

[NT] Hidden Gamespy Code Leads to Vulnerabilities in Several Games

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-03/0006.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 03/01/04

To: list@securiteam.com

Date: 1 Mar 2004 19:05:21 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

Hidden Gamespy Code Leads to Vulnerabilities in Several Games

SUMMARY

Gamespy's CDKey validation toolkit is an SDK for games developers which enables them to easily implement online management of users and cd-key validation.

A flaw exists in the handling of query packets that can cause the game server to crash. Yet more "security through obscurity" flaws are described which give rise to the question of exactly how secure the user's information is.

DETAILS

Vulnerable Systems:

- * Battlefield 1942 versions 1.6.19 and 1.6rc1
- * Contract Jack version 1.1
- * Gore version 1.48 (1.49)
- * Haegemonia version 1.0.7
- * Halo version 1.031
- * Hidden & Dangerous 2 version 1.031
- * Project IGI 2 version 1.04

Securiteam: [NT] Hidden Gamespy Code Leads to Vulnerabilities in Several Games

- * Judge Dredd: Dredd vs. Death, version 1.01
- * Need For Speed Hot Pursuit 2, version 2.42
- * Terminator 3: War of the Machines version 1.0
- * TRON 2, version 1.042

The list of games is by no means complete. Other games are probably making use of the CDKey validation toolkit as well.

Remote Server Crash Bug

A bug exists in the code that handles a query packet. The code that copies the buffer that is delimited by backslashes does so in an unsafe manner. The following snippet of code demonstrates the programming error:

```
int size = strchr(buff + 1, '\\') - buff;
if(size > 32) return;
strncpy(querybuff, buff + 1, size);
```

It's plain to see two common programming errors. The first is not checking the return value of the `strchr()` function. In fact there is no test for a failure condition, such as a return value of 0. The second error is that `size` is in fact a signed integer. If `strchr()` fails and returns 0 which causes the `strncpy` function to reference negative offsets and therefore to cause an exception.

Privacy Issues

Some hidden commands can be used to query the server about specific user/s. For example, the 'ison' command is used to query whether a user with a specified CD key or MD5 hash is playing on the server. If the CD-key or hash of the user is known it becomes trivial to track the user across the gaming network.

Analysis of Hidden Code

There are some undocumented, hidden functions in the CDKey SDK. These functions are only activated when a packet with a special character is received by the server, or when the server needs to validate a user's CD-key.

The hidden function used to manage the "undocumented" queries is activated when a packet starting with the char ';' (byte 0x3B) reaches the query port of the game server of any user online. The query port is just the same UDP port used to receive the information queries "basic", "info", "status", "rules", etc.

In fact, the 0x3B is the backslash character (which is used to delimit a normal query) XORed with the char 'g' of the gamespy string. The packet sent to the server is simply XORed. The following code can be used to encode/decode the packet:

```
void gamespyxor(u_char *string, int len) {
    u_char gamespy[] = "gamespy",
        *gs;
```

Securiteam: [NT] Hidden Gamespy Code Leads to Vulnerabilities in Several Games

```
for(gs = gamespy; len; len--, gs++, string++) {
    if(!*gs) gs = gamespy;
    *string ^= *gs;
}
}
```

In order to hide the undocumented commands from prying eyes of users opening the executable using any hex editor, the code contains means for building the commands on the fly in order to evaluate them against the commands received from the packets.

```
:004422B7 B175 mov cl, 75
:004422B9 B06F mov al, 6F
:004422BB 56 push esi
:004422BC 8BF2 mov esi, edx
:004422BE B26E mov dl, 6E
:004422C0 884C240C mov byte[esp+0C], cl
:004422C4 884C2410 mov byte[esp+10], cl
:004422C8 884C2420 mov byte[esp+20], cl
:004422CC 884C2423 mov byte[esp+23], cl
:004422D0 33C9 xor ecx, ecx
:004422D2 85F6 test esi, esi
:004422D4 8844240D mov byte[esp+0D], al
:004422D8 88442412 mov byte[esp+12], al
:004422DC 8844241A mov byte[esp+1A], al
:004422E0 88442422 mov byte[esp+22], al
:004422E4 C644240E6B mov byte[esp+0E], 6B
:004422E9 C644240F00 mov byte[esp+0F], 00
:004422EE 88542411 mov byte[esp+11], dl
:004422F2 C64424136B mov byte[esp+13], 6B
:004422F7 C644241400 mov byte[esp+14], 00
:004422FC C644241869 mov byte[esp+18], 69
:00442301 C644241973 mov byte[esp+19], 73
:00442306 8854241B mov byte[esp+1B], dl
:0044230A C644241C00 mov byte[esp+1C], 00
:0044230F C644242163 mov byte[esp+21], 63
:00442314 88542424 mov byte[esp+24], dl
:00442318 C644242574 mov byte[esp+25], 74
:0044231D C644242600 mov byte[esp+26], 00
```

The portion of code comes directly from the file BF1942_w32ded.exe of Battlefield 1942 Win32 dedicated server 1.6.19 but this "hiding technique" is the same used in all the other vulnerable games and moreover also in all the Gamespy products (... "Gamers trust us" ...).

The generated commands are exactly: "uok", "unok", "ison" and "ucount". The first 2 commands in reality are replies sent by the Gamespy master server to the games servers when they request the validation of a cd-key using the "auth" query.

Examples of some practical queries which can be used:

Securiteam: [NT] Hidden Gamespy Code Leads to Vulnerabilities in Several Games

```
\uok\cd\0123456789abcdef0123456789abcdef\skey\1\errmsg\Valid CD Key
\unok\cd\0123456789abcdef0123456789abcdef\skey\1\errmsg\Invalid CD Key
\ison\skey\1\cd\0123456789abcdef0123456789abcdef
\ucount\
```

Where "skey" is an ID number used to track replies to a specific query and "cd" is the CD key hash. The CD key hash is simply the MD5 hash calculated by the client on his original CD key, it is sent by each client to the game server that uses it to validate the client through the master server.

After the server assembles the commands in memory and they are evaluated against the data from the packet, the server sends the reply. Examples of replies follow:

```
\uon\skey\1 the requested CD key is used in the target server
\uoff\skey\1 the requested CD key is not used in the target server
\ucount\9 in the target server there are 9 players using CD key
```

Proof of Concept Examples

Just sending the magic backslash character should be enough to crash the server. However, a small proof of concept code can be found at

<<http://aluidgi.altervista.org/poc/gshboom.zip>>
<http://aluidgi.altervista.org/poc/gshboom.zip>

A tool that can be used to send the undocumented commands to the server can be found at <<http://aluidgi.altervista.org/papers/gshinfo.zip>>

<http://aluidgi.altervista.org/papers/gshinfo.zip>

A tool that analyzes the packets can be obtained from

<<http://aluidgi.altervista.org/papers/gshsniff.zip>>
<http://aluidgi.altervista.org/papers/gshsniff.zip>

And a tool that logs the encoded UDP commands can be downloaded from

<<http://aluidgi.altervista.org/papers/gshlog.zip>>
<http://aluidgi.altervista.org/papers/gshlog.zip>

ADDITIONAL INFORMATION

The information has been provided by <<mailto:aluidgi@altervista.org>> Luigi Auriemma.

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:
list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====

=====

Securiteam: [NT] Hidden Gamespy Code Leads to Vulnerabilities in Several Games

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.