

[Full-disclosure] Multiple vulnerabilities in Toribash 2.71

Source: <http://www.derkeiler.com/Mailing-Lists/Full-Disclosure/2007-08/msg00352.html>

- *From:* Luigi Auriemma <aluigi@xxxxxxxxxxxxx>
 - *Date:* Sun, 19 Aug 2007 00:06:26 +0200
-

#####

Luigi Auriemma

Application: Toribash
<http://www.toribash.com>
Versions: <= 2.71
Platforms: Windows, Mac and Linux
Bugs: A) dedicated server format string
B) client commands buffer-overflow
C) client unicode buffer-overflow in the SAY command
D) server crash through uninitialized values
E) line-feed dropping
F) Windows dedicated server hell bell
G) clients kicked by malformed packet
Exploitation: A, D and F versus server
B locally versus clients
all the others remotely versus clients using servers as
"bridge" for the attacks (the attacker acts as a client)
Date: 17 Aug 2007
Author: Luigi Auriemma
e-mail: aluigi@xxxxxxxxxxxxx
web: aluigi.org

#####

- 1) Introduction
- 2) Bugs
- 3) The Code
- 4) Fix

#####

=====

1) Introduction

=====

Toribash is a turn-based multiplayer game in which two players fight using violent puppets.

The game servers naturally support spectators and there are some official and non-official leagues and championship for this game, other than some mods for emulating specific martial arts.

#####

=====

2) Bugs

=====

A) dedicated server format string

A format string vulnerability is exploitable when a client enters in the match, in this occasion a string containing "BOUT ID; 1 0 0 0 0 NICKNAME 0" is passed directly to `vfprintf()`, so the nickname of the client, limited to 32 chars, can be used by an attacker as format argument.

B) client commands buffer-overflow

A buffer-overflow is located in the client's function which reads the game commands.

The problem is caused by the calling of `sscanf()` with the format string "%s %i" and an output buffer of about 256 bytes.

This bug can be exploited in two different ways:

- locally using a malicious replay file (*.rpl)
- remotely through a malicious server controlled by the attacker

Replays are an essential component of the game since are very used for recording and watching the best matches.

The other way for exploiting the bug isn't so much realistic since doesn't exist a master server for making the own server public for anyone.

C) client unicode buffer-overflow in the SAY command

This problem is directly related to bug E.

As written there that bug forces the server to send commands without the final line-feed and so they are not processed by the client until the reception of this char.

An attacker can use this same bug for concatenating two or more commands (ever using the server as a "bridge"), in the case of the SAY command we will have that the server sends max 512 bytes of data for this command and an unicode buffer-overflow happens in the client if receives a SAY of over 1024 chars.

The only limitation is that the attacker (client) doesn't seem to be able to control the return address because it's overwritten by the subsequent command sent by the server:

```
SAY 0;nick: aaa...aaa??@SAY 0;nick: aaa...aaa??@COMMAND
first 512 bytes second 512 bytes subsequent command
```

The other possibility of exploiting this bug is naturally with the controlling of a server in which is possible to overwrite the return address with our unicode chars, but as already written in the previous bug it's not a realistic way.

D) server crash through uninitialized values

When a client joins a server an ID of -1 is assigned to it and no data is allocated until the ENTER command is called.

An attacker can join a server and send the GRIP command with the ID set to -1 for forcing the server to handle it (since the ID is correct) but the structure which will contain the values received by the client is NULL and so it will fall in the following situation:

```
sscanf("0 0\n", "%i %i", &client.integer1, &client.integer2);
```

where "0 0\n" is the second part of the GRIP command sent by the client ("GRIP -1;0 0\n") while client.integer1 points to 0x000030d0 and client.integer2 to 0x000030d4 since the structure which should contain them is a NULL pointer.

E) line-feed dropping

The protocol used by Toribash is composed by commands delimited by line-feed chars, like common telnet connections.

An attacker can block the clients which are playing in the server simply sending a chat message (or possibly other commands) which forces the server to send only a part of the incoming data to the other clients since, in the case of the SAY command, it automatically limits

the outgoing data to max 512 bytes forgetting to add the line-feed char needed by the client to handle the received command. The effect of this problem is that the clients will remain frozen until a line-feed is received.

F) Windows dedicated server hell bell

This type of Denial of Service could seem something like a joke but it works terribly well.

The problem of the dedicated server is that it shows tons of informations in the console and the clients can force the server to show how much chars they want using some specific commands. These chars are not filtered so an attacker could use many invalid chars (max 4096, line-feed included) like the bell 0x07 for freezing the Windows dedicated server through the bell heard in the console. The effects are just the slowness of the entire system, the complete freezing of the game server and the PC speaker yelling as a damned.

G) clients kicked by malformed packet

If an attacker joins the match (ENTER command) and sends a too long emote or SPEC command to a server, all the clients playing in it will be disconnected with the "malformed packet" message.

#####

=====
3) The Code
=====

<http://aluigi.org/poc/toribashish.zip>

#####

=====
4) Fix
=====

Vulnerability E and a variant of C were reported to the developers in October 2006. I understand that they are not code execution bugs and I considered them as low priority at that time but in all the versions

[Full-disclosure] Multiple vulnerabilities in Toribash 2.71

which have been released from that date they have never been fixed.

#####

Luigi Auriemma
<http://alugi.org>
<http://mirror.alugi.org>

Full-Disclosure – We believe in it.
Charter: <http://lists.grok.org.uk/full-disclosure-charter.html>
Hosted and sponsored by Secunia – <http://secunia.com/>