CVE-2021-33032: HomeMatic - Unauthenticated Remote Code Execution

2021-05-14 | Security | #homematic , #cve-2021-33032

Overview

- CVE: CVE-2021-33032
- Type: CWE-78: Improper Neutralization of Special Elements used in an OS Command (‘OS Command Injection’)
- CVSSv3.1 Base Score: 10.0
- Vendor: eQ-3 AG
- Product: HomeMatic CCU2/CCU3
- Affected version CCU2: 2.57.5 and below
- Affected version CCU3: 3.57.5 and below
- Patched versions: 2.59.7/3.59.6

Background

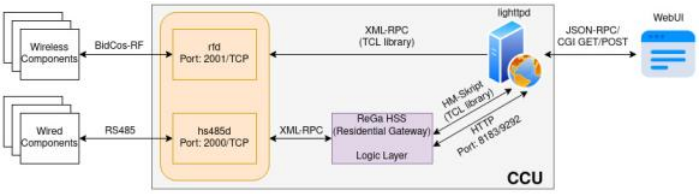
HomeMatic is a highly customizable home automation system consisting of more than 100 components that cover a wide range of home automation applications. All components can be connected to a central control unit (CCU) via a proprietary radio protocol that transmits in the ISM Type B band in the 868 MHz range.

From the vendor’s website (eQ-3):

> “
> As the central local component of the smart home system, the CCU3 ensures all communication between Homematic IP and/or Homematic devices. Thanks to numerous, individual configuration and control options via the proven “WebUI” user interface, the system has virtually no limits. …

Technical Background

It is important to have a basic understanding of the general architecture of the HomeMatic CCU.



rfd and hs485d

Actuator and sensor components are managed by the rfd and hs485d daemons depending on the connectivity of the component. These daemons are only used to abstract the different connectivity types and do not hold any state themselves except for a list of paired components.
For the sake of simplicity I will only talk about the rfd daemon from now on.

Each device (and channel) is uniquely identifiable by a device ID. Via this ID the status of a component can be changed directly via the XML-RPC interface of the rfd.

ReGa HSS

The ReGa service is a Java application that manages and holds the entire state of the system and can automatically react to state changes according to the user configuration.
ReGa subscribes to all state changes received by the rfd daemon via the XML-RPC interface.

Please note that ReGa *does not* provide an API for communication. The only way to interface with the ReGa application is via its own scripting language HM-Skript (German). ReGa provides a TCL library and a TCP interface on port 8181 to which scripts can be sent for execution.

ReGa also provides two web servers on port 8183/TCP and 9292/TCP. The web server running on port 8183 serves HTML files with embedded HM-Script. Port 9292 serves

It looks like ReGa originally had no user management capabilities. Until today there is no privilege context for HM-Script. Once an attacker gains access to an HM-Script interface, ReGa is fully exposed to the attacker.

HM-Script has an undocumented but well-known function called `system.Exec` that allows the execution of arbitrary shell commands.
Since ReGa runs with root privileges and `system.Exec` executes commands in the same context, an attacker immediately gains root access.

WebUI

WebUI is a web application that enables the management of the entire HomeMatic system.

The WebUI backend consists of TCL scripts being executed via the CGI. The backend can publish state changes directly to the rfd and also communicate with ReGa through a TCL library using ReGa's own scripting language HM-Script.

WebUI also consists of certain HTML resources with embedded HM-Script that are located at `/www/rega/`. Lighttpd proxies them directly to ReGa for evaluation and forwards the result back to the client.

In earlier versions of WebUI, user and session management was implemented in the TCL backend of WebUI. The session ID used to be sent in the 'icsessionid' query parameter.
Meanwhile, sessions are managed exlusively by ReGa using the 'sid' query parameter. However, in addition to HTML resources with embedded HM-Script, WebUI still uses TCL CGI scripts that also need to validate the session ID before executing HM-Script queries (remember that HM-Script itself has no privilege context).

/www/config/cp_software.cgi

```
cgi_eval {
...
  set action "put_page"

  catch { import action }

  if {[session_requestisvalid 8] > 0} then action_$action
}
```

Taking `/www/config/cp_software.cgi` as an example, this works as follows: When the script is called, it first checks if the current session is valid and the user has the required privilege level by calling the `session_requestisvalid` function.

/www/tcl/eq3_old/session.tcl

```
proc session_requestisvalid {needed_upl} {
    global sidname sid

    if {$sidname == "sid" && $sid != ""} then {
        return [session_requestisvalid_ise $needed_upl]
    }

    # Old TCL backend session management - not used anymore
...
}

proc session_getsessiondata_ise {} {
    global REGA_SESSION

    array set result [rega_script "string sd = system.GetSessionVarStr(\"$REGA_SESSION(
    set data $result(sd)
...
}

proc session_urlsid_short {} {
    # Extracts the session ID from the 'sid' query parameter
    set sid [session_urlsid]
    set mod_sid ""

    regexp {^@([a-zA-Z0-9]*)@$} $sid dummy mod_sid

    return $mod_sid
}

set REGA_SESSION(SID)       $sid
set REGA_SESSION(REGASID)   [session_urlsid_short]
```

`session_requestisvalid` originally validated sessions managed by the TCL backend. With the new ReGa session management, the script now calls the function `session_requestisvalid_ise` which in the end calls `session_getsessiondata_ise`.
The latter builds a HM-Script that requests the session information associated with the session ID from ReGa.
Since the session ID can only consist of lowercase and uppercase letters as well as numbers, no HM-Script injection is possible in this place.

Most of the code in the file `/www/tcl/eq3_old/session.tcl`, as well as in the whole directory `/www/tcl/eq3_old/` is part of the deprecated session management. As the `eq3_old` directory exists since at least 2015, it is not expected to be part of an ongoing refactoring.
Similar functions for validating sesssions against ReGa are also present in `/www/tcl/eq3/session.tcl` but not widely used.

Issue Description

While analyzing unauthenticated, public-facing components of the CCU's web interface (WebUI), a remote code exection vulnerability has been identified. This vulnerability can be exploited by unauthenticated attackers with access to the web interface.

The vulnerable component is located at `/www/tcl/eq3_old/verifysid.cgi` and exposed at `http://<CCUIP>/config/verifysid.cgi`.
When sessions were still managed by the TCL backend, this component was responsible for re-authenticating the user when a session had expired or the user's privilege level was not sufficient for the desired action.
As far as I can see, this functionality is no longer used. Starting with the switch to ReGa session management, the user is redirected to the login page managed by ReGa.

/www/tcl/eq3_old/verifysid.cgi

```
cgi_eval {
    set sid ""
    set tbUsername ""
    set tbPassword ""
    catch { import_as $sidname sid }
    catch { import tbUsername }
    catch { import tbPassword }

    if {$sidname == "sid"} then {
        set tbUid [user_uid_ise $tbUsername]
    } else {
        set tbUid [user_verify $tbUsername $tbPassword]
    }

    set s_uid [session_uid_sid $sid]

    if {$tbUid > 0 && ($s_uid == 0 || ($tbUid == $s_uid))} then {
        session_setuid_sid $tbUid $sid
        session_ResetTimeOut_sid $sid
        write_sessions

        session_redirecturl_sid redirect $sid
        redirect $redirect
    } else {
        session_putloginpage $sid "Anmeldung abgewiesen."
    }
}
```

`sidname` determines which session management is used. In `/www/tcl/eq3_old/session.tcl` `sidname` is set to 'sid' if the 'sid' query parameter is found, otherwise to 'icsessionid'.

In previous versions of WebUI, a new empty session was created when accessing the login page.

When using the WebUI session management, this script is responsible for validating the username and password and assigning the user to the current session if it is not already assigned to another user.
Should the session be already assigned to the requesting user, the session timeout is reset.
This functionality was used in early versions of WebUI to keep sessions alive and reauthenticate with higher privileges.

When using ReGa session management, the password is not checked. This would have allowed an attacker to associate an arbitrary user with an empty session without

I do not know why this functionality exists and also could not find any resource that uses this.

However, the critical part is the call to `user_uid_ise` with the submitted username.

/www/tcl/eq3_old/user.tcl

```
proc user_uid_ise {uname} {
    set    cmd ""
    append cmd "object oUSERS = dom.GetObject(ID_USERS);"
    append cmd "object obj = oUSERS.Get(\"$uname\");"
    append cmd "integer uid = -1;"
    append cmd "if (obj)"
    append cmd "{"
    append cmd "  uid = obj.ID();"
    append cmd "}"

    array set result [rega_script "$cmd"]

    return $result(uid)
}
```

The full username sent to the `verifysid.cgi` resource is embedded into a HM-Script. No validation or escaping is applied to the username.
This allows the unauthenticated execution of arbitrary HM-Script commands. With the undocumented `system.Exec` command a full takeover of the CCU is possible.

At the time of discovery more than 2,500 vulnerable systems were reachable on the internet (Source: Censys).

Proof-of-Concept

Create new system variable:

```
curl \
    -o /dev/null \
    -X POST \
    -d 'tbUsername=");var uid=-1;object olv=dom.GetObject(ID_SYSTEM_VARIABLES);object o
    'http://<HOMEMATIC_IP>/config/verifysid.cgi?sid=@@'
```

Create new file /tmp/pwned (root): [REDACTED]

Reverse shell (root): [REDACTED]

CVE

CVE-2021-33032

CVSSv3.1 Base Score

CVSS Base Score: 10.0
CVSS: AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H/E:H/RL:O/RC:C

Vulnerability Disclosure Timeline

- 2021/05/14: Discovery
- 2021/05/14: Vendor Notification
- 2021/05/17: Vendor Response (Screenshot of the user guide warning users of port forwardings. Case closed.)
- 2021/05/17: Response (Repeated request for contact details of a technical contact and notice that local installations are also at risk.)
- 2021/05/18: Vendor Response (Communication with customers exclusively via the support ticket system. Request for further details.)
- 2021/05/20: Response (Technical details with proof-of-concept and proposed solution.)
- 2021/05/20: Vendor Response (No short-term solution. Included in release planning.)
- 2021/06/29: CCU3 Patch Release (3.59.6 / HMCCU-810)
- 2021/07/06: CCU2 Patch Release (2.59.7 / HMCCU-810)
- 2021/07/21: Public Disclosure

Disclaimer

The information provided is released “as is” without warranty of any kind. The publisher disclaims all warranties, either express or implied, including all warranties of merchantability. No responsibility is taken for the correctness of this information. In no event shall the publisher be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if the publisher has been advised of the possibility of such damages.