICPSVR_CHECK_NANOENGINE_FILE, ANTICPSVR_CHECK_ALL로 정의되어 있습니다. 단, 안전을 위해 최초 호출 시는 ANTICPSVR_CHECK_ALL Option을 이용하여 전체에 대한 안전 유무를 검사하고 그 다음부터는 Performance를 위해 ANTICPSVR_CHECK_GAME_MEMORY Option만 사용하길 권장합니다.

RETURN VALUE

ERROR_SUCCESS

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPSVR_MAKEREQMSG_INVALIDPARAM (Value = 0x1C010)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : pbyAckMsg , pbyGuidReqInfo, pCrcInfo 값이 NULL 일때 발생합니다.
- 확인사항 : 세번째 파라미터, uloption 값이 '0' 이거나, 입력 파라미터 값이 'NULL' 이면 발생할 수 있습니다

ERROR_ANTICPSVR_MAKEREQMSG_MAKESKEY_FAIL (Value = 0x1C011)

- 설 명 : 메시지 암호화 실패입니다..
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_MAKEREQMSG_INITCRYPT_FAIL (Value = 0x1C012)

- 설 명 : 암복호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_MAKEREQMSG_ENCRYPT_FAIL (Value = 0x1C013)

- 설 명 : 메시지 복호화 에 실패했습니다.
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPSVR_MAKEREQMSG_GETRNDHASHINFO_FAIL (Value = 0x1C014)

- 설 명 : CRC 정보를 가져오는데 실패하였습니다.
- 원 인 : pCrcInfo 이 'NULL' 또는 잘못된 값을 가지고 있다면 발생할 수 있습니다
- 확인사항 :

Example

`_AntiCpSvr_MakeReqMsg` 함수를 호출한 예제는 다음과 같습니다.

예제

```
// byClientInfo, CrcInfo 는 GUID 요청 과정에서 구해진 값으로
// 클라이언트가 접속되는 동안 계속 유지되고 있는 값입니다.

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

if( dwRet != ERROR_SUCCESS )
{
    // 에러 처리
}
```

OUT으로 리턴되는 `byReqInfo`는 해당 요청 메시지에 대한 응답 메시지를 분석할 때 사용되므로 해당 요청 메시지에 대한 응답 메시지를 분석하기 위해 `_AntiCpSvr_AnalyzeAckMsg` 함수를 호출할 때까지 정보를 저장해야 합니다. 즉, Multi-Thread Safe하게 보존되어야 합니다.

AntiCpSvr_AnalyzeAckMsg

DESCRIPTION

클라이언트에서 전달한 암호화된 CRC 응답 메시지를 복호화하여 게임 파일 및 패킷의 무결성, 핵섀드 정상 동작 유무 및 핵섀드 모듈의 무결성, 그리고 메모리 무결성을 확인합니다.

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	_AntiCpSvr_AnalyzeGuidAckMsg 함수에서 리턴된 해당 클라이언트가 사용할 CRC 정보에 대한 구조체 포인터
pbyAckMsg	unsigned char *	클라이언트에서 보낸 암호화된 CRC 응답 메시지
pbyReqInfo	unsigned char *	_AntiCpSvr_MakeReqMsg 함수에서 리턴된 CRC 요청 메시지 정보

RETURN VALUE

ERROR_SUCCESS (Value = 0x000000)

- 설 명 : 성공적으로 리턴하였습니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPSVR_ANALACKMSG_INVALIDPARAM (Value = 0x1C020)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : pbyAckMsg , pbyReqInfo, pCrcInfo 값이 NULL 일때 발생합니다.
- 확인사항 : pbyAckMsg , pbyReqInfo, pCrcInfo 값이 NULL 인지 확인합니다.

ERROR_ANTICPSVR_ANALACKMSG_MAKESKEY_FAIL

(Value = 0x1C021)

- 설 명 : 메시지 암호화 실패입니다..
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

**ERROR_ANTICPSVR_ANALACKMSG_INITCRYPT_FAIL
(Value = 0x1C022)**

- 설 명 : 암호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

**ERROR_ANTICPSVR_ANALACKMSG_DECRYPT_FAIL
(Value = 0x1c023)**

- 설 명 : 메시지 복호화 실패입니다.
- 원 인 : 메시지가 정상적으로 받아지지 않아서 발생할 수 있습니다.
- 확인사항 : 클라이언트에서 보낸 버퍼와 사이즈가 위 함수에 입력 버퍼 (pbyAckMsg) 로 정상적으로 받아졌는지 확인해 봅니다.

클라이언트에서 메시지를 보낼 때 암호화를 하기 때문에 암호화된 버퍼가 정상적으로 입력버퍼로 들어와야 복호화가 진행됩니다. 버퍼가 비정상적으로 잘려서 들어오면 복호화가 실패되어 해당 에러가 리턴됩니다.

**ERROR_ANTICPSVR_ANALACKMSG_HSHIELD_ERROR
(Value = 0x1c024)**

- 설 명 : 핵셴드 모듈 변조가 감지되었습니다
- 원 인 : Ehsvc.dll 과 HShield.dat 파일은 항상 동일한 버전이 배포됩니다. 두 파일중 모듈이 변조 되었다면 위 에러가 리턴됩니다.
- 확인사항 : 해당 클라이언트의 핵셴드 파일을 확인해 주시고 업데이트가 정상적으로 이루어졌는지 확인해 봅니다. 해킹으로 인하여 핵셴드 모듈이 변조되었을 가능성도 있습니다.

**ERROR_ANTICPSVR_ANALACKMSG_PACKET_ERROR
(Value = 0x1c025)**

- 설 명 : 패킷 에러가 발생하였습니다.
- 원 인 : 서버에서 MakeReqMsg 호출하고 클라이언트에서 MakeAckMsg 로 응답합니다.
이 동기화가 이루어지지 않는다면 문제가 발생할 수 있습니다.

- 확인 사항 : 클라이언트와 서버와 통신은 항상 동기화 되어야 합니다. 동기화 또는 네트워크 문제로 서버에서 요청한 패킷이 손상되었지 확인해 보아야 합니다.

ERROR_ANTICPSVR_ANALACKMSG_FILECRC_ERROR ERROR (Value = 0x1c026)

- 설 명 : 클라이언트 모듈 변조가 감지되었습니다
- 원 인 : 클라이언트 파일이 변조되면 해당 에러를 리턴하게 됩니다.
- 확인사항 : 해킹툴에 의해 변조 되었거나 클라이언트 패치가 정상적으로 이루어졌는지 확인해 보아야 합니다.

ERROR_ANTICPSVR_ANALACKMSG_MEMORYCRC_ERROR (Value = 0x1c027)

- 설 명 : 메모리 변조가 감지되었습니다.
- 원 인 : 메모리가 변조되면 해당 에러를 리턴하게 됩니다.
- 확인사항 :

ERROR_ANTICPSVR_ANALACKMSG_INVALIDSESSION_ERROR (Value = 0x1c028)

- 설 명 : 서버에서 해당 클라이언트의 버전을 허용하지 않습니다.
- 원 인 : 파일이 패치되면서 이미 연결된 세션을 끊도록 옵션을 주었을 때 발생하는 에러입니다.
- 확인사항 : 서버를 재시작하지 않고 crc 파일을 업로드 하였을 때 발생할 수 있습니다. (crc 파일을 생성할 때 구 버전의 클라이언트에 대하여 허용을 하지 않았는지 확인합니다.)

ERROR_ANTICPSVR_ANALACKMSG_NANOENGINECRC_ERROR (Value = 0x1c02B)

- 설 명 : 3N.mhe 파일 변조가 감지되었습니다
- 원 인 : 3N.mhe 파일 변조되면 해당 에러를 리턴하게 됩니다.
- 확인사항 : 해킹툴에 의해 변조 되었거나 클라이언트 패치가 정상적으로 이루어졌는지 확인해 보아야 합니다.

주의

위의 에러 코드가 리턴되었을 경우에는 클라이언트와의 세션을 끊어서 게임을 더 이상 진행할 수 없도록 할 것을 권장합니다.

Example

_AntiCpSvr_AnalyzeAckMsg 함수를 호출한 예제는 다음과 같습니다.

```
예제

// byClientInfo, CrcInfo 는 MakeReqMsg 요청 과정에서 구해진 값으로
// 클라이언트가 접속되는 동안 계속 유지되고 있는 값입니다.

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

if( dwRet != ERROR_SUCCESS )
{
    // 에러 처리
}
```

AhnHS_MakeGuidAckMsg

DESCRIPTION

클라이언트에서 사용하며 서버에서 전달한 암호화된 버전 요청 메시지를 복호화하고 현재 클라이언트 파일의 버전을 암호화하여 응답 메시지를 만듭니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
pbyGuidReqMsg	unsigned char *	서버에서 보낸 암호화된 버전 요청 메시지로 버퍼의 크기는 HShield.h에 정의된 SIZEOF_GUIDREQMSG이어야 합니다.
pbyGuidAckMsg	unsigned char *	서버로 보낼 암호화된 버전 응답 메시지로 버퍼의 크기는 HShield.h에 정의된 SIZEOF_GUIDACKMSG이어야 합니다.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 성공적으로 리턴하였습니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPNT_MAKEGUIDACKMSG_INVALIDPARAM (Value = 0x10010)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : pbyGuidReqMsg , pbyGuidAckMsg 값이 NULL 일때 발생합니다.
- 확인사항 :

ERROR_ANTICPNT_MAKEGUIDACKMSG_INITCRYPT_FAIL (Value = 0x1C012)

- 설 명 : 암호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 :

ERROR_ANTICPCNT_MAKEGUIDACKMSG_DECRYPTMESSAGE_FAIL (Value = 0x10013)

- 설 명 : 메시지 복호화 실패입니다.
- 원 인 : 메시지가 정상적으로 받아지지 않아서 발생할 수 있습니다.
- 확인사항 : 서버에서 보낸 버퍼와 사이즈가 위 함수에 입력 버퍼 (`pbyGuidReqMsg`) 로 정상적으로 받아졌는지 확인해 봅니다.

서버에서 버퍼를 보낼 때 암호화를 하기 때문에 암호화된 버퍼가 정상적으로 입력버퍼로 들어와야 복호화가 진행됩니다. 버퍼가 비정상적으로 잘려서 들어오면 복호화가 실패되어 해당 에러가 리턴됩니다.

ERROR_ANTICPCNT_MAKEGUIDACKMSG_GETIDENTIFIER_FAIL (Value = 0x10014)

- 설 명 : GUID 값을 찾을 수 없습니다
- 원 인 : GUID 를 생성하는 도중에 핵셴드 모듈 또는 게임 클라이언트 파일에 문제가 생긴 것 입니다.
- 확인사항 : 핵셴드 모듈 또는 게임 클라이언트 파일이 정상적인지 확인하시기 바랍니다.

ERROR_ANTICPCNT_MAKEGUIDACKMSG_ENCRYPTMESSAGE_FAIL (Value = 0x10016)

- 설 명 : 메시지 암호화에 실패했습니다
- 원 인 :
- 확인사항 : (주)안철수연구소에 문의하시기 바랍니다.

Example

`_AhnHS_MakeGuidAckMsg`를 사용하는 예제는 다음과 같습니다.

예제

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



```
if( dwRet != ERROR_SUCCESS )  
{  
    // 에러 처리  
}
```

DESCRIPTION

클라이언트에서 사용하며 서버에서 전달한 암호화된 CRC 요청 메시지를 복호화하고 게임 파일의 각종 정보를 구해 암호화된 CRC 응답 메시지를 생성합니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
pbyReqMsg	unsigned char *	서버에서 보낸 암호화된 CRC 요청 메시지로 버퍼의 크기는 HShield.h에 정의된 SIZEOF_REQMSG이어야 합니다.
pbyAckMsg	unsigned char *	서버로 보낼 암호화된 CRC 응답 메시지로 버퍼의 크기는 HShield.h에 정의된 SIZEOF_ACKMSG이어야 합니다.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 성공적으로 리턴하였습니다.
- 원 인 : 정상적인 경우 발생합니다.
- 확인사항 :

ERROR_ANTICPNT_MAKEACKMSG_INVALIDPARAM (Value = 0x10000)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : pbyReqMsg , pbyAckMsg 값이 NULL 일때 발생합니다.
- 확인사항 :

ERROR_ANTICPNT_MAKEACKMSG_INITCRYPT_ (Value = 0x10012)

- 설 명 : 암호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

Example

_AhnHS_MakeAckMsg를 사용하는 예제는 다음과 같습니다.

```
예제

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

if( dwRet != ERROR_SUCCESS )
{
    // 에러 처리
}
```

DESCRIPTION

보호하고자 하는 게임 클라이언트내의 함수 포인터를 인자로 받아 해당 함수들의 CRC를 HackShield.crc 형태의 파일로 저장합니다. 메모리 무결성을 보장하기 위해 서버에서는 이 함수를 통해 생성된 HackShield.crc 파일을 관리해야 합니다. 함수의 인자로 함수 포인터를 넘길 경우 최대 32개까지의 함수를 보호할 수 있으며 첫 번째 인자로 넘어가는 함수의 개수와 뒤에 오는 함수 포인터의 개수가 정확하게 일치해야 합니다.

주의

이 함수는 EXE에 존재하는 함수여야 하며 동일한 이름을 가진 함수들(오버로딩)에 대해서는 지원하지 않습니다.

SYNTAX

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

PARAMETERS

Parameter	Type	Description
unNumberOfFunc	unsigned int	보호할 함수의 개수
...	...	가변 인자이며 함수의 포인터를 입력합니다.

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 성공적으로 리턴하였습니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

ERROR_ANTICPNT_SAVEFUNCADDRESS_INVALIDPARAM (Value = 0x10020)

- 설 명 : 입력 값이 올바르지 않습니다.
- 원 인 : 입력 포맷이 올바르지 않아서 발생한 문제입니다.
- 확인사항 :

ERROR_ANTICPCNT_SAVEFUNCADDRESS_INITCRYPT_FAIL
(Value = 0x10023)

- 설 명 : 암호화 초기화에 실패했습니다.
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPCNT_SAVEFUNCADDRESS_DECRYPTMESSAGE_FAIL
(Value = 0x10024)

- 설 명 : 메시지 복호화에 실패했습니다.
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

ERROR_ANTICPCNT_SAVEFUNCADDRESS_ENCRYPTMESSAGE_FAIL FAIL
(Value = 0x10029)

- 설 명 : 메시지 암호화에 실패했습니다
- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

6. 모니터링 서비스 기능

6.1. 개요

AhnLab HackShield Hacking Monitoring System은 핵쉴드가 적용된 프로그램의 해킹 상황과 에러 상황을 중앙에서 모니터링 하는 프로그램입니다. 해킹의 확산과 에러로 인한 장애는 전체 서비스에 큰 피해를 입힐 수 있기 때문에, 해킹 상황과 에러 상황을 실시간으로 모니터링하여 초기에 대응하는 것이 매우 중요합니다. 또한, AhnLab HackShield Hacking Monitoring System은 보고서 기능을 제공함으로써, 서비스 상황의 통계 자료로서 매우 유용한 기능을 제공합니다.

기능

실시간 해킹/에러 모니터링

핵쉴드가 적용된 프로그램의 해킹 상황과 에러 상황에 대해 핵쉴드가 전달한 정보를 실시간으로 모니터링 할 수 있습니다

보고서

수집된 로그를 다양하게 통계를 내어 필요한 정보만 보고서로 출력하여 볼 수 있습니다.

정책 관리

필요한 상황에 맞추어 다양한 정책을 설정하고 DB 정보를 백업할 수 있습니다.

특징

인터페이스 함수(API) 제공

Ehsvc 에서 제공하는 기능을 편리하게 사용하고, 그 실행 결과 값을 받아 볼 수 있도록 인터페이스 DLL과 라이브러리를 제공합니다. 제공되는 인터페이스 DLL과 라이브러리를 사용하여 개발자는 고유 정책에 따라서 게임 파일의 무결성 및 핵쉴드의 정상 동작 유무를 확인할 수 있습니다.

모니터링 서버 프로그램 제공

클라이언트에서 보내온 해킹 및 에러 정보를 실시간으로 확인 할 수 있도록 오라클을 사용하여 DB 를 관리하고 실시간으로 확인할 수 있도록 모니터링 프로그램을 제공합니다.

테스트 프로그램 제공

Ehsvc에서 제공하는 API를 사용하여 구현한 테스트용 프로그램인 Amazon.exe를 제공합니다. Amazon.exe는 기존의 핵쉴드 테스트 기능과

Ehsvc의 테스트 기능을 제공합니다.

시스템 구조(System Architecture)

HShield.lib, EHsvc.dll을 제공하여 서버와 클라이언트에 DLL과 라이브러리 형태로 적용됩니다. 모니터링 서비스의 전체 구조 및 동작 원리는 다음과 같습니다.

Ehsvc.dll (인터페이스 dll)

해킹 및 에러가 발생했을 경우 모니터링 서버에 알릴수 있도록 기본적인 정보를 세팅하는 API 를 제공합니다.

HShield.lib (핵싧드 라이브러리)

해킹 및 에러가 발생했을 경우 모니터링 서버에 알릴수 있도록 기본적인 정보를 세팅하는 API 를 제공합니다.

HSMS_Setup.exe(모니터링 서버 사이드 설치 파일)

클라이언트에서 보내온 정보에 대해 DB로 관리하고 웹상에서 쉽게 관리할 수 있도록 제공해주는 서버 사이드 프로그램입니다.

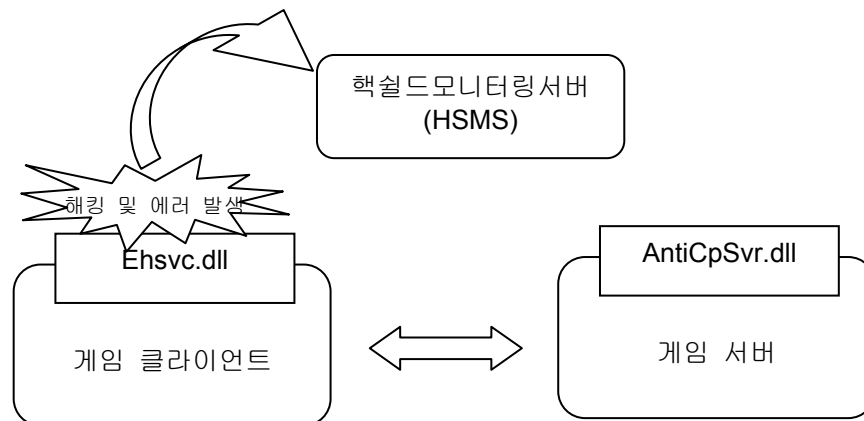


그림 6-1 모니터링 서비스의 동작원리

게임 클라이언트에서 `_AhnHS_StartMonitor` 함수를 호출하게 되면 해당 클라이언트는 클라이언트에 대한 해킹 및 에러가 발생했을 경우 그 정보를 모니터링 서버에 전송하게 됩니다.

모니터링 서버에서는 클라이언트 프로그램을 특정 게임코드로 관리하게 되며 해당 게임코드에 관련된 클라이언트가 전송한 정보를 DB로 관리하게 되고 웹 화면으로 쉽게 액세스 할 수 있습니다.

6.2. Application Programming

Ehsvc에서 제공하는 API를 이용해서 모니터링 서버에 관한 정보 및 유저 정보를 세팅합니다.

참고

이 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

프로그래밍 적용 방법

모니터링을 사용하여 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

6.2.1.1.모니터링 관련 파일

모니터링 관련 파일

표 6-1 모니터링 관련 파일

파일 이름	설치 폴더	설명
HShield.h	[프로그램 소스 폴더]	클라이언트에서 사용할 헤더 파일, 핵셴드 기능도 포함되어 있음.
HShield.lib	[프로그램 소스 폴더]	클라이언트에서 사용할 라이브러리 파일, 핵셴드 기능도 포함되어 있음.
EHSvc.dll	[프로그램 소스 폴더]	클라이언트에서 사용할 DLL 파일, 핵셴드 기능도 포함되어 있음.

6.2.1.2.적용 방법

서버 적용 방법

1. 핵셴드 모니터링 서버 설치 가이드를 참고하여 해당 서버에 핵셴드 모니터링 서버 프로그램을 설치합니다.
2. 핵셴드 모니터링 서버 운용 가이드를 참고하여 운용하도록 합니다.

클라이언트 적용 방법

1. HShield.lib 파일을 작업할 프로젝트에 포함시킵니다.
2. 제공되는 HShield.h 파일을 작업할 소스 파일에 포함시킵니다.
3. HShield.h 에 정의된 AHNHS_EXT_ERRORINFO 구조체에 대한 변수를 생성하고 초기화 합니다.

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

4. HsExtError 구조체 내 szServer(모니터링 서버주소), szUserId(유저 계정), szGameVersion(게임 버전) 에 정보를 대입합니다.

```
HsExtError.szServer = "127.0.0.1"    //모니터링 주소  
HsExtError.szUserId = "Test"        //유저 ID  
HsExtError.szGameVersion = "3.0.0.1" //Game 버전
```

참고

모니터링 IP/Port 정보는 HSUpdate.env 파일을 통해 추후 수정이 가능합니다.

5. 위 스텝에서 생성한 구조체 변수 값과 Ehsvc.dll의 전체 경로를 인자값으로 전달하여 _AhnHS_StartMonitor함수를 호출합니다.

주의

반드시 _AhnHS_Initialize 함수 전에 호출 되어야합니다.

그렇지 않으면 핵싧드가 구동되기 전의 에러는 전송되지 않습니다.

```
lstrcat (szFullFileName, _T("\\HShield\\Ehsvc.dll" ));  
  
dwRet = _AhnHS_StartMonitor ( HsExtError    // [in]  
                             szFullFileName // [in]  
                             );  
  
if( dwRet != ERROR_SUCCESS )  
{  
    // 실패가 발생한다면 모니터링 기능만 동작이 안되는 것이므로  
    // 에러에 관한 로그만 기록되도록 한다.  
}  
  
dwRet = _AhnHS_Initialize ( ...
```

6. 4번 스텝에서 유저 ID 정보를 얻어올 수 없는 시점이라면 유저 ID를 얻어올 수 있는 시점에 _AhnHS_SetUserId 함수를 호출하면 유저 ID 를 세팅할 수 있습니다.

6.3. Application Programming Interface

AhnHS_StartMonitor

DESCRIPTION

해킹 및 에러발생시 데이터를 전송할 서버 정보를 세팅한다.
_AhnHS_Initialize 함수 이전에 호출합니다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
HsExtErrorInfo	AHNHS_EXT_ERRORINFO	서버 URL 주소, User ID, Game Version 정보를 가지고 있는 구조체
szFileName	LPCSTR	Ehsvc.dll의 전체 경로

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 함수 호출을 성공했을 때 리턴하는 값입니다.
- 원 인 : 정상적인 상황입니다.
- 확인사항 :

HS_ERR_INVALID_FILES (Value = 0x1C001)

- 설 명 : 입력 값이 올바르지 않습니다
- 원 인 : 파라미터가 NULL인 경우 발생합니다.
- 확인사항 : HsExtErrorInfo, szFileName 값이 정상적인지 확인해 봅니다.

HS_ERR_UNKNOWN (Value = 0x1C002)

- 설 명 : 알 수 없는 에러입니다.

- 원 인 :
- 확인사항 : ㈜안철수연구소에 문의하시기 바랍니다.

Example

`_AhnHS_StartMonitor` 함수를 호출한 예제는 다음과 같습니다.

```

예제

AHNHS_EXT_ERRORINFO HsExtError;    //HShield.h 에 정의된 구조체

HsExtError.szServer = "127.0.0.1"    //모니터링 주소
HsExtError.szUserId = "Test"         //유저 ID
HsExtError.szGameVersion = "3.0.0.1" //Game 버전

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

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

if( dwRet != ERROR_SUCCESS )
{
    // 실패가 발생한다면 모니터링 기능만 동작이 안되는 것이므로
    // 에러에 관한 로그만 기록되도록 한다.
}

dwRet = _AhnHS_Initialize ( ...

```

AhnHS_SetUserId

DESCRIPTION

모니터링 서버에 전송될 메시지의 유저 ID를 저장한다.
_AhnHS_StartMonitor에서도 유저 ID정보를 받으나 해설드 초기화 시점에 유저 정보를 알지 못하는 경우가 있다. 따라서 유저 ID를 아는 시점에 이 함수를 호출하여 유저정보를 얻는다. 아이디를 얻기 전까지는 아이디 정보가 없이 에러값이 전송된다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
szUserUD	LPCSTR	게임 클라이언트의 유저 정보

RETURN VALUE

없음.

Example

_AhnHS_SetUserId 함수를 호출한 예제는 다음과 같습니다.

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

7. LMP 기능

7.1. 개요

Local Memory Protection 은 일부 패커²들을 사용했을 때, 기존의 서버연동 기능의 메모리 보호 방식으로 보호 할 수 없는 경우를 대비하여 클라이언트 단에서 메모리 위/변조를 감지할 수 있는 기능입니다. 이는 기존의 서버연동에서의 메모리 변조 감지 기능과 유사한 기능이지만 서버를 거치지 않고 클라이언트 자체적으로 자신의 메모리 영역을 보호할 수 있게 합니다

LMP 1.0

CSInspector 툴을 이용하여 EXE 또는 DLL 에 보호 파일을 적용할 수 있으며, THEMIDA 의 API Wrapping 옵션을 지원하지 않습니다.

- 보호 파일: EXE, DLL 파일
- 사용 툴: CSInspector.exe

LMP 2.0

기존 LMP 1.0 기능을 그대로 가지고 있고, THEMIDA 의 API Wrapping을 지원하기 위하여 개발된 로컬 메모리 보호 기능으로써, 서버연동의 hsb 파일 생성기를 통하여 게임 클라이언트 파일에 적용할 수 있습니다.

- 보호 파일: EXE, DLL 파일
- 사용 툴: HSBGen.exe

주의사항

보호하려는 파일에 LMP 1.0 또는 2.0 둘중 하나만 적용할 것을 권장합니다.

기능

실행 파일의 메모리 보호 기능 (LMP 1.0, LMP 2.0 지원)

실행 파일의 메모리에서 코드 영역에 대한 위/변조를 감지하는 기능입니다.

DLL 파일의 메모리 보호 기능 (LMP 1.0, LMP 2.0 지원)

지정한 DLL 파일의 코드 영역에 대해서도 위/변조를 감지하며, 해당 DLL 이 직접 빌드하지 않는 파일이라 해도 보호가 가능합니다.

주의사항

동적으로 로딩되는 DLL 파일을 보호할 경우,

² Themida 패커에서 메모리 보호를 위해 매번 실행될 때마다 서로 다른 메모리 주소의 코드를 변경하기 때문에, 서버에서 메모리 CRC를 체크하는 서버연동 방식과 충돌이 발생한다.

핵심드 초기화(_AhnHS_Initialize)->시작(_AhnHS_StartService)

호출시점 이후에 해당DLL을 로딩한다면 DLL로딩 이후에

반드시 _AhnHS_IsModuleSecure(szDllPath) 함수를 호출해야만 합니다.

메모리 무결성 검증 기능 (LMP 1.0, LMP 2.0 지원)

LMP 기능이 동작하기 전에 메모리를 수정하는 행위를 방지하기 위해 일부 시스템 파일의 후킹여부를 조사하고 메모리에 대해 무결성을 검증하는 기능입니다.

특징

옵션 제공

Initialize 부분에서 옵션에 `AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION` 옵션을 적용을 하면 해당 기능이 실행이 됩니다.

콜백 메시지 제공

해당 기능에 의해 메모리 조작이 감지되면 핵심드 콜백 함수에서 다음과 같은 `AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP` 메시지가 전달됩니다.

테스트 프로그램 제공

Ehsvc에서 제공하는 API를 사용하여 구현한 테스트용 프로그램인 `Amazon.exe`를 제공합니다. `Amazon.exe`는 기존의 핵심드 테스트 기능과 Ehsvc의 테스트 기능을 제공합니다.

시스템 구조(System Architecture)

`CSInspector.exe` 를 이용하여 보호하고자 하는 모듈에 섹션 정보를 입력하여 `HShield.lib`, `EHSvc.dll`을 제공하여 클라이언트에 DLL과 라이브러리 형태로 적용됩니다.

Ehsvc.dll (인터페이스 dll)

해킹 및 에러가 발생했을 경우 모니터링 서버에 알릴수 있도록 기본적인 정보를 세팅하는 API 를 제공합니다.

HShield.lib (핵심드 라이브러리)

해킹 및 에러가 발생했을 경우 모니터링 서버에 알릴수 있도록 기본적인 정보를 세팅하는 API 를 제공합니다.

CSInspector.exe(보호 모듈 설정 유틸) (LMP 1.0)

보호하고자 하는 클라이언트 파일 또는 서드파티 모듈에 섹션 정보를 입력하여 핵심드에서 해당 모듈에 대한 코드 영역을 보호 하도록 해줍니다.

HSBGen.exe(보호 모듈 설정 유틸) (LMP 2.0)

보호하고자 하는 클라이언트 파일에 섹션 정보를 입력하여 핵셴드에서 해당 모듈에 대한 코드 영역을 보호 하도록 해줍니다.

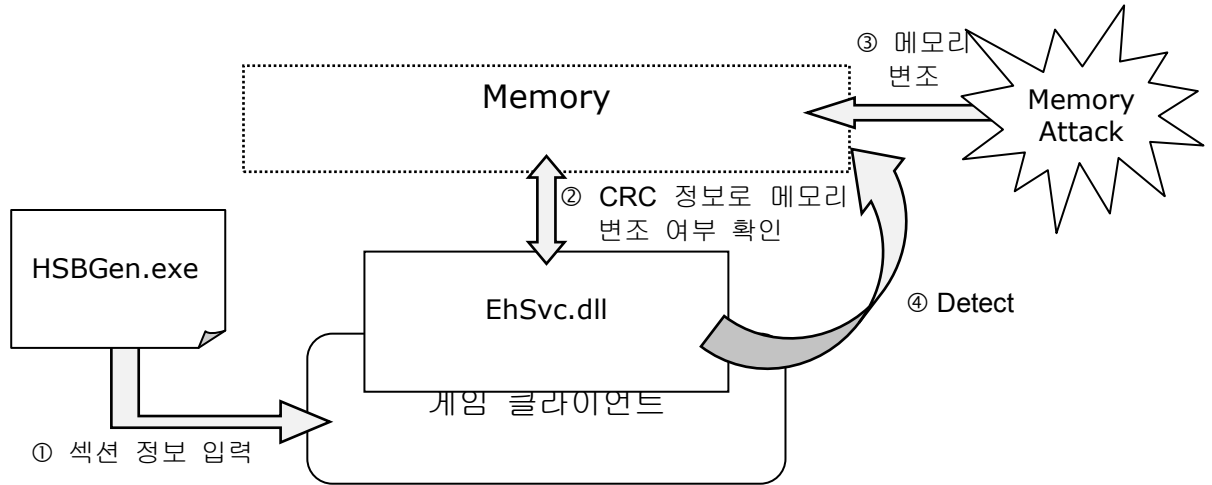


그림 7-1 Local Memory Protection 동작 구조

Local Memory Protection 동작 구조는 다음과 같습니다.

- ④ 핵셴드에서 제공하는 HSBGen.exe 툴을 이용하여 Game Client의 PE 구조 특정 영역에 게임 클라이언트 또는 DLL의 섹션 정보를 저장한다.
- ⑤ 핵셴드가 실행되면 게임 클라이언트 또는 DLL의 PE 구조에서 섹션 정보를 로딩하고 실행 이전에 메모리가 조작되었는지 여부를 체크한다.
- ⑥ 섹션 정보를 바탕으로 현재 메모리의 CRC 정보를 구성한다.
- ⑦ CRC 정보를 바탕으로 주기적으로 메모리 조작 여부를 체크한다.

7.2. Application Programming

Ehsvc에서 제공하는 핵셴드 초기화 함수 (_AhnHS_Initionlize)를 이용해서 LMP 기능을 사용할 수 있습니다.

참고

이 문서에서 사용하고 있는 샘플 코드는 Microsoft Visual C++ 6.0을 기준으로 한 C/C++ 언어로 작성되었습니다. 프로그래밍에 사용되는 언어는 각 프로그램의 특성 및 시스템 환경에 따라서 변경할 수 있습니다.

프로그래밍 적용 방법

LMP기능을 사용하기 전에 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

7.2.1.1.LMP 관련 파일

LMP 관련 파일

표 7-1 LMP 관련 파일

위치	파일 이름	설치 폴더	설명
[SDK]\Include\	HShield.h	[프로그램 소스 폴더]	클라이언트에서 사용할 헤더 파일
[SDK]\Lib	HShield.lib	[프로그램 소스 폴더]	클라이언트에서 사용할 라이브러리 파일,
[SDK]\Bin\Util	CSInspector.exe		보호할 모듈에 설정하는 유틸리티
[SDK]\Bin\Anti Crack\	HSBGen.exe		보호할 모듈에 설정하는 유틸리티

7.2.1.2.적용 방법

클라이언트 적용 방법

1. HShield.lib 파일을 작업할 프로젝트에 포함시킵니다
2. 제공되는 HShield.h 파일을 작업할 소스 파일에 포함시킵니다.
3. 해당 소스에 `_AhnHS_Initialize` 함수를 호출할 때 5번째 파라미터에 다음과 같은 옵션을 추가 시킵니다.

```
// _AhnHS_Initialize 함수 호출에 쓰일 옵션 플래그를 정의합니다 (기존의  
// 옵션에 추가)  
dwOption = AHNHS_CHKOPT_ALL |  
AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION;  
  
// _AhnHS_Initialize 함수를 호출하여 핵샷드 서비스를 초기화 합니다.  
nRet = _AhnHS_Initialize ( szFullFilePath,  
                           HS_CallbackProc,      // 콜백함수  
                           1000,                  // 게임 코드  
                           " B228F291B7D7FAD361D7A4B7" ,  
                           // 라이선스 키  
                           dwOption,  
                           // 옵션 플래그  
                           AHNHS_SPEEDHACK_SENSING_RATIO_NORMAL
```



```

        ...
    );
}

```

4. 핵셴드와 관련하여 이벤트 전달 함수 작성 부분에
AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP 처리를
추가합니다.

```

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

    // 각 경우에 대하여 알맞은 에러 메시지를 출력합니다
    switch ( lCode )
    {
        // LMP Callback
        // 참조: LMP 콜백에서는 변조된 모듈 이름과 페이지상의 주소값도
        // 함께 넘어오지만 이를 사용자에게 직접 노출할 필요는
        // 없습니다.
        case AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP:
            wsprintf(szMsg, "모듈에서메모리
                        변조가 감지되었습니다.\n" );
            MessageBox( NULL, szMsg, szTitle, MB_OK );
            break;

        ....
    }
}

```

파라미터로 전달되는 내용은 다음과 같습니다.

- lCode
AHNHS_ACTAPC_DETECT_MEM_MODIFY_FROM_LMP(0x10705)
- pParam
“조작된 모듈이름(모듈Base주소): 조작된 실제 페이지 주소”
파라미터로 조작이 발생한 모듈의 이름과 페이지 주소 값이
넘어옵니다. 해당 정보는 조작 발생시 정보로 활용하며 사용자에게
보여줄 필요는 없습니다

보호 모듈 적용 - CSInspector.exe (LMP 1.0)

해당 모듈을 패킹해서 배포하는 경우 CSInspector.exe가 적용된 이후에 패커를
적용합니다.

1. LMP 기능을 위하여 제공되는 CSInspector.exe 를 보호하려는 모듈에
사용합니다.
2. CSInspector.exe 툴은 커맨드 라인 방식으로 동작합니다. 커맨드 라인
입력 창에서 다음과 같이 입력합니다.
C:\> CSInspector.exe Target.exe
3. 추가로 보호할 dll 파일이 있다면 동일하게 CSInspector.exe를 적용합니다.

C:\> CSInspector.exe Target.dll

4. SUCCESS 문구가 출력되면 정상적으로 적용된 것입니다.
5. CSInspector.exe를 실행한 이후에 패킹이나 CRC 추출 작업을 진행해 주시기 바랍니다.

참고

CSInspector 에 대한 자세한 사용법을 확인하려면 다음 섹션을 참고하시기 바랍니다. [11.9 CSInspector 툴 \(HackShield 전용, 5.1 이후 버전 \)](#)

보호 모듈 적용 – HSBGen.exe (LMP 2.0)

참고

HSBGen 에 대한 자세한 사용법을 확인하려면 다음 섹션을 참고하시기 바랍니다. [11.5 HSBGen 사용방법](#)

부가기능

AhnHS_IsModuleSecure

CSInspector.exe(LMP1.0), HSBGen.exe(LMP2.0)를 정보를 삽입한 보호대상 DLL 내에 삽입된 정보가 정상적으로 존재하는지 무결성 유무를 확인합니다.

AhnHS_IsModuleSecure 함수를 호출한 예제는 다음과 같습니다.

```
예제
_AhnHS_Initialize (..) ;                // 핵싧드 초기화
_AhnHS_StartService (..);              // 핵싧드 시작

LoadLibrary("C:\\\\GAME\\\\GameEngine.dll"); // Dll-loading

// 반드시 핵싧드가 정상적으로 초기화되고, 보호대상 DLL이 로딩된이후에
// 호출합니다.
BOOL bRet = _AhnHS_IsModuleSecure ("C:\\\\GAME\\\\GameEngine.dll");
```

설명.

bRet 이 TRUE이면, CSInspector 삽입정보가 정상적으로 존재하는 것이며 FALSE일 경우는 변조되거나 존재하지 않는것입니다. 해당 기능은 보호대상 dll을 Dummy형태로 교체하는 유형의 해킹을 감지할 수 있습니다.

AhnHS_IsModuleSecure

DESCRIPTION

LMP기능 사용시,
CSInspector(LMP1.0), HSBGen.exe(LMP2.0) 를 통해 보호대상 DLL내에 삽입된
정보의 무결성 유무를 검증합니다.

SYNTAX

```
BOOL __stdcall _AhnHS_IsModuleSecure (IN LPCSTR szModulePath )
```

PARAMETERS

Parameter	Value	Description
szModulePath	LPCSTR	DLL 경로 (전체 경로)

RETURN VALUE

TRUE

- 설 명 : CSInspector를 통해 삽입한 정보가 유효합니다.
- 원 인 : 정상적인 상황입니다.

FALSE

- 설 명 : CSInspector를 통해 삽입한 정보가 유효하지 않습니다.
- 원 인 : 보호 대상 DLL이 다른 DLL로 교체되었거나 손상되었습니다.
- 확인사항 : CSInspector를 이용해서 해당 DLL내에 정보를 정상적으로
삽입했는지 다시 확인합니다.

Example

_AhnHS_IsModuleSecure 함수를 호출한 예제는 다음과 같습니다.

```
예제  
BOOL bRet = _AhnHS_IsModuleSecure ("C:\\\\GAME\\\\GameEngine.dll");
```

주의사항

- 해당 함수는 핵싧드가 정상적으로 초기화 (_AhnHS_Initialize)된 이후에
호출해야만 합니다.

서버 연동 사용시 주의할 점

CSInspector를 이용하여 구서버연동 및 확장서버연동을 LMP 와 동시에 사용을 하려는 경우에는 다음과 같은 사항에 유의를 하셔야 합니다.

1. 구서버연동

- 게임 클라이언트 파일에 LMP기능을 적용하려고 한다면 반드시 CSInspector.exe 를 사용하고 난 후에 AntiCpSvrTool.exe 를 사용하여 HackShield.crc 를 생성해야 합니다.
- 기존에 서버 연동을 적용 중이라면 해당 모듈에 LMP 기능을 적용하고 난 후 AntiCpSvrTool.exe 를 이용하여 HackShield.crc 를 다시 생성합니다.

2. 확장서버연동 (단, HSBGen [LMP2.0] 을 사용할경우 해당사항 없음)

게임 클라이언트 파일에 LMP기능을 적용하려고 한다면 반드시 CSInspector.exe 를 사용하고 난 후에 HSBGen.exe 를 사용하여 anticpx.hsb 를 생성 해야 합니다.

- 기존에 확장서버연동을 적용 중이라면 해당 모듈에 LMP 기능을 적용하고 난 후 HSBGen.exe 를 이용하여 anticpx.hsb 를 다시 생성합니다.

지원 패커

현재 LMP 기능에서 지원하는 패커는 다음과 같습니다.

표 7-2 LMP 지원 패커

패커 이름	버전	비고
Themida	2.0.6.5	LMP1.0은 ApiWrap기능 지원안됨. LMP 2.0부터 지원가능
Armadillo	V6.4.0.640	패킹 옵션에 따라 부분적으로 지원가능 (※Doc\Additional\Packer_HackShield_호환성.pdf 문서 참조)

8. 부가 기능

8.1. 데이터 파일/메시지 암호화 기능

8.1.1. 개요

HsCryptoUtil은 데이터 암호/복호화 SDK입니다. 최근 컴퓨터의 발달로 인해 더욱 강력한 암호화 기술이 요구되고 있고 그 결과 DES(Data Encryption Standard)보다 진보된 AES(Advanced Encryption Standard)가 표준으로 지정되어 사용되고 있습니다. HsCryptoUtil은 128bit AES를 사용해서 더욱 강력한 데이터 암호/복호화를 제공합니다.

기능

데이터 암호/복호화

파일 및 메시지 암호/복호화 라이브러리는 HsCryptLib.lib(Windows 용)와 libhscrypt.so(Linux 용), HsCryptLib.h를 제공하여 게임 개발사가 간단한 방법으로 메시지 및 파일 암호/복호화 처리를 할 수 있습니다.

특징

인터페이스 함수(API) 제공

HsCryptLib에서 제공하는 기능을 편리하게 사용하고, 그 실행 결과 값을 받아 볼 수 있도록 인터페이스 라이브러리를 제공합니다. 제공되는 인터페이스 라이브러리를 사용하여 개발사는 파일 및 메시지를 손쉽게 암호/복호화할 수 있습니다.

데이터 암호화 프로그램 제공

파일 암호화 툴인 HsCryptoUtil.exe를 제공하며 암호화된 파일을 HsCryptLib에서 제공하는 API를 사용하여 복호화할 수 있습니다.

시스템 구조(System Architecture)

HsCryptLib은 독립된 실행 파일 형태가 아닌 SDK를 이용한 라이브러리 형태로 제공됩니다. HsCryptLib의 전체 구조 및 동작 원리는 다음과 같습니다.

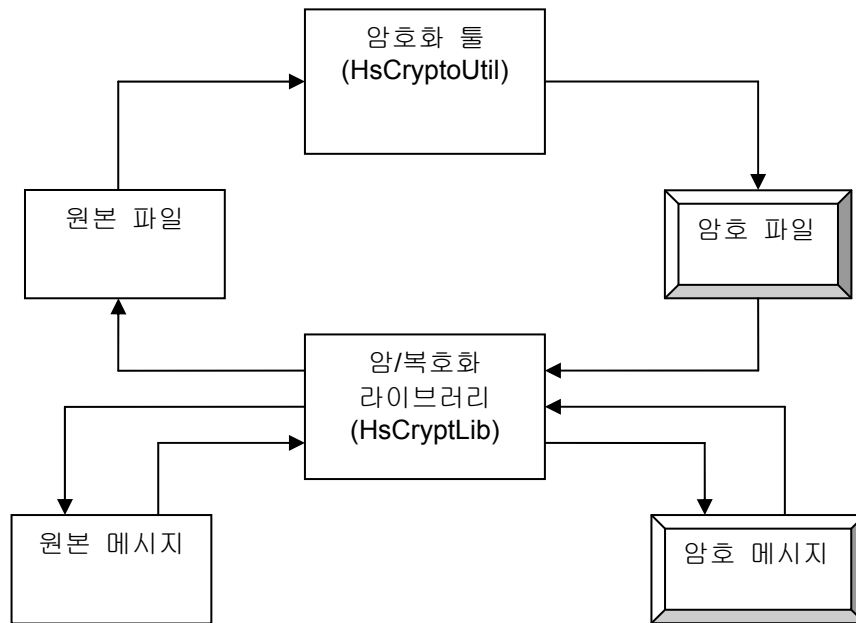


그림 8-1 HsCryptLib의 전체 구조 및 동작 원리

인터페이스 라이브러리: HsCryptLib.lib(Windows 용), libhscrypt.so(Linux 용)

인터페이스 라이브러리 파일로서 데이터를 암/복호화할 때 사용하는 API를 제공합니다.

HsCryptoUtil 프로그램

암호화 프로그램으로 HsCryptLib.lib를 이용하여 파일을 암호화하는 기능을 제공합니다.

8.1.2.Application Programming

프로그래밍 순서

게임 개발사는 다음과 같은 순서로 HsCryptLib 기능을 구현할 수 있습니다.

1. 준비 작업: 제공받은 HsCryptLib 파일 목록을 확인하고, 필요한 파일을 복사합니다.
2. 암호/복호화 초기화 함수 호출: 암호/복호화를 초기화하는 함수를 작성하여 데이터를 암호/복호화하기 전에 호출합니다. 만약 초기화하지 않으면 파일 및 메시지 암호/복호화를 할 수 없습니다.
1. 파일 복호화 함수 호출: 파일이 암호화되었다면 파일을 부분적으로 혹은 전체를 복호화하는 함수 호출 코드를 작성하여 데이터를 특정 부분이나 전체를 복호화할 수 있습니다. 복호화된 데이터는 버퍼로 출력됩니다.
2. 메시지 암호화 함수 호출: 메시지를 암호화하는 함수 호출 코드를 작성하여 메시지를 암호화할 수 있습니다. 암호화된 데이터는 버퍼로 출력됩니다.
3. 메시지 복호화 함수 호출: 메시지를 복호화하는 함수 호출 코드를 작성하여 메시지를 복호화할 수 있습니다. 복호화된 데이터는 버퍼로 출력됩니다.
4. 작성된 소스 코드가 정상적으로 동작하는지 테스트합니다.
5. 사용자에게 배포합니다.

프로그래밍 준비

HsCryptLib을 사용하여 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

HsCryptLib 파일

HsCryptLib 파일

표 8-1 HsCryptLib 파일

파일 이름	설치 폴더	설명
HsCryptLib.h	[프로그램 소스 폴더]	헤더 파일
HsCryptLib.lib	[프로그램 소스 폴더]	윈도우용 라이브러리 파일 : 멀티스레드 및 싱글스레드 라이브러리 제공
libhscrypt.so	[프로그램 소스 폴더]	리눅스용 라이브러리 파일

컴파일러 설정

HsCryptLib을 사용하는 암/복호화 프로그램의 프로젝트 파일에는 반드시 HsCryptLib.lib(libhscrypt.so) 파일을 라이브러리나 소스 코드 목록에 포함시켜야 합니다. 단, Windows 용 HsCryptLib를 사용하는 프로젝트는 멀티스레드 기반 라이브러리를 사용하는지 싱글스레드 기반 라이브러리를 사용하는지 확인하여 적합한 HsCryptLib.lib를 적용합니다.

HsCrypt_InitCrypt

프로그래밍을 하기 위한 준비 작업이 완료되면 먼저 HsCrypt_InitCrypt를 호출합니다. HsCrypt_InitCrypt 함수 호출이 성공적으로 완료되어야 데이터 암/복호화를 하기 위한 실제 암호키 및 복호키가 생성되며 이 암/복호키를 사용하여 암/복호화할 수 있습니다.

HsCrypt_InitCrypt 함수를 호출한 예제는 다음과 같습니다.

예제

```
typedef struct _HSCRYPT_KEYINFO
{
    BYTE byInitKey[HSCRYPTLIB_INITKEY_SIZE]; //초기화키
    BYTE AesEncKey[HSCRYPTLIB_KEY_SIZE];      //암호키
    BYTE AesDecKey[HSCRYPTLIB_KEY_SIZE];      //복호키
} HSCRYPT_KEYINFO, *PHSCRYPT_KEYINFO;

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

dwRet = _HsCrypt_InitCrypt ( &HsKeyInfo );
```

위의 적용 예제에서 HSCRYPT_KEYINFO 구조체를 선언한 다음 byInitKey에 초기화 키 값을 입력합니다. HsKeyInfo.byInitKey에 들어가는 초기화 키 크기는 반드시 16byte로 설정해야 합니다.

HsCrypt_InitCrypt를 호출하면 HSCRYPT_KEYINFO 구조체의 AesEncKey(암호키)와 AesDecKey(복호키)에 실제로 키 값이 할당됩니다. 이 키값을 이용하여 메시지 및 파일 암/복호화를 수행합니다.

주의

초기화 키에 의해 생성된 암/복호화 키 쌍이 일치해야 파일 및 메시지 암/복호화를 할 수 있습니다.

HsCrypt_GetEncMsg

메시지를 암호화하는 함수입니다. HsCrypt_InitCrypt로 암/복호화를 초기화하고 HsCrypt_GetEncMsg를 호출하여 데이터를 암호화할 수 있습니다. 이 때 암호화된 데이터는 버퍼로 출력됩니다.

예제

```
dwRet = _HsCrypt_GetEncMsg (
    byPlainMsg,          // [in] 암호화할 버퍼
    sizeof(byPlainMsg),  // [in] 암호화할 사이즈
    HsKeyInfo.AesEncKey, // [in] 암호화키
    byEncMsg             // [out] 암호화된 버퍼
);
```

참고

암호화하기 전의 메시지 크기와 암호화된 메시지의 크기는 같습니다.

_HsCrypt_GetDecMsg

메시지를 복호화하는 함수입니다. 암/복호화를 초기화하고 _HsCrypt_GetDecMsg를 호출하여 데이터를 복호화할 수 있습니다. 이 때 복호화된 데이터는 버퍼로 출력됩니다.

예제

```
dwRet = _HsCrypt_GetDecMsg (
    byEncMsg,          // [in] 복호화할 버퍼
    sizeof(byEncMsg),  // [in] 복호화할 사이즈
    HsKeyInfo.AesDecKey, // [in] 복호화키
    byDecMsg           // [out] 복호화된 버퍼
);
```

참고

복호화하기 전의 메시지 크기와 복호화된 메시지의 크기는 같습니다.

_HsCrypt_FRead

파일 구조체 포인터를 이용하여 파일을 부분 및 전체 복호화하는 함수입니다. 암/복호화 키를 초기화하고 **fseek** 함수를 호출하여 원하는 파일 포인터로 이동한 다음 데이터를 복호화할 수 있습니다. 이 때 복호화된 데이터는 버퍼로 출력됩니다.

예제

```
dwRet = _HsCrypt_FRead (
    byPlainBuf,          // [out] 복호화된 버퍼
    dwDecSize,          // [in] 복호화할 사이즈
    InputStream,         // [in] 읽을 파일 포인터
    HsKeyInfo.AesDecKey, // [in] 복호키
    &dwReadLen           // [out] 복호화된 사이즈
);
```

```
);
```

블록 암호(Block Cipher)를 사용하여 암호/복호화를 합니다. 복호화를 원하는 부분을 블록으로 나누어서 블록을 복호화하고 필요한 부분만 버퍼에 저장하여 리턴합니다.

파일 전체를 복호화하고 싶다면, **fseek** 함수를 사용하여 파일 구조체 포인터를 파일의 맨 앞으로 설정한 다음 **_HsCrypt_FRead**의 두 번째 파라미터에 파일 전체 크기를 입력하면 됩니다.

주의

대상 파일을 암호화했던 암호키의 쌍인 복호키를 입력해야 하며, 키 관리가 어렵다면 암호/복호키를 초기화했던 초기화키와 **_HsCrypt_InitCrypt**를 이용하여 생성된 복호키를 사용합니다.

8.1.3.Application Programming Interface

_HsCrypt_InitCrypt

DESCRIPTION

암/복호화 함수를 초기화합니다.

SYNTAX

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

PARAMETERS

Parameter	Description
PHSCRYPT_KEYINFO 구조체 정의 typedef struct_HSCRYPT_KEYINFO { BYTE byInitKey[HSCRYPTLIB_INITKEY_SIZE]; BYTE AesEncKey[HSCRYPTLIB_KEY_SIZE]; BYTE AesDecKey[HSCRYPTLIB_KEY_SIZE]; } HSCRYPT_KEYINFO, *PHSCRYPT_KEYINFO;	[in][out]암/복호키 구조체 초기화키(16bytes) 암호키(550bytes) 복호키(550bytes)

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 초기화에 성공했을 때 리턴하는 값입니다
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_INITCRYPT_INVALIDPARAM (Value = 0x0001B002)

- 설 명 : 잘못된 파라미터를 입력했을 때 리턴하는 값입니다
- 원 인 :
- 확인사항 :

기타

WIN32 Defined Error (Value = WIN32 Defined)

REMARKS

데이터를 암호/복호화 하기 위해 반드시 호출해야 하는 함수로서 암호/복호화를 하기 위해 필요한 여러 데이터 값을 설정합니다. **HSCRYPT_KEYINFO** 구조체의 **byInitKey**를 할당한 다음 **_HsCrypt_InitCrypt** 함수를 호출하면 **AesEncKey**(암호키) 및 **AesDecKey**(복호키)를 구할 수 있습니다. 이 키를 이용하여 메시지 및 파일 암호/복호화를 수행합니다.

HsCrypt_GetEncMsg

DESCRIPTION

메시지 암호화를 하고 나서 암호화된 데이터 버퍼로 출력합니다.

SYNTAX

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

PARAMETERS

Parameter	Description
pbyInput	[in] 암호화할 버퍼
nInLength	[in] 암호화할 크기
pAesEncKey	[in] 암호화키
pbyOutput	[out] 암호화된 버퍼

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 메시지 암호화에 성공했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_GETENCMMSG_INVALIDPARAM (Value = 0x0001B003)

- 설 명 : 잘못된 파라미터를 입력했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

기타

WIN32 Defined Error (Value = WIN32 Defined)

HsCrypt_GetDecMsg

DESCRIPTION

메시지를 복호화한 다음 복호화된 데이터 버퍼로 출력합니다.

SYNTAX

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

PARAMETERS

Parameter	Description
pbyInput	[in]복호화할 버퍼
nInLength	[in]복호화할 크기
pAesEncKey	[in]복호화키
pbyOutput	[out] 복호화된 버퍼

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 메시지 암호화에 성공했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_GETDECMMSG_INVALIDPARAM(Value = 0x0001B004)

- 설 명 : 잘못된 파라미터를 입력했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

기타

WIN32 Defined Error (Value = WIN32 Defined)

HsCrypt_FRead

DESCRIPTION

파일에서 원하는 부분만 복호화하여 데이터를 버퍼로 출력합니다.

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]복호화된 버퍼
dwDecryptSize	[in]복호화할 크기
pInputStream	[in]복호화할 파일 구조체 포인터
pAesDecKey	[in]복호키
pdwReadLen	[out]복호화된 크기

RETURN VALUE

ERROR_SUCCESS (Value = 0x00000000)

- 설 명 : 복호화에 성공했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_INVALIDPARAM (Value = 0x0001B005)

- 설 명 : 잘못된 파라미터를 입력했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_GETFILELEN (Value = 0x0001B009)

- 설 명 : 파일 크기 얻기에 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_SIZEZERO (Value = 0x0001B00B)

- 설 명 : 파일 사이즈가 0일 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_GETPOSITION (Value = 0x0001B00A)

- 설 명 : 파일 포인터 현재 위치 얻기에 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_FSEEK (Value = 0x0001B00C)

- 설 명 : 현재 파일 포인터 위치의 블록으로 이동하는 것에 실패했을 때 리턴하는 값입니다
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_DECRYPT_RANGE (Value = 0x0001B006)

- 설 명 : 복호화할 블록 크기가 파일보다 클 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_FREAD_DECRYPT_FREAD (Value = 0x0001B007)

- 설 명 : 파일 읽기에 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

**ERROR_HSCRYPTLIB_FREAD_DECRYPT_GETDECMMSG
(Value = 0x0001B008)**

- 설 명 : 메시지 복호화에 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

ERROR_HSCRYPTLIB_EXCEPTION (Value = 0x0001B001)

- 설 명 : 예외의 경우가 발생했을 때 리턴하는 값입니다.

- 원 인 :
- 확인사항 :

기타

WIN32 Defined Error (Value = WIN32 Defined)

8.2. User 권한 실행 지원 기능

8.2.1. 개요

HsUserUtil은 NT 계열의 운영 체제를 사용하는 경우 관리자 계정이 아닌 일반 사용자 계정으로 로그인한 환경에서 HackShield의 게임 해킹 방어 기능이 정상적으로 동작하도록 하는 여러 기능입니다.

기능

비 관리자 계정 게임 지원

HsUserUtil.lib를 제공하여 게임 클라이언트 및 HackShield를 관리자 계정으로 로그인하지 않고 일반 사용자 계정으로 로그인하더라도 게임을 실행할 수 있고 HackShield의 게임 해킹 방어 기능이 정상적으로 동작하도록 지원합니다.

특징

인터페이스 함수(API) 제공

HsUserUtil에서 제공하는 기능을 편리하게 사용하고, 그 실행 결과 값을 받아 볼 수 있도록 인터페이스 라이브러리를 제공합니다. 제공되는 인터페이스 라이브러리를 사용하여 게임 개발사는 정책에 따라서 게임 클라이언트 및 HackShield를 일반 사용자 계정으로 실행되도록 프로그래밍할 수 있습니다.

테스트 프로그램 제공

HsUserUtil에서 제공하는 API를 사용하여 구현한 테스트용 게임 클라이언트 프로그램인 HsUserUtilTest.exe를 제공합니다. 게임 개발사는 테스트 프로그램을 참고하여 HsUserUtil의 기능 및 예제 코드를 확인하고, 클라이언트 프로그램 개발에 적용할 수 있습니다.

시스템 구조(System Architecture)

HsUserUtil은 독립된 실행 파일 형태가 아닌 SDK를 이용한 라이브러리 형태로 제공됩니다. HsUserUtil의 전체 구조 및 동작 원리는 다음과 같다.

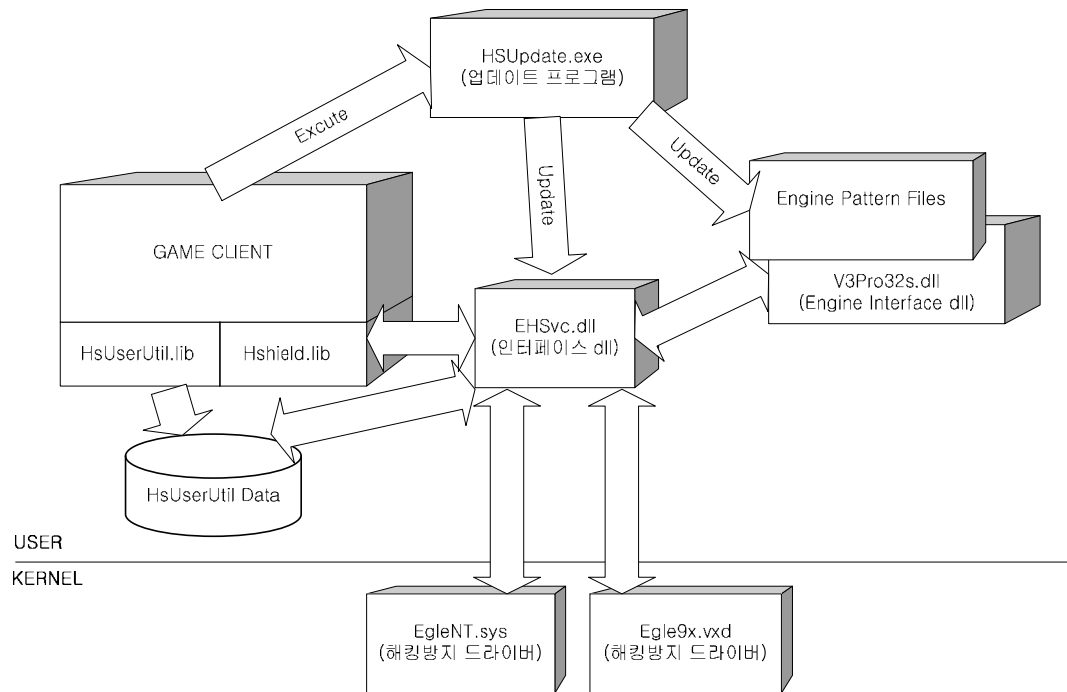


그림 8-2 HsUserUtil의 전체 구조 및 동작 원리

HsUserUtil.lib (인터페이스 라이브러리)

인터페이스 파일로서 게임 프로그램이 일반 사용자 계정으로 동작할 수 있게 하는 API를 제공합니다

HsUserUtil Data

게임 프로그램이 일반 사용자 계정으로 동작할 수 있게 하는 윈도우 계정 정보 및 관련 데이터를 저장합니다.

EhSvc.dll (HackShield 서비스 모듈)

HackShield 서비스 모듈로서 실제 게임 프로그램에 로드되어 해킹 방어 기능을 수행합니다. 일반 사용자 권한으로 게임이 실행되는 경우 HsUserUtil 데이터의 정보를 이용해서 게임이 정상적으로 실행되고 게임 방어 기능이 정상 동작하도록 지원합니다.

8.2.2.Application Programming

프로그래밍 순서

게임 클라이언트 개발자는 다음과 같은 순서로 HsUserUtil 기능을 구현합니다.

참고

이 문서에서는 게임 클라이언트 프로그램에 HsUserUtil을 사용하는 경우를 예로 들어 설명하고 있습니다. 만약 게임 클라이언트 프로그램 외에 게임 런처(launcher) 프로그램이 별도로 존재할 경우에는 HsUserUtil 라이브러리를 게임 런처 프로그램에서 사용할 수도 있습니다. 게임의 구조에 맞게 적용해야 합니다.

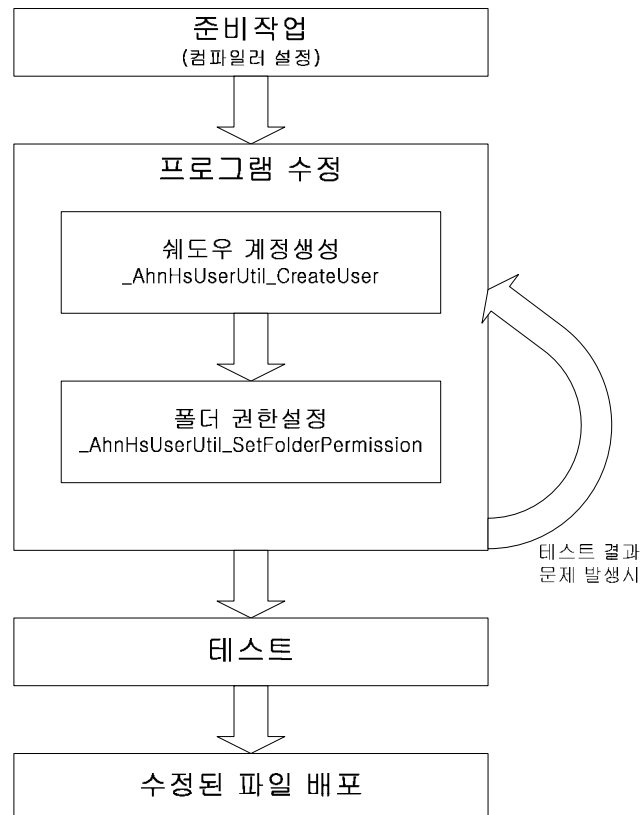


그림 8-3 HsUserUtil 프로그래밍 순서

1. 준비 작업: 제공받은 HsUserUtil 파일 목록을 확인하고, 필요한 파일을 복사합니다.
2. 쉐도우 계정 생성 함수 호출: 쉐도우 계정을 생성하는 함수 호출 코드를 작성하여 일반 사용자 계정으로 게임 실행 및 게임 해킹 방어 기능이

동작하도록 합니다.

3. NTFS 권한 설정 함수 호출: 서비스 시작 함수 호출 후 폴더의 NTFS 권한 설정 함수 호출 코드를 작성하여 일반 사용자 계정에서도 게임에 필요한 파일에 대하여 쓰기 기능이 동작할 수 있도록 합니다.
4. 소스 코드가 정상적으로 동작하는지 테스트합니다.
5. 클라이언트 사용자에게 배포합니다.

프로그래밍 준비

HsUserUtil을 사용하여 프로그래밍을 시작하기 전에 먼저 다음과 같은 준비 작업을 실행합니다.

8.2.2.1.HsUserUtil 파일

HsUserUtil 파일

표 8-2 HsUserUtil 파일

파일 이름	설치 폴더	설명
HsUserUtil.h	[게임 소스 폴더]	헤더 파일
HsUserUtil.lib	[게임 소스 폴더]	라이브러리 파일

컴파일러 설정

HsUserUtil을 사용하는 게임 클라이언트 프로그램의 프로젝트 파일에는 반드시 HsUserUtil.lib 파일을 라이브러리나 소스 코드 목록에 포함시켜야 합니다.

_AhnHsUserUtil_CreateUser

프로그래밍을 하기 위한 준비 작업이 완료되면 먼저 _AhnHsUserUtil_CreateUser를 호출합니다. _AhnHsUserUtil_CreateUser 함수 호출이 성공적으로 완료되어야 일반 사용자 계정에서도 게임을 실행하고 게임 해킹 방어 기능을 수행할 수 있습니다.

이 때 _AhnHsUserUtil_CreateUser 함수는 관리자 계정으로 게임 프로그램 또는 게임 런처 프로그램에서 호출합니다.

_AhnHsUserUtil_CreateUser 함수를 호출한 예제는 다음과 같습니다.

예제

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

참고

_AhnHsUserUtil_CreateUser는 이전에 생성된 쉘도우 계정의 사용자 정보가 없거나 기존에 생성된 쉘도우 계정의 사용자 정보로 로그인되지 않는 경우에만 새로운 사용자를 생성합니다. 또한 쉘도우 계정 명명 규칙에 해당하는 계정을 삭제하는 로직이 포함되어 있어 사용하지 않거나 불필요한 계정은 새로운 쉘도우 계정 생성 시에 자동으로 삭제됩니다.

[_AhnHsUserUtil_SetFolderPermission](#)

게임 프로그램이 **NTFS** 볼륨에 설치된 경우에는 해당 폴더에 대해서 일반 사용자 권한으로는 파일 쓰기 권한이 없어 게임이 정상적으로 동작하지 않을 수 있습니다. 예를 들어 게임 모듈에 대한 업데이트 프로그램이 실행되어도 게임이 설치된 폴더에 최신 파일을 쓰지 못하는 상황이 발생한다든지, 게임 실행 중에 게임 데이터를 저장하는데 실패할 수도 있습니다.

일반 사용자 계정에 대하여 게임 설치 폴더에 파일 쓰기 권한을 부여하려면 [_AhnHsUserUtil_SetFolderPermission](#) 함수를 사용함으로써 **NTFS** 쓰기 권한을 부여할 수 있습니다. 이 함수는 다음 예제와 같이 호출합니다.

예제

```
dwRet = _AhnHsUserUtil_SetFolderPermission("게임이 설치된 경로");
```

일반 사용자 계정에 대한 **NTFS** 쓰기 권한을 부여하기 위해서는 해당 경로를 반드시 전체 경로(Full Path)로 전달해야 하며 상대 경로 또는 올바른지 않은 경로를 전달할 경우에는 오동작할 수 있습니다.

전달된 게임 설치 경로가 **NTFS** 볼륨이 아니거나 **NTFS** 파일 시스템을 사용하지 않는 **Windows 95, 98, ME** 계열의 PC에서는 이 함수가 호출되더라도 아무런 동작을 하지 않습니다.

이 함수가 실행되면 해당 경로에 대해서 **Users** 그룹에 대한 **NTFS** 쓰기 권한을 부여하게 되며, 이를 반영하기 위해서는 반드시 관리자 권한으로 실행된 상태에서 이 함수가 호출되어야 합니다.

주의

게임 클라이언트 프로그램이 실행하는 위치가 사용자에 따라 다르게 설치되므로 게임 개발사는 주의하여 작성합니다. 게임이 C:\W나 바탕 화면, 윈도우 폴더 등 어디에 설치될지 모르는 상황에서 이러한 폴더에 대한 **NTFS** 권한을 변경하는 것은 보안상 취약점이 될 수도 있습니다.

[_AhnHsUserUtil_DeleteUser](#)

[_AhnHsUserUtil_DeleteUser](#) 함수를 사용함으로써 [_AhnHsUserUtil_CreateUser](#)

함수를 통해 생성한 윈도우 계정을 삭제합니다.

예제

```
dwRet = _AhnHsUserUtil_DeleteUser ();
```

AhnHsUserUtil_IsEnableHSAdminRights

현재 로그인된 계정이 핵싯드를 실행할 수 있는 권한을 가지고 있는지 확인하는 함수입니다.

예제

```
dwRet = _AhnHsUserUtil_IsEnableHSAdminRights ();
```

AhnHsUserUtil_CheckHSShadowAccount

AhnHsUserUtil_CreateUser 함수를 통해 생성한 윈도우 계정이 시스템내에 정상적으로 등록되었는지를 확인합니다.

예제

```
DWORD dwRet = HSUSERUTIL_ERR_OK;

dwRet = _AhnHsUserUtil_CheckHSShadowAccount ();

switch ( dwRet )
{
    case HSUSERUTIL_ERR_NOT_NT:
        AfxMessageBox ( "HSUSERUTIL_ERR_NOT_NT" );
        break;
    case HSUSERUTIL_ERR_OK:
        AfxMessageBox ( "HSUSERUTIL_ERR_OK" );
        break;
    case HSUSERUTIL_ERR_SHADOWACNT_NOT_EXIST:
        AfxMessageBox ( "HSUSERUTIL_ERR_SHADOWACNT_NOT_EXIST" );
        break;
    default:
        AfxMessageBox ( "HSUSERUTIL_ERR_UNKNOWN" );
        break;
}
```

윈도우 계정 생성여부에 대해 확인이 필요한 경우 사용하는 함수입니다.

AhnHSUserUtil_IsAdmin

현재 로그인된 계정이 어드민 권한을 가지고 있는지 확인하는 함수입니다.

예제

```
if ( TRUE == _AhnHSUserUtil_IsAdmin() )
```

```

{
    AfxMessageBox ( "TRUE" );
}
else
{
    AfxMessageBox ( "FALSE" );
}

```

8.2.3.Application Programming Interface

AhnHsUserUtil_CreateUser

DESCRIPTION

일반 사용자 권한으로 로그인했을 때 게임 해킹 방어 기능에 사용할 쉘도우 계정을 생성합니다.

SYNTAX

```

DWORD __stdcall
_AhnHsUserUtil_CreateUser ( );

```

PARAMETERS

없음.

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- 설 명 : 쉘도우 계정 생성에 성공했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

HSUSERUTIL_ERR_NOT_ADMIN (Value = 0x0005A002)

- 설 명 : 현재 로그인한 계정이 관리자 계정이 아닌 경우 리턴하는 값입니다
- 원 인 :
- 확인사항 : 적용 방식에 따라 차이가 있겠지만 일반 User 권한에서 AhnHsUserUtil_CreateUser가 호출될 수 있도록 적용이 된 경우에는 이 에러는 처리하지 않도록 해야 합니다.

HSUSERUTIL_ERR_NOT_NT (Value = 0x0005A003)

- 설 명 : NT 계열 시스템이 아닌 경우 리턴하는 값입니다.
- 원 인 :
- 확인사항 : 사용중인 OS 버전이 NT 이상인지 확인해주시기 바랍니다.

HSUSERUTIL_ERR_DELSHADOWACNT_FAIL (Value = 0x0005A005)

- 설 명 : 기존에 생성된 쉘도우 계정 삭제에 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

HSUSERUTIL_ERR_DELHIDEIDINFO_FAIL (Value = 0x0005A006)

- 설 명 : Windows XP 시작 화면에서 쉘도우 계정을 감추기 위한 정보를 삭제하는데 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

HSUSERUTIL_ERR_DELSHADOWACNTINFO_FAIL (Value = 0x0005A007)

- 설 명 : 쉘도우 계정 정보 삭제에 실패했을 때 리턴하는 값입니다
- 원 인 :
- 확인사항 :

HSUSERUTIL_ERR_ADDSHADOWACNT_FAIL (Value = 0x0005A008)

- 설 명 : 쉘도우 계정 생성에 실패했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

REMARKS

_AhnHsUserUtil_CreateUser 함수는 Administrator 권한을 가진 계정으로 로그인한 상태에서 쉘도우 계정이 생성됩니다. 일반 사용자 권한으로 HackShield의 게임 방어 기능을 정상적으로 사용하기 위해서는 최초 한번은 이 함수가 호출되어야 쉘도우 계정이 생성되어야 합니다.

AhnHsUserUtil_SetFolderPermission

DESCRIPTION

게임 클라이언트가 설치된 디렉터리에 일반 사용자 계정에 대한 NTFS 쓰기 권한을 부여함으로써 일반 사용자 계정으로 로그인하여도 게임 클라이언트가 업데이트 및 실행에 필요한 파일 쓰기 작업 등을 수행할 수 있습니다.

SYNTAX

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

PARAMETERS

Parameter	Value	Description
szPath		NTFS 권한을 설정할 경로의 전체 경로 예제: C:\Program Files\My Company\My Game

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- 설 명 : 해당 폴더에 대하여 NTFS 권한 부여에 성공했을 때 리턴하는 값입니다.
- 원 인 :
- 확인사항 :

HSUSERUTIL_ERR_NOT_ADMIN (Value = 0x0005A002)

- 설 명 : 현재 로그인한 계정이 관리자 계정이 아닌 경우 리턴하는 값입니다
- 원 인 :
- 확인사항 : 적용 방식에 따라 차이가 있겠지만 일반 User 권한에서 _AhnHsUserUtil_CreateUser가 호출될 수 있도록 적용이 된 경우에는 이 에러는 처리하지 않도록 해야 합니다.

HSUSERUTIL_ERR_SETFLDRPERMISSION_FAIL (Value = 0x0005A10A)

- 설 명 : 해당 폴더에 대하여 NTFS 권한 부여에 실패했을 때 리턴하는 값입니다.
- 원 인 :

- 확인사항 :

HSUSERUTIL_ERR_GETGROUPSID_FAIL (Value = 0x0005A10B)

- 설명 : 그룹 SID 얻기에 실패했을 때 리턴하는 값입니다.
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_GETSECINFO_FAIL (Value = 0x0005A10C)

- 설명 : SD와 DACL 정보 얻기에 실패했을 때 리턴하는 값입니다.
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_ADDNEWACE_FAIL (Value = 0x0005A10D)

- 설명 : 새 ACE 생성에 실패했을 때 리턴하는 값입니다.
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_ADDNEWDACL_FAIL (Value = 0x0005A10E)

- 설명 : 새 DACL 생성에 실패했을 때 리턴하는 값입니다.
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_COPYOLDDACL_FAIL (Value = 0x0005A10F)

- 설명 : 새 DACL에 기존 DACL를 복사하는데 실패했을 때 리턴하는 값입니다.
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_ADDNEWACETONEWDACL_FAIL (Value = 0x0005A110)

- 설명 : 새 DACL에 새 ACE를 추가하는데 실패했을 때 리턴하는 값입니다.
- 원인 :

- 확인사항 :

REMARKS

게임 클라이언트 프로그램이 설치되는 위치가 사용자에게 의해서 임의로 지정이 가능한 경우에는 이 함수를 실행할 때 유의해야 합니다. 만약 사용자가 C나 D 드라이브와 같은 루트 디렉터리에 게임을 설치했다면 이 함수가 호출되면 해당 드라이브 전체에 대해서 NTFS 권한이 부여됩니다. 이는 보안상의 취약점이 될 수도 있으므로 주의해서 호출해야 합니다.

지정된 폴더 하위에 많은 수의 하위 폴더와 파일이 존재할 경우 이 함수가 호출되면 NTFS 권한을 설정하는데 수 초에서 수 분의 시간이 소요될 수도 있습니다. 하지만 이것은 최초에 NTFS 권한을 부여하는 과정에서만 발생할 수 있으며 이미 권한이 부여된 이후에는 영향을 주지 않습니다.

_AhnHsUserUtil_DeleteUser

DESCRIPTION

_AhnHsUserUtil_CreateUser를 통해 생성한 윈도우 계정을 삭제합니다.

SYNTAX

```
DWORD __stdcall _AhnHsUserUtil_DeleteUser (void);
```

PARAMETERS

Parameter	Value	Description
void		

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- 설명 : 윈도우 계정 삭제에 성공했을 때 리턴하는 값입니다.
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_LOADDLL_FAIL (Value = 0x0005A004)

- 설명 : 해당 기능 수행에 필요한 DLL이 로드되지 못해 발생하는 오류입니다.
- 원인 :

- 확인사항 :

HSUSERUTIL_ERR_NOT_NT (Value = 0x0005A003)

- 설명 : NT 계열 시스템이 아닌 경우 리턴하는 값입니다.
- 원인 :
- 확인사항 : OS 계열을 확인합니다.

REMARKS

_AhnHsUserUtil_DeleteUser 함수는
_AhnHsUserUtil_CreateUser에서 생성된 쉘도우 계정을 삭제하는 함수입니다.

AhnHsUserUtil_IsEnableHSAdminRights

DESCRIPTION

현재 로그인된 계정이 핵셸드가 동작할 수 있는 권한을 가지고 있는지 확인하는 함수입니다.

(Admin 권한을 가지고 있는 경우 이거나, 유저 권한이면서 핵셸드 쉘도우 계정이 생성되어 있는 경우에 성공을 리턴합니다.)

SYNTAX

```
DWORD __stdcall _AhnHsUserUtil_IsEnableHSAdminRights(void);
```

PARAMETERS

Parameter	Value	Description
void		

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- 설명 : 쉘도우계정이 정상적으로 등록된 경우 리턴하는 값입니다
- 원인 :
- 확인사항 :

HSUSERUTIL_ERR_NOT_NT (Value = 0x0005A003)

- 설 명 : NT 계열 시스템이 아닌 경우 리턴하는 값입니다.
- 원 인 :
- 확인사항 : OS 계열을 확인합니다.

REMARKS

현재 로그인된 계정이 핵셴드가 동작할 수 있는 권한을 가지고 있는지 확인하는 함수 이기 때문에, Admin 권한을 가지고 있는 경우 이거나, 유저 권한이면서 핵셴드 쉼도우 계정이 생성되어 있는 경우에 성공을 리턴하고 그렇지 않다면 다른 에러 값을 리턴합니다.

단순히 핵셴드 쉼도우 계정이 존재하는지 확인하기 위해서는 `_AhnHsUserUtil_CheckHSShadowAccount` 함수를 사용하시기 바랍니다.

`_AhnHsUserUtil_CheckHSShadowAccount`

DESCRIPTION

`_AhnHsUserUtil_CreateUser`를 통해 생성한 쉼도우 계정이 정상적으로 등록되었는지 확인합니다.

SYNTAX

```
DWORD __stdcall _AhnHsUserUtil_CheckHSShadowAccount ( );
```

PARAMETERS

Parameter	Value	Description
void		

RETURN VALUE

HSUSERUTIL_ERR_OK (Value = 0x00000000)

- 설 명 : 쉼도우계정이 정상적으로 등록된 경우 리턴하는 값입니다
- 원 인 :
- 확인사항 :

HSUSERUTIL_ERR_NOT_NT (Value = 0x0005A003)

- 설 명 : NT 계열 시스템이 아닌 경우 리턴하는 값입니다.

- 원 인 :
- 확인사항 : OS 계열을 확인합니다.

HSUSERUTIL_ERR_SHADOWACNT_NOT_EXIST (Value = 0x0005A009)

- 설 명 : 핵섀드 쉐도우 계정이 존재하지 않습니다.
- 원 인 :
- 확인사항 : `_AhnHsUserUtil_CreateUser` 함수를 통해서 핵섀드 쉐도우 계정이 정상적으로 생성되었는지 확인합니다.

REMARKS

`_AhnHsUserUtil_CheckHSShadowAccount` 함수는, `_AhnHsUserUtil_CreateUser`에서 생성된 쉐도우 계정이 정상적으로 등록 되었는지를 확인하는 함수입니다.

`_AhnHSUserUtil_IsAdmin`

DESCRIPTION

현재 로그인된 계정이 어드민 권한이 있는지 확인한다.

SYNTAX

```
BOOL __stdcall _AhnHSUserUtil_IsAdmin ();
```

PARAMETERS

Parameter	Value	Description
void		

RETURN VALUE

TRUE

NT 계열이 아니거나, NT 계열이고, ADMIN 권한을 가지고 있습니다.

FALSE

NT 계열이고, 어드민 권한을 가지고 있지 않습니다.

REMARKS

9. 툴 사용 방법

9.1. AntiCpSvr 툴

기능

서버에서 CRC 정보 파일(HackShield.crc)을 생성하여 게임 파일 및 메모리, HackShield의 무결성 여부를 판단할 수 있습니다. 또한 서버를 다시 시작하지 않고 새로운 버전을 패치하거나 옵션을 설정해서 이전 버전에 대한 연결 컨트롤이 가능하도록 합니다.

생성된 CRC 정보 파일은 생성할 때마다 다르므로 게임 서버가 여러 개이면 서버마다 CRC 정보 파일을 다르게 적용시켜 향상된 보안을 유지합니다. 하지만 게임 클라이언트가 동일하다면 메모리 정보 데이터는 모두 동일해야 합니다.

9.2. AntiCpSvr 툴 사용 방법

UI 수동 설정 기반의 CRC 정보 파일 생성

1. **CRC File Path**를 입력합니다. [...] 버튼을 눌러서 HackShield.crc 파일을 생성할 경로를 선택합니다. 선택한 파일의 경로가 나타납니다.

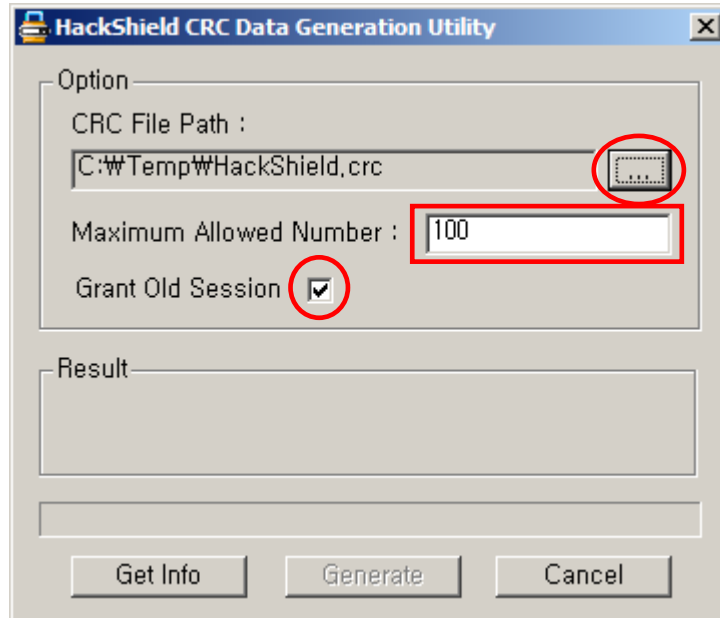


그림 9-1 HackShield CRC 정보 생성 툴

2. **Maximum Allowed Number**를 입력합니다. 새롭게 생성된 CRC 파일이 서버에 적용되었을 때 클라이언트가 이전 버전으로 서버에 연결을 시도할 경우 최대 몇 개의 이전 버전까지 허용할지를 입력합니다. 예를 들어 이 값이 1이면 현재 패치되는 최신 버전만 접근을 허용하며, 2이면 현재 패치되는 버전과 이전 버전으로 접근하는 클라이언트는 모두 허용하고 나머지는 허용하지 않습니다.
3. **Grant Old Session**을 선택할지 결정합니다. 새롭게 생성된 CRC 파일이 서버에 적용되었을 때 이미 연결되어 있던 클라이언트 중 **Maximum Allowed Number**를 벗어나는 버전을 가진 클라이언트에 대해서 연결을 허용할지를 선택합니다. **Grant Old Session**을 선택하면 연결을 허용하는 것이며, 선택을 하지 않으면 새로운 버전으로 패치를 할 경우 기존에 연결되어 있던 클라이언트들 중에 **Maximum Allowed Number**를 벗어나는 버전을 사용하는 클라이언트들은 연결이 끊기게 됩니다.
4. **GetInfo**를 눌러 정보를 가져올 준비를 합니다.
5. 새롭게 패치할 게임을 실행시켜 게임 클라이언트 내에서 `_AhnHS_SaveFuncAddress` 함수가 호출되는 지점까지 수행한 다음

Generate 버튼이 활성화될 때까지 게임을 진행합니다.

6. **Generate** 버튼이 활성화되면 실행하고 있던 게임을 종료하고 게임이 완전히 종료되면 2~3초 대기 후 **Generate**를 눌러 **HackShield.crc** 파일을 생성합니다. 이 때 새롭게 패치되는 게임의 정보를 **HackShield** 모듈인 **Ehsvc.dll**에 저장하게 되므로 반드시 게임을 종료한 다음 **Generate**를 눌러야 합니다.

HackShield.crc 파일이 생성되면 해당 **CRC** 파일을 서버에 패치한 다음 **CRC** 파일을 만드는데 사용된 게임 클라이언트 파일을 패치하도록 합니다.

자동 CRC 정보 파일 생성

AntiCpSvrTool은, **CRC**파일 생성에 필요한 인자정보를 지정함으로써, **CRC** 파일을 자동으로 생성하는 기능을 제공합니다. 사용 방법은 아래와 같습니다.

사용법:>

AntiCpSvrTool.exe **CRC**정보를 추출할 게임의 경로, 생성될 **CRC**파일의 경로, **Grant Old Session**(1=true 0=false), **Maximum Allowed Number**

주의

인자정보는 ' , ' 단위로 구분됩니다

예:>

Game.exe의 **CRC**정보를 추출하여 **Grant Old Session = true / Maximum Allowed Number = 5**로 지정하여 **HackShield.crc**파일을 생성할 경우 아래와 같이 실행합니다.

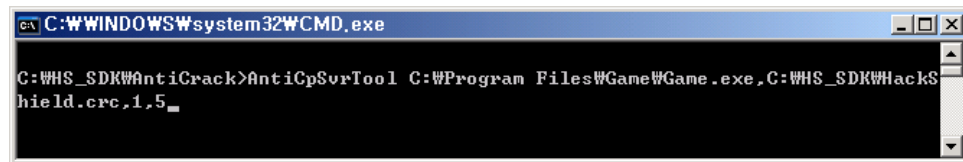


그림 9-2 Command-line 방식의 **AntiCpSvrTool.exe**

9.3. HSBGen 툴(HackShield 전용, 4.2 이후 버전)

기능

서버에서 HackShield Briefcase 파일(AntiCpX.hsb)을 생성하여 게임 파일 및 메모리, HackShield의 무결성 여부를 판단할 수 있습니다. 또한 서버를 다시 시작하지 않고 새로운 버전을 패치하거나 옵션을 설정해서 이전 버전에 대한 연결 컨트롤이 가능하도록 합니다.

생성된 HSB 정보 파일은 생성할 때마다 다르므로 게임 서버가 여러 개이면 서버마다 HSB 정보 파일을 다르게 적용시켜 향상된 보안을 유지합니다. 하지만 게임 클라이언트가 동일하다면 메모리 정보 데이터는 모두 동일해야 합니다.

향상된 LMP 기능을 이용하기 위하여 배포되는 게임 클라이언트 파일에 필요한 관련 정보를 추가합니다. 이전의 CSInspector.exe 툴을 이용하여 제공하던 부분을 HSBGen 툴을 이용하여 해당 기능을 제공합니다.

9.4. HSBGen 툴 사용 방법

UI 수동 설정 기반의 HSB 정보 파일 생성

\Bin\AntiCrack\HSBGen.exe를 이용하여 AntiCpX.hsb파일을 생성합니다.

주의

클라이언트 실행 파일은 패킹되기 이전의 원본 릴리즈 파일이어야 합니다.

HSBGen.exe를 통해 AntiCpX.hsb 파일을 생성하는 방법은 다음과 같습니다.

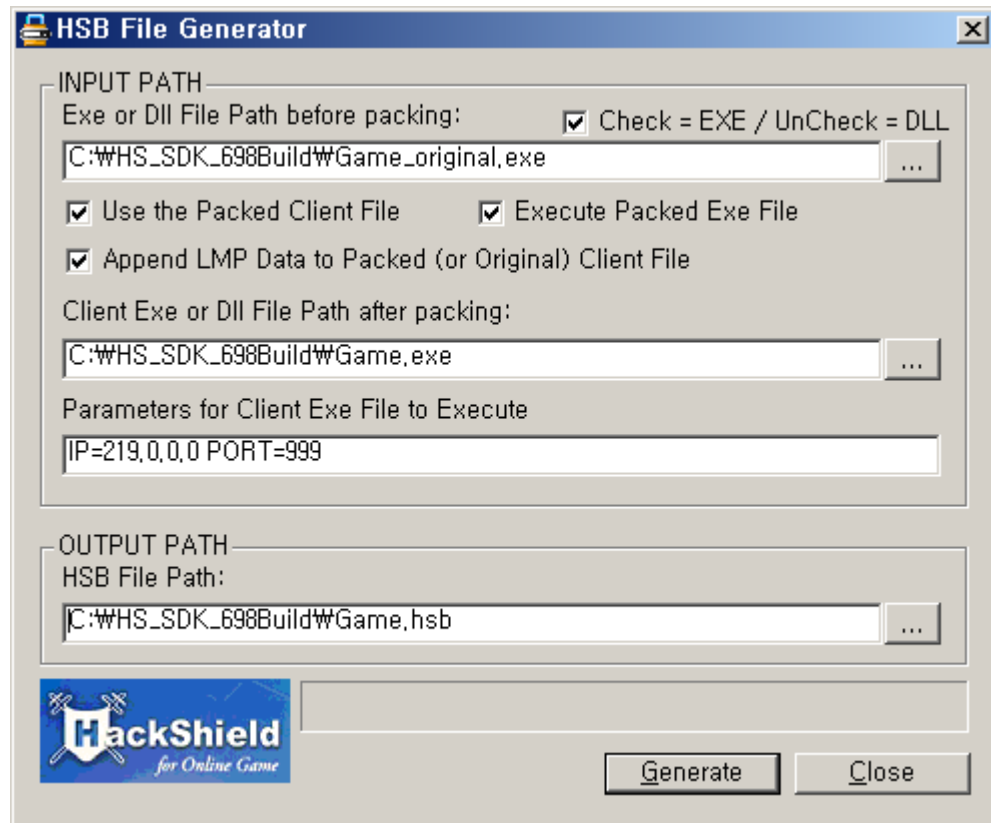


그림 9-3 HSB File Generator

1. 대상파일이 EXE인 경우 Check, DLL인 경우는 UnCheck를 하시기 바랍니다.

주의.

: 대상파일이 DLL인 경우, 대상파일내에 LMP정보가 자동으로 추가됩니다.

2. (주)안철수연구소에서 별도로 전달한 **HSBGen.ini** 파일이 있을 경우 **HSBGen.exe**와 동일한 폴더에 저장합니다. (게임별로 최적화하여 확장 서버 연동의 설정 값을 다르게 적용할 수 있습니다)
3. **Exe or Dll File Path before packing** 입력 상자에 패킹되지 않은 원본 실행 파일 경로를 설정합니다. [...] 버튼을 눌러 파일을 선택합니다.
4. 패킹된 클라이언트 실행파일을 배포할 계획이라면, 앞서 설명한 첫번째 필드에 패킹되기 이전의 파일 경로를 입력합니다.
5. **Use the Packed Client File** 항목을 선택하고 **Client Exe or Dll File Path after packing** 입력 상자에 패킹된 파일 경로를 입력합니다.
6. **Execute Packed Client File**은 Client Executable File Path after packing에서 입력한 실행파일을 자동으로 실행시킨 후 hsb파일을 생성할것인지를 결정합니다.

주의.

- Executable Packed Client File의 옵션을 사용하실 경우, hsb파일 생성버튼을 누르면 게임이 진행될 수 있도록 파라미터 정보 또는 게임 실행시 필요한 파일을 준비하여 게임이 실제로 실행될 수 있는 환경이 준비되어야 합니다.

7. **Parameters for Client Executable File to Execute** 항목은 Execute Packed Client File에서 입력된 실행 파일을 실행시에 필요로 하는 값들이 있다면 해당 값들을 본 항목에 입력하시기 바랍니다.
8. **Append LMP Data to Packed (or Original) Client File** 항목은 LMP기능을 이용하고자 할 경우, 배포되는 게임 클라이언트 파일에 LMP 정보를 추가합니다. 해당 LMP 정보는 패킹되지 않은 원본 실행 파일에 추가하거나 패킹된 클라이언트 실행 파일 둘 다 가능합니다.

주의.

대상파일이 패킹된 경우 execute clientfile 항목을 선택해야만 합니다.

6. **HSB File Path** 입력 상자에 **AntiCpX.hsb** 파일이 생성될 Full Path를 설정합니다. [...] 버튼을 누르면 파일을 생성하는 창이 열립니다. (참고: 실행 파일과 같은 곳을 설정하면, 클라이언트 파일과 같이 배포될 수 있다는 경고창이 나타납니다. 이것은 올바른 메시지입니다.)
7. 모든 설정이 완료되면 **Generate** 버튼을 누릅니다. 진행 상황이 프로그레스 막대에 표시되며, 완료와 함께 정상적으로 완료되었다는 메시지가 나타납니다.

anticpx.hsb 파일이 생성되면 해당 anticpx.hsb 파일을 서버에 패치한 다음

anticpx.hsb 파일을 만드는데 사용된 게임 클라이언트 파일을 패치하도록 합니다.

자동 HSB 정보 파일 생성

HSBGen은, HSB파일 생성에 필요한 인자정보를 지정함으로써, HSB 파일을 자동으로 생성하는 기능을 제공합니다. 사용 방법은 아래와 같습니다.

사용법:>

HSBGen.exe (1),(2),(3),(4),(5),(6)

- (1): HSB정보를 추출할 게임의 경로
- (2): 패키징된 실행파일 지원여부(1 = 지원 / 0 = 지원X)
- (3): HSB정보를 추출할 패키징된 게임의 경로 ((2)가 1일경우만 사용)
- (4): 패키징된 실행 파일 실행 여부(1 = 실행 / 0 = 실행X)
- (5): 패키징된 파일 실행 파라미터 ((4)가 1인 경우에만 사용)
- (6): HSB파일 경로

주의

인자정보는 ' , ' 단위로 구분되며, 구분자 앞뒤로 공백은 허용하지 않습니다.

예1:>

패키징되지 않은 게임일 경우,
게임의 실행 경로가 C:\HS_SDK\Game.exe 이며
HSB 파일이 생성되길 원하는 경로가
C:\HS_SDK\AntiCrack\anticpx.hsb일 경우 아래와 같이 실행합니다.



그림 9-4 Command-line 방식의 HSBGen.exe (일반 파일의 경우)

예2:>

패키징된 게임의 경우,
패키징되지 않은 원본 게임의 실행 경로가 C:\HS_SDK\Game.exe 이며,
패키징된 게임의 실행 경로가 C:\HS_SDK\Game_packed.exe 이고,
HSB 파일이 생성되길 원하는 경로가
C:\HS_SDK\AntiCrack\anticpx.hsb일 경우 아래와 같이 실행합니다.



그림 9-5 Command-line 방식의 HSBGen.exe (패킹된 파일의 경우)

예3:>

패킹된 게임을 실행하는 경우,
패킹되지 않은 원본 게임의 실행 경로가 C:\HS_SDK\Game.exe 이며,
패킹된 게임의 실행 경로가 C:\HS_SDK\Game_packed.exe 이고,
HSB 파일이 생성되길 원하는 경로가
C:\HS_SDK\AntiCrack\anticpx.hsb이며,
패킹된 게임 클라이언트를 실행 하는 경우 아래와 같이 실행합니다.

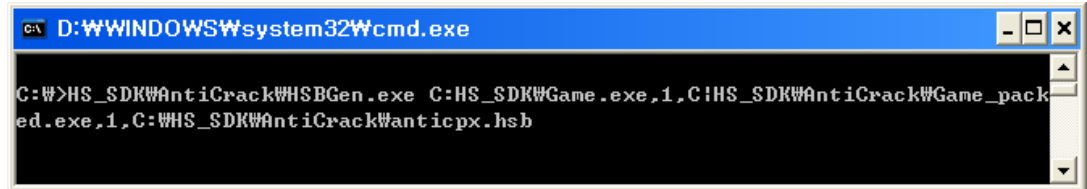


그림 9-6 Command-line 방식의 HSBGen.exe(패킹된 파일을 실행 하는 경우)

예4:>

패킹된 게임을 실행하는 경우,
패킹되지 않은 원본 게임의 실행 경로가 C:\HS_SDK\Game.exe 이며,
패킹된 게임의 실행 경로가 C:\HS_SDK\Game_packed.exe 이고,
HSB 파일이 생성되길 원하는 경로가
C:\HS_SDK\AntiCrack\anticpx.hsb 이며,
패킹된 게임 클라이언트를 실행 하며,
게임 클라이언트 실행 시 파라미터가 필요한 경우 아래와 같이 실행합니다.



그림 9-7 Command-line 방식의 HSBGen.exe(패킹된 파일을 실행 시 파라미터가 필요한 경우)

HSBGen.ini 설명

```
[VERCT]

GrantOldSession=1
MaxAllowedNumber=1

[REQRE]
InitStep1=1
InitStep2=2
InitStep3=8
```

```

InitStep4=4

[DELAY]
UseDelayedSpiking=0
MinDelayCount=1
MaxDelayCount=3

[CKMEM]
PageGroupSize=40
QueryPages=10

[FPATH]
ClientFileName=D:\W Wgame_ori.exe
UsePackedClientFile=1
PackedClientFileName=D:\W Wgame_packed.exe
GetInfoFromRunningEXE=1
Parameters=-d
AppendLMPInfoToClientFile=1
HsbFileName=D:\W Wanticpx.hsb
UseEXE=0

[CKHSB]
UseHSB=1

```

1. VERCT

GrantOldSession

위 값은 1 또는 0 값을 가질 수 있습니다. '1'이라면 현재 **hsb** 파일에 맞는 클라이언트 버전 이외에도 전에 올렸던 **hsb** 파일에 대한 클라이언트 버전도 허용하겠다는 의미 입니다.

MaxAllowedNumber

GrantOldSession이 1일때만 의미가 있는 값이고 항상 1이상의 값을 가져야 합니다. 예를 들어 '1'이라고 하면 현재 적용된 **hsb**에 대한 클라이언트에만 허용하겠다는 의미이고, '5'라면 이전에 적용된 **hsb**파일에 대한 클라이언트 버전을 4개까지 허용하겠다는 의미 입니다.

2. REQRE

서버연동을 할 때 검사하는 기능은 **GUID** 검사, 클라이언트 파일 검사, 핵쉴드 모듈 검사, 메모리 검사, 핵쉴드 동작 상태 검사 이렇게 총 5가지가 있습니다.

```

GUID 검사 : 1
클라이언트 파일 검사 : 2
메모리 검사 : 4
핵쉴드 엔진검사 : 8
핵쉴드 동작 상태 검사: 16

```

각 검사 옵션은 1 ,2 ,4 ,8 ,16 의 비트로 이루어져 있으며 중복해서 사용이 가능합니다.

InitStep 1

GUID(1) 만 검사 혹은 GUID(1) 검사와 핵셴드 동작 상태 검사(16) 의 동시 검사(17) 만 가능합니다.

InitStep 2

GUID 검사(1) 및 클라이언트 파일 검사(2) 를 사용할 수 있습니다.

InitStep 3

모든 검사를 사용할 수 있습니다.

예를 들어 '31' 값을 준다면 모든 검사를 사용하겠다는 의미이고 '5'를 준다면 메모리 검사와 핵셴드 모듈 검사를 하겠다는 의미 입니다.

InitStep 4

위 3개의 스텝이 완료가 되고 주기적으로 반복하게될 검사를 지정합니다. 보통 메모리 검사 및 핵셴드 동작 상태 검사 옵션인 '20' 을 사용하도록 합니다.

3. DELAY

UseDelayedSpiking

'1'을 사용하여 기능을 사용할 경우 `_AhnHS_VerifyResponse` 함수에서 에러를 리턴할 때 바로 에러를 발생하지 않고 일정 딜레이를 발생시켜 해커들에게 혼란을 야기시키게 합니다. '0'일 경우 위 기능을 사용하지 않습니다.

MinDelayCount, MaxDelayCount

딜레이 스파이킹 기능이 적용될 경우 `MinDelayCount` 와 `MaxDelayCount` 사이에 랜덤한 값으로 딜레이가 적용됩니다. 1 은 함수가 호출되는 하나의 주기를 의미합니다.

4. CKMEM

PageGroupSize , QueryPages

메모리 검사할 때 한번에 검사되는 메모리 페이지 수를 지정하는 옵션입니다. 디폴트로 사용할 것을 권장합니다.

5. FPATH

HSBGen.exe 툴을 통하여 사용자가 입력한 정보로, 게임 클라이언트의 메모리 CRC 정보 파일인 .hsb 파일 생성을 위한 게임 클라이언트 설정 정보입니다.

ClientFileName

패킹 되지 않은 게임 클라이언트가 위치한 경로입니다.

UsePackedClientFile

패킹한 게임 클라이언트를 사용할 지 여부에 대한 설정입니다.

패킹한 게임 클라이언트를 사용한다면 이 값은 1이며, 패킹한 게임 클라이언트를 사용하지 않는다면 이 값은 0입니다.

PackedClientFileName

패킹된 게임 클라이언트가 위치한 경로입니다.

UsePackedClientFile 값이 1로 설정된 경우엔 반드시 필요한 정보입니다.

GetInfoFromRunningEXE

패킹된 게임 클라이언트를 사용하는 경우, 게임 클라이언트를 실행시켜 메모리 CRC 정보를 추출하기 위한 설정 값입니다.

이 값이 1이면 PackedClientFileName에 설정된 경로의 파일을 실행시켜 메모리 CRC 정보를 추출하게 됩니다.

Parameters

게임 클라이언트 실행 시 필요한 파라미터 정보입니다.

PackedClientFileName에 경로가 설정되고, GetInfoFromRunningEXE 값이 1인 경우, 이 값이 유효합니다.

AppendLMPInfoToClientFile

LMP 기능을 이용하기 위하여 필요한 관련 정보를 게임 클라이언트 파일에 추가하기 위한 설정 값입니다.

해당 값이 '1'이면 LMP 기능을 이용하기 위하여 필요한 관련 정보를 게임 클라이언트 파일에 추가하고, 해당 값이 '0'이면 작업을 진행하지 않습니다.

HsbFileName

게임 클라이언트의 메모리 CRC 정보 파일(.hsb 파일)이 생성 될 경로입니다. 이 값을 설정하지 않으면 .hsb 파일을 생성할 수 없습니다. (반드시 .hsb 확장자로 설정 해야 합니다.)

UseEXE

대상파일이 EXE일 경우는 1, DLL인 경우는 0으로 설정합니다.

주의.

SDK에 있는 기본hsbgen.ini파일을 open시에는 FPATH경로가 없습니다. 최초 hsb파일을 생성시부터 해당 FPATH정보가 저장됩니다.

6. CKHSB

UseHSB

신규 게임 클라이언트 파일 배포시에, 해당 게임 클라이언트 파일을 대상으로 하여 HSBGen.exe 툴을 실행하여 생성된 HSB 파일을 게임 서버에 업로드할지의 여부에 대한 설정 값입니다.

값이 '0' (UseHSB=0) 이면 '신규 게임 클라이언트 파일을 대상으로 생성된 HSB 파일을 게임 서버에 업로드하지 않고서 확장서버 연동하는 것'을 의미하며, 해당 값이 '1' (UseHSB=1) 이면 '신규 게임 클라이언트 파일을 대상으로 생성된 HSB 파일을 게임 서버에 업로드하고서 확장서버 연동하는 것'을 의미한다.

[CKHSB]섹션내의 UseHSB=0 인 경우, 기능 사용시의 주의사항

위의 섹션 내용이 적용된 HSB 파일을 (최초 한번) 서버에 반드시 업로드되어야 함.

신규 게임 클라이언트 파일 배포 전에,
HSBGen 툴을 이용하여 정상적으로 HSB 파일이 생성이 되었는지 확인 필요함
(단, 해당 HSB 파일을 게임 서버에 업로드 할 필요는 없음)

해당 HSB 파일을 생성시에 HSBGen.ini 파일내의 내용이 이후에 접속되는
게임 클라이언트에 계속 반영되게 되므로, 최초 생성시에 HSBGen.ini 파일내의
내용에 대해서 주의하시기 바랍니다.

‘ 클라이언트 파일 검사’ , ‘ 메모리 검사’ 는 지원되지 않습니다.
[REQRE] 섹션내의 ‘ InitStep2’ , ‘ InitStep3’ , ‘ InitStep4’ 내에
‘ 클라이언트 파일 검사’ 값 및 ‘ 메모리 검사’ 값을 의미하는 숫자
‘ 2’ , ‘ 4’ 값은 빼고 적용되어야 함.

```
Ex> InitStep1=1  
      InitStep2=1  
      InitStep3=8  
      InitStep4=1
```

주의

HSBGen.exe 툴을 이용하여 새롭게 생성된 HSB 파일의 내용이 적용되는
시점은 해당 HSB 파일을 생성시에 입력되었던 게임 클라이언트 파일이 서버로
접속이 된 이후입니다.

즉, 새롭게 생성된 HSB 파일을 게임서버에 업로드하였다고 해서 해당 내역이
반영되는 것이 아니라, HSB 파일을 생성시에 입력되었던 해당 게임
클라이언트로 최초 한 번은 게임 서버에 접속이 되어야 적용이 됩니다.

9.5. HSUpSetEnv 툴(HackShield 전용, 5.1 이후 버전)

기능

업데이트 환경 설정 파일 (HSUpdate.env) 을 생성하는 프로그램입니다.
FTP/ HTTP 방식을 선택할 수 있으며 다중 URL을 지원합니다..

9.6. HSUpSetEnv 툴 사용 방법

HSUpdate.env 파일 생성

[HackShield SDK]\Bin\Util\HSUpSetEnv.exe 를 사용하여 HSUpdate.env 파일을 생성합니다. 사용방법은 아래와 같습니다.

사용법:>

[HackShield SDK]\Bin\Util\HSUpSetEnv.exe 파일을 실행합니다.

주의

HSUpSetEnv.exe 는 업데이트 설정파일을 생성하는 툴입니다. 이 툴은 배포되어선 안됩니다

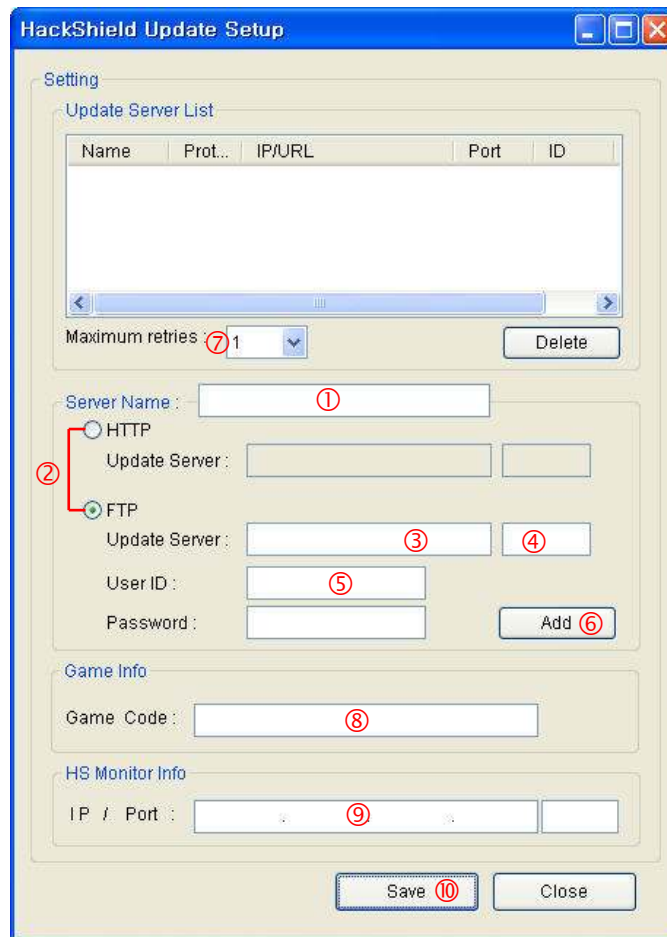


그림 9-8 HSUpSetEnv.exe

- ① 업데이트 서버이름을 입력합니다.
- ② 업데이트 서버에서 사용할 프로토콜에 따라 HTTP나 FTP 여부를 체크합니다

- 다.
- ③ 서버 주소를 입력합니다. **PatchSet**을 저장한 폴더의 위치까지를 지정합니다.
 - ④ 앞서 설정한 포트 번호를 입력합니다.
 - ⑤ **FTP**의 경우, 계정 정보를 설정합니다. 입력하지 않으면 **anonymous** 로 접속합니다.
 - ⑥ **Add** 버튼을 누르면 상단의 **Server List**에 입력한 서버가 추가됩니다. 업데이트 서버를 둘 이상 운영하는 경우 계속해서 서버리스트를 추가합니다.
 - ⑦ 업데이트가 실패할 경우 재시도 횟수를 설정합니다. 전체 리스트의 접속이 실패한다면 설정한 재 접속 횟수만큼 시도하도록 합니다.
 - ⑧ **Game Code** 를 입력합니다. **_AhnHS_HSUpdateEx** 함수를 사용한 경우에만 유효한 정보입니다. **_AhnHS_HSUpdate** 를 사용하실 때는 입력할 필요가 없습니다.
 - ⑨ **HackShield Monitor** 기능을 사용하는 경우 모니터링 **IP / Port** 정보를 입력합니다. (**IP / Port** 정보만 입력됩니다. 모니터링 기능을 사용하시려면, 별도로 **_AhnHS_StartMonitor** 함수를 호출해야합니다.)
 - ⑩ **Save** 버튼을 누르면 아래와 같은 화면과 함께 **HSUpdate.env** 파일이 생성됩니다.



설정이 완료되면 툴을 종료합니다.

9.7. CSInspector 툴

기능

특정 EXE파일이나 DLL파일을 보호 대상으로 지정합니다. 대상파일이 패킹되기 이전에 실행해야 하며, 서버연동 기능과 함께 사용할 때에는 AntiCpSvrTool.exe이나 HSBGen.exe보다 먼저 실행되어야 하는 툴입니다. 현재는 커맨드라인 방식으로만 제공됩니다.

클라이언트에서 AHNHS_CHKOPT_LOCAL_MEMORY_PROTECTION 옵션과 함께 사용되어야 합니다.

대상 파일을 보호 대상으로 지정

\\Bin\\Util\\CSInspector.exe를 이용하여 특정 파일을 보호 대상으로 지정합니다. EXE파일과 DLL파일을 지정 가능합니다. 사용방법은 아래와 같습니다.

사용법:>

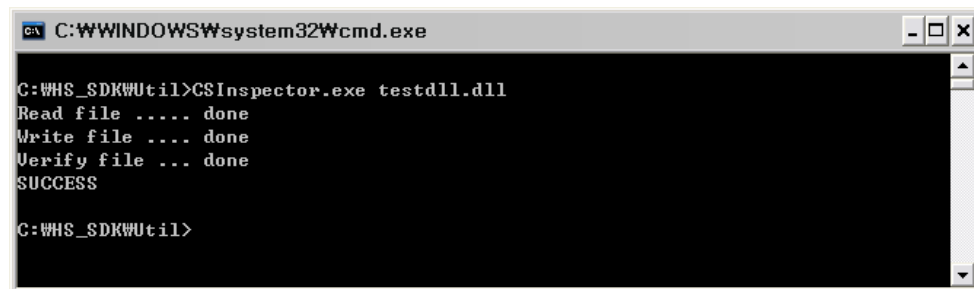
CSInspector.exe 대상파일

주의

패킹이나 CRC 생성 작업보다 선행되어야 합니다.

예:>

대상 파일이 testdll.dll 이고, CSInspector.exe가 C:\\HS_SDK\\Util 폴더 안에 존재한다면, 아래와 같이 실행합니다



```
C:\WINDOWS\system32\cmd.exe

C:\HS_SDK\Util>CSInspector.exe testdll.dll
Read file ..... done
Write file .... done
Verify file ... done
SUCCESS

C:\HS_SDK\Util>
```

그림 9-9 CSInspector.exe

9.8. SetServerList 툴(HackShield 전용, 5.1 이후 버전)

기능

ANTIFREESERVER 기능을 사용하기 위해서는 유효한 IP 주소를 지정해주어야 합니다. SetServerList 툴에서 이러한 유효한 IP 주소를 지정해 줄 수 있습니다. ANTIFREESERVER 옵션을 사용하고 SetServerList 툴로 IP 주소를 지정해 주면 게임 프로세스에서 지정하지 않은 IP 주소를 접속하려 하면 콜백이 발생합니다. 또한 로컬(127.0.0.1)로 접속하는 경우에도 콜백이 발생합니다.

클라이언트에서 AHNHS_CHKOPT_ANTIFREESERVER 옵션과 함께 사용되어야 합니다.

9.9. SetServerList 툴 사용 방법

afs.dat 파일 생성

[HackShield SDK]\Bin\Util\SetServerList.exe 를 사용하여 afs.dat 파일을 생성합니다. 사용방법은 아래와 같습니다.

사용법:>

[HackShield SDK]\Bin\Util\SetServerList.exe 파일을 실행합니다.

주의

SetServerList.exe 는 업데이트 설정파일을 생성하는 툴입니다. 이 툴은 배포되어선 안됩니다

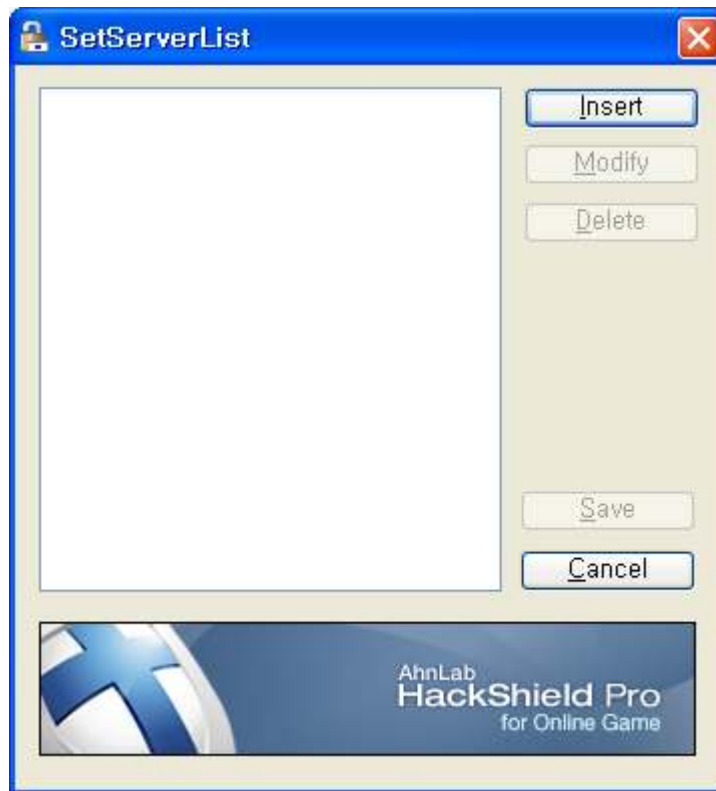


그림 9-10 SetServerList.exe

- ① “Insert” 버튼을 클릭하면 IP 주소를 입력하는 팝업 창이 생성됩니다.

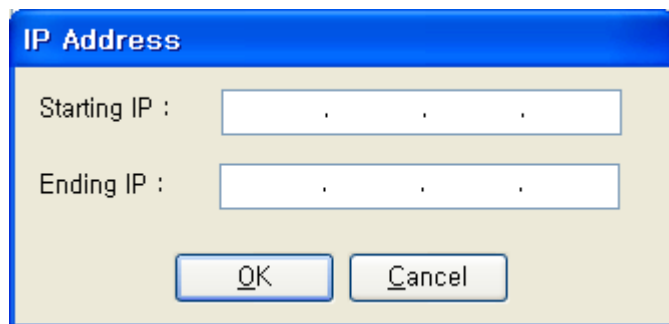


그림 9-11 SetServerList 툴의 주소 입력 창

- ② 유효한 **IP** 주소의 범위를 지정할 수 있습니다. 시작 주소와 끝 주소를 입력합니다. 만약 하나의 **IP** 만 추가하려면 시작 주소와 끝 주소를 동일하게 입력합니다.
- ③ **OK** 버튼을 클릭하면 ②에서 입력한 **IP** 주소가 추가된 것을 볼 수 있습니다.
- ④ 리스트에서 **IP** 주소를 더블 클릭하거나, 선택 후 **Modify** 버튼을 클릭하면 수정할 수 있습니다.
- ⑤ 리스트에서 **IP** 주소를 선택하고 **Delete** 버튼을 클릭하면 해당 주소가 삭제됩니다.
- ⑥ **Save** 버튼을 클릭하면 지정한 **IP** 주소 값들이 **afs.dat** 파일에 저장되고 프로그램이 종료됩니다. 저장하지 않으면 수정 사항이 반영되지 않으므로 반드시 수정 후 저장 버튼을 클릭해야 합니다.
- ⑦ **afs.dat** 파일은 배포시 핵셴드 모듈과 동일한 위치에 배포되어야 합니다.
(예. [Game Directory]/hshield/afs.dat)
- ⑧ **SerServerList.exe** 는 배포되지 않도록 주의해 주시기 바랍니다.

afs.dat 파일 배포

afs.dat 파일은 게임사에서 관리 및 배포하는 것을 원칙으로 합니다.

다만, 게임사에서 **afs.dat** 파일에 대한 배포가 불가능할 경우를 대비하여 핵셴드 업데이트를 통하여 배포할 수 있도록 지원하고 있습니다.

(핵셴드 업데이트를 통하여 **afs.dat** 파일을 배포하는 경우에도 **afs.dat** 파일은 게임사에서 관리하여야 합니다.)

업데이트 서버에서 핵셴드 패치셋 하위(**ahn.ui**, **autoup.exe** 등과 동일한 위치)에 **afs.dat** 파일을 위치시키면 핵셴드 업데이트 시 자동으로 다운로드 하게 됩니다.

주의

핵셴드 업데이트 구조에 대해서는 매뉴얼의 다음 섹션을 참고하세요.

[핵셴드 업데이트 기능] - [시스템 구조(System Architecture)]

9.10.HSBHelper 툴 사용 방법

기능

\Bin\AntiCrack\HSBHelper를 이용하여 HSB 파일과 그에 상응하는 게임 클라이언트 파일이 맞는지를 확인합니다. 사용방법은 아래와 같습니다.

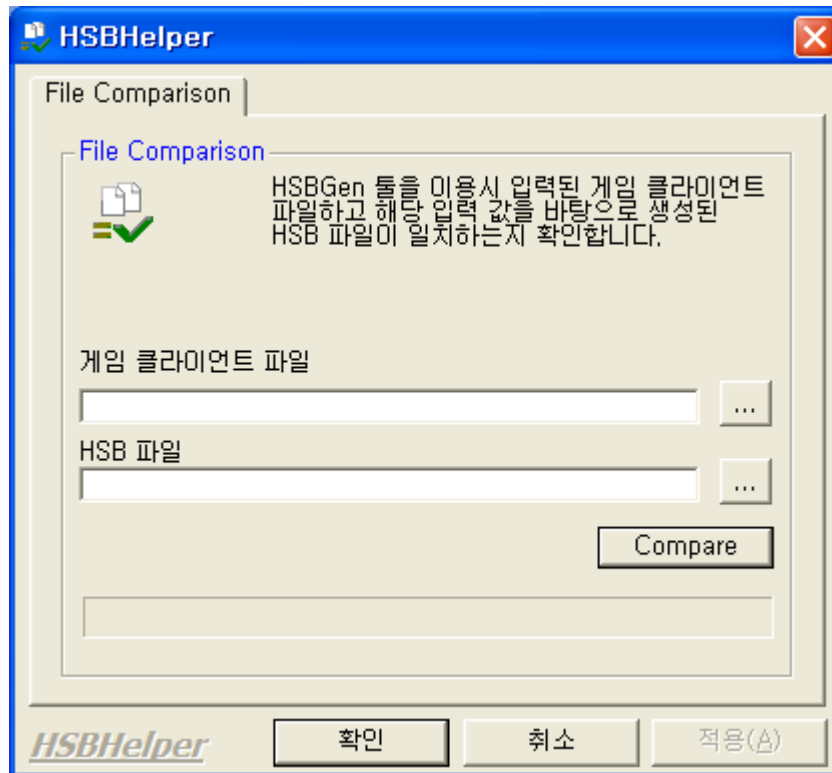
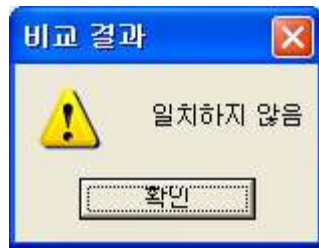


그림 9-12 HSBHelper 툴

- ① 게임 클라이언트 파일이 위치한 전체 경로명을 입력합니다.
(ex. C:\R.5.1.41.1(build 671)\Bin\Amazon.exe)
- ② HSB 파일이 위치한 전체 경로명을 입력합니다.
(ex. C:\R.5.1.41.1(build 671)\Bin\AntiCrack\AntiCpX.hsb)
- ③ 'Compare' 버튼을 클릭한다.
- ④ 일치하는 경우에는 다음과 같은 메시지 박스가 보여집니다.



- ⑤ 일치하지 않는 경우에는 다음과 같은 메시지 박스가 보여집니다.



주의

HSBHelper 실행시, HSBHelper.log 로그 파일이 항상 생성됩니다.

10.부록

10.1.FAQ

현재 시스템에 **SoftICE**가 설치되어 있습니다. **HackShield** 초기화가 정상적으로 이루어지지 않는데, 어떻게 해야 하나요? 또는 개발 및 디버깅 작업을 하는 경우 초기화 옵션을 어떻게 설정합니까?

개발 과정이나, 테스트 과정에서 게임 클라이언트를 디버깅 하기 위해서는 개발자용 **HShield.lib**, **Ehsvc.dll**, **HShield.dat**을 사용하셔야 합니다.

(**Ehsvc.dll** 과 **HShield.dat** 파일이 릴리즈 버전과 섞이면 서버연동시 문제가 발생할 수 있으니 유의하시기 바랍니다.)

핵실드 Developer 버전 위치

- ✓ EhSvc.dll - \SDK\Korean(kr)-SDK\Developer\HShield
- ✓ Hshield.dat - \SDK\Korean(kr)-SDK\Developer\HShield
(서버연동 적용 시(후) 디버깅 시 서버에 복사 필요)
- ✓ HShield.lib - \SDK\Korean(kr)-SDK\Developer\Lib
(게임 클라이언트 프로젝트의 컴파일 항목에 맞는 적절한 라이브러리 파일 사용)

주의

VC++ 컴파일러의 exception 셋팅이 Microsoft C++ Exception : Stop always로 설정 되어 있다면 다음과 같은 에러가 발생할 수 있습니다.
"First chance exception in Game.exe(KERNEL32.dll) 0xE0607063 Microsoft C++ Corporation"

VC++ 컴파일러의 exception 셋팅이 Microsoft C++ Exception : Stop always로 설정 되어 있는지 확인하시고 해당 설정 값을 Stop if not handled 로 변경하시기 바랍니다.

Windows Vista 에서 **Manifest** 기능을 사용하려고 하면 어떻게 해야합니까?

참고 (Manifest 기능)

Vista에서는 **Administrators** 권한이 있는 경우에만 정상적으로 수행될 수 있는 어플리케이션을 만들기 위해서, **manifest**를 이용하여 '이 프로그램을 수행하기 위해서는 반드시 권한상승이 필요하다'는 정보를 실행파일에 포함시킬 필요가

있습니다. 물론, 오른쪽마우스를 클릭해서 "관리자 권한으로 실행"으로 어플리케이션을 수행하거나, '속성'에서 '관리자 권한으로 이 프로그램 실행'을 선택할 수도 있겠지만, 사용자가 이렇게 어플리케이션을 수행하는 것은 상당히 번거로운 일이므로, 실행파일 자체에 '이 어플리케이션은 반드시 Administrators permission이 필요하다' 라는 정보를 추가하여, 자동적으로 권한상승이 이뤄지도록 하는 것을 권장해 드립니다.

manifest파일내용(xml형식)을 게임리소스에 적용하는 방법

- ① visual Studio6.0을 실행합니다.
- ② 게임리소스 폴더에 Resource를 추가합니다.
- ③ resource 항목의 new custom resource(Resource type) 에 "24"를 입력합니다.

참고

"24"는 Microsoft에서 define한 manifest resource value입니다.)

- ④ 아래의 xml 내용을 위 스텝에서 추가된 IDR_DEFAULT1에 복사
를 합니다.

```
<?xml version="1.0" encoding="utf-8" ?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1"
manifestVersion=
"1.0">
<assemblyIdentity version="1.0.0.0"
processorArchitecture="X86"
name="Game"
type="win32" />
<description>HspL</description>
<trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
<security>
<requestedPrivileges>
<requestedExecutionLevel level="requireAdministrator"
/>
</requestedPrivileges>
</security>
</trustInfo>
</assembly>
```

- ⑤ 복사 후 아래 표시 된 스크린샷 처럼 바이너리와 ASCII 코드의
xml문서 내용을 확인할 수 있습니다..

000000	3C 3F 78 6D 6C 20 76 65	72 73 69 6F 6E 3D 22 31	<?xml version="1
000010	2E 30 22 20 65 6E 63 6F	64 69 6E 67 3D 22 75 74	.0" encoding="ut
000020	66 2D 38 22 20 3F 3E 0D	0A 3C 61 73 73 65 6D 62	f-8" ?>.<assemb
000030	6C 79 20 78 6D 6C 6E 73	3D 22 75 72 6E 3A 73 63	ly xmlns="urn:sc
000040	68 65 6D 61 73 2D 6D 69	63 72 6F 73 6F 66 74 2D	hemas-microsoft-
000050	63 6F 6D 3A 61 73 6D 2E	76 31 22 20 6D 61 6E 69	com:asm.v1" mani
000060	66 65 73 74 56 65 72 73	69 6F 6E 3D 22 31 2E 30	festVersion="1.0
000070	22 3E 0D 0A 3C 61 73 73	65 6D 62 6C 79 49 64 65	">.<assemblyIde
000080	6E 74 69 74 79 20 76 65	72 73 69 6F 6E 3D 22 31	ntity version="1
000090	2E 30 2E 30 2E 30 22 20	0D 0A 20 20 20 20 70 72	.0.0.0" .. pr
0000a0	6F 63 65 73 73 6F 72 41	72 63 68 69 74 65 63 74	rocessorArchitect
0000b0	75 72 65 3D 22 58 38 36	22 0D 0A 20 20 20 20 6E	ure="X86" .. n
0000c0	61 6D 65 3D 22 4D 69 6E	69 41 22 0D 0A 20 20 20	ame="MiniA" ..
0000d0	20 74 79 70 65 3D 22 77	69 6E 33 32 22 20 2F 3E	type="win32" />
0000e0	20 0D 0A 20 20 3C 64 65	73 63 72 69 70 74 69 6F	.. <descriptio
0000f0	6E 3E 4D 69 6E 69 41 3C	2F 64 65 73 63 72 69 70	n>MiniA</descrip
000100	74 69 6F 6E 3E 0D 0A 20	20 3C 74 72 75 73 74 49	tion>.. <trustI
000110	6E 66 6F 20 78 6D 6C 6E	73 3D 22 75 72 6E 3A 73	nfo xmlns="urn:s
000120	63 68 65 6D 61 73 2D 6D	69 63 72 6F 73 6F 66 74	chemas-microsoft
000130	2D 63 6F 6D 3A 61 73 6D	2E 76 33 22 3E 0D 0A 20	-com:asm.v3">..
000140	20 20 20 3C 73 65 63 75	72 69 74 79 3E 0D 0A 20	<security>..
000150	20 20 20 20 20 3C 72 65	71 75 65 73 74 65 64 50	<requestedP
000160	72 69 76 69 6C 65 67 65	73 3E 0D 0A 20 20 20 20	rivileges>..
000170	20 20 20 20 20 3C 72	65 71 75 65 73 74 65 64	<requested
000180	45 78 65 63 75 74 69 6F	6E 4C 65 76 65 6C 20 6C	ExecutionLevel 1
000190	65 76 65 6C 3D 22 72 65	71 75 69 72 65 41 64 6D	evel="requireAdm
0001a0	69 6E 69 73 74 72 61 74	6F 72 22 20 20 2F 3E 0D	inistrator" />.
0001b0	0A 20 20 20 20 20 3C	2F 72 65 71 75 65 73 74	. </request
0001c0	65 64 50 72 69 76 69 6C	65 67 65 73 3E 0D 0A 20	edPrivileges>..
0001d0	20 20 20 3C 2F 73 65 63	75 72 69 74 79 3E 0D 0A	</security>..
0001e0	20 20 3C 2F 74 72 75 73	74 49 6E 66 6F 3E 0D 0A	</trustInfo>..
0001f0	3C 2F 61 73 73 65 6D 62	6C 79 3E	</assembly>

⑥ 새로 생성 된 리소스 파일인 IDR_DEFAULT1을 오른쪽 클릭 후 properties 메뉴 클릭 합니다.

⑦ “IDR_DEFAULT1”을 “1”로 변경 합니다.

참고

“24 manifest기능을 사용하려면, 아이디 값이 “1”로 설정되어야 합니다.

⑧ Save 및 게임 리소스를 재빌드 합니다.

참고 (Manifest 기능을 사용해야 하는 이유)

가령 해커 혹은 일반 유저가 악의의 목적으로 해킹툴을 관리자 권한으로 실행시키면 해킹툴은 유저레벨이 아닌 관리자권한에서 동작을 하게 되고, 이 때 게임이 유저권한에서 동작한다면, 게임은 하위레벨에서 동작하게 되어 해킹툴에 대한 방어가 불가능합니다. (낮은 권한의 어플리케이션은 상위권한의 어플리케이션에 대한 액세스를 취할 수가 없습니다.) 또한, Vista에서는 프로세스 실행 중에는 권한상승을 할 수가 없다는 것도 한가지 이유라고 할 수 있습니다.

관리자권한의 해킹툴에 대한 방어를 위해서, 핵실프 또한 반드시 관리자 권한에서 실행되어야 합니다. 핵실프는 이러한 이유로 Vista에서 윈도우계정생성이 불필요하기때, manifest기능의 추가가 요청 되어 집니다.

HackShield Update 서버 구축은 어떻게 할까?

Windows 2000 또는 Windows XP에서 Administrator 권한이 아닌 사용자로 시스템에 로그인하면 HackShield를 사용할 수 있습니까?

HackShield에서는 커널 레벨에서 해킹 차단 드라이버를 구동시킵니다. 따라서 기본 설정으로는 Administrator 권한이 아닌 사용자는 HackShield를 구동시킬 수 없습니다. 관리자 계정이 아닌 일반 사용자 계정으로 HackShield의 게임 해킹 방어 기능을 실행하기 위해서는 HackShield에서 사용할 쉘도우 계정(shadow account)를 생성하는 작업이 필요합니다.

게임 서비스 중에서 여러 가지 원인으로 HackShield 종료 함수(_AhnHS_StopService, _AhnHS_Uninitialize)가 호출되지 않고 게임 프로그램이 비정상적으로 종료되었습니다. 게임 프로그램을 다시 시작하면 HackShield가 다시 실행됩니까?

HackShield 종료 함수를 호출하지 않고 게임 프로그램을 종료하면 해킹 차단 드라이버가 시스템에서 언로드되지 않은 상태가 됩니다. 그러나 이 때 HackShield 초기화 함수를 호출하여 게임 프로그램을 다시 실행하면, 자동으로 기존 드라이버를 강제로 언로드시키고 새 드라이버를 로딩시켜 정상적으로 초기화를 시켜줍니다.

새로운 해킹 툴이 발견되었을 경우 이를 차단하려면 어떻게 해야 합니까?

새로운 해킹 툴이 발견되면 다음과 같은 정보를 ㈜안철수연구소로 전달합니다. 전달된 정보를 분석하여 매주 발생하는 정기 엔진 업데이트 또는 긴급 엔진에 적용될 수 있도록 처리하겠습니다.

해킹 툴이 동작하는 게임

동작 운영 체제 버전

해킹 툴에 대한 간략한 설명

해킹 툴 실행 파일

10.2. 색인

-	AhnHS_Uninitialize	43
_AhnHS_HSUpdate	AHNHS_ACTAPC_DETECT_ALREADY HOOKED	58, 62 35
_AhnHS_MakeAckMsg	AHNHS_ACTAPC_DETECT_AUTOMAC RO	122 36
_AhnHS_MakeGuidAckMsg	AHNHS_ACTAPC_DETECT_AUTOMOU SE	91, 119 35
_AhnHS_PauseService	AHNHS_ACTAPC_DETECT_DRIVERFA ILED	44 36
_AhnHS_ResumeService	AHNHS_ACTAPC_DETECT_HOOKFUN CTION	46 35
_AhnHS_SaveFuncAddress	AHNHS_ACTAPC_DETECT_KDTRACE	124 36
_AhnHsUserUtil_CreateUser	AHNHS_ACTAPC_DETECT_KDTRACE CHANGED	156, 159 37, 38
_AhnHsUserUtil_SetFolderPermission	AHNHS_ACTAPC_DETECT_SPEEDHA CK	138, 157, 158, 161, 163, 164, 165 36
_AntiCpSvr_Initialize	AHNHS_ALLOW_CSRSS_OPENPROC ESS	74, 102, 130 27
_AntiCpSvr_AnalyzeAckMsg	AHNHS_ALLOW_LSASS_OPENPROCE SS	83, 85, 115 27
_AntiCpSvr_AnalyzeGuidAckMsg	AHNHS_ALLOW_SVCHOST_OPENPR OCESS	79, 107 26
_AntiCpSvr_Finalize	AHNHS_ALLOW_SWITCH_WINDOW	76, 104, 132 27
_AntiCpSvr_MakeGuidReqMsg	AHNHS_CHKOPT_ALL	77, 105 19, 26
_AntiCpSvr_MakeReqMsg	AHNHS_CHKOPT_ANTIFREESERVER	80, 112 26
_HsCrypt_FRead	AHNHS_CHKOPT_AUTOMOUSE	145, 151 26
_HsCrypt_GetDecMsg	AHNHS_CHKOPT_DETECT_VIRTUAL_ MACHINE	145, 150 29
_HsCrypt_GetEncMsg	AHNHS_CHKOPT_KDTARCE	144, 149 26
_HsCrypt_InitCrypt	AHNHS_CHKOPT_LOCAL_MEMORY_ PROTECTION	144, 147 26
ㄷ	AHNHS_CHKOPT_OPENPROCESS	
데이터 암호화 프로그램	AHNHS_CHKOPT_PROCESSSCAN	141 26
ㅎ	AHNHS_CHKOPT_PROTECT_D3DX	
확장 서버 연동(AntiCpX)	AHNHS_CHKOPT_READWRITEPROCE SSMEMORY	66 26
A	AHNHS_CHKOPT_SELF_DESTRUCTIO N	
ACTAPCPARAM_DETECT_AUTOMOU SE	AHNHS_CHKOPT_SPEEDHACK	35 26
ACTAPCPARAM_DETECT_HOOKFUNC TION	AHNHS_CHKOPT_STANDALONE	35 27
AHNHS	AHNHS_DISPLAY_HACKSHIELD_LOG O	
_SPEEDHACK_SENSING_RATIO_HI GH	AHNHS_DISPLAY_HACKSHIELD_TRAY ICON	29 29
AHNHS	AHNHS_DONOT_TERMINATE_PROCE SS	
_SPEEDHACK_SENSING_RATIO_HI GHEST	AHNHS_ENGINE_DETECT_GAME_HA CK	29 34
AHNHS	AhnHS_Initialize	
_SPEEDHACK_SENSING_RATIO_LO W	AhnHS_StartService	29 38
AHNHS		
_SPEEDHACK_SENSING_RATIO_LO WEST		
AHNHS		
_SPEEDHACK_SENSING_RATIO_NO RMAL		
AhnHS_StopService		

AHNHS_USE_LOG_FILE	26
AHNLAB_DEFINED_ERROR_CODE	116
AntiCPSvr.dll.....	67, 96, 127, 134
AntiCpSvrTool.exe.....	67, 96

E

Eagle9x.vxd or EagleNT.sys	14
EhSvc.dll.....	154
EHSvc.dll	13
ERROR_ANTICPXSVR_CLIENT_FILE_	
ATTACK (0xE904000B).....	87
ERROR_ANTICPXSVR_DETECT_CALL	
BACK_IS_NOTIFIED.....	89
ERROR_ANTICPXSVR_HSHIELD_FILE	
ATTACK (0xE904000A).....	86
ERROR_ANTICPXSVR_INVALID_PARA	
METER (0xE9040003).....	81, 85
ERROR_ANTICPXSVR_MEMORY_ATT	
ACK (0xE904000C)	87
ERROR_ANTICPXSVR_NOT_YET_REC	
EIVED_RESPONSE (0xE9040005)	81, 86
ERROR_ANTICPXSVR_UNKNOWN_CL	
IENT (0xE904000E).....	87
ERROR_HSCRYPTLIB_EXCEPTION	153
ERROR_HSCRYPTLIB_FREAD_DECRY	
PT_FREAD	152
ERROR_HSCRYPTLIB_FREAD_DECRY	
PT_GETDECMMSG	152
ERROR_HSCRYPTLIB_FREAD_DECRY	
PT_RANGE.....	152
ERROR_HSCRYPTLIB_FREAD_FSEEK	
.....	152
ERROR_HSCRYPTLIB_FREAD_GETFIL	
ELEN.....	151
ERROR_HSCRYPTLIB_FREAD_GETPO	
SITION	152
ERROR_HSCRYPTLIB_FREAD_INVALI	
DPARAM.....	151
ERROR_HSCRYPTLIB_FREAD_SIZEZE	
RO.....	152
ERROR_HSCRYPTLIB_GETDECMMSG_I	
NVALIDPARAM	150
ERROR_HSCRYPTLIB_GETENCMMSG_I	
NVALIDPARAM	149
ERROR_SUCCESS	102, 105, 107, 115, 124, 130, 149, 150, 151

H

HS_ERR_ALREADY_GAME_STARTED	40, 41
HS_ERR_ALREADY_INITIALIZED.....	31
HS_ERR_COMPATIBILITY_MODE_RU	
NNING.....	30
HS_ERR_DEBUGGER_DETECT.....	31
HS_ERR_DRV_FILE_CREATE_FAILED	

.....	30, 40
HS_ERR_INIT_DRV_FAILED	31
HS_ERR_INVALID_FILES	31
HS_ERR_INVALID_LICENSE.....	30
HS_ERR_INVALID_PARAM	30, 45
HS_ERR_NEED_ADMIN_RIGHTS.....	32
HS_ERR_NOT_INITIALIZED	39, 42, 43, 44, 46
HS_ERR_OK.....	30, 39, 44, 46
HS_ERR_REG_DRV_FILE_FAILED ...	40
HS_ERR_START_DRV_FAILED.....	40
HS_ERR_START_ENGINE_FAILED...	39
HS_ERR_UNKNOWN	32
HS_ERR_VIRTUAL_MACHINE_DETEC	
T	41
HsCryptLib	143
HsCryptLib.lib	142
HsCryptoUtil 프로그램	142
HSERROR_ENVFILE_NOTREAD	60, 63
HSERROR_ENVFILE_NOTWRITE	60, 63
HSERROR_LIB_NOTEDIT_REG...	60, 64
HSERROR_NETWORK_CONNECT_FAI	
L	60, 64
HSERROR_NOTFINDFILE	60, 64
HShield.lib.....	49, 67, 96, 127, 134
HSUpdate.exe	13
HsUserUtil.....	156
HsUserUtil Data	154
HsUserUtil.lib	154
HSUSERUTIL_ERR_ADDNEWACE_FAI	
L	162
HSUSERUTIL_ERR_ADDNEWACETON	
EWDACL_FAIL	163
HSUSERUTIL_ERR_ADDNEWDACL_F	
AIL.....	162
HSUSERUTIL_ERR_ADDSHADOWACN	
T_FAIL	160
HSUSERUTIL_ERR_DELHIDEIDINFO_F	
AIL.....	160
HSUSERUTIL_ERR_DELSHADOWACN	
T_FAIL	160
HSUSERUTIL_ERR_DELSHADOWACN	
TINFO_FAIL.....	160
HSUSERUTIL_ERR_GETGROUPSID_F	
AIL.....	162
HSUSERUTIL_ERR_GETSECINFO_FAI	
L	162
HSUSERUTIL_ERR_NOT_ADMIN....	159
HSUSERUTIL_ERR_NOT_NT	160, 164, 165, 166
HSUSERUTIL_ERR_OK	159, 161, 163, 164, 165
HSUSERUTIL_ERR_SETFLDRPERMIS	
SION_FAIL.....	162, 164, 165, 166

P	V
PFN_AhnHS_Callback34	V3Pro32s.dll 13

10.3.변경 사항

수정 날짜	수정 사항
2008-02-26	- 기존 AhnLab_HackShield_프로그래밍_가이드.doc를 수정하여 AhnLab_HackShield_2.0_프로그래밍_가이드.doc 작성
2008-11-26	- [BT50597] AHNHS_ENGINE_DETECT_WINDOWED_HACK 콜백 추가
2009-01-07 김태훈	- 2008 -> 2009 - AntiFreeServer기능 추가(툴 설명 포함)