Wi-Fi Training - Hands On

EMEA application

life.augmented

Hands On content

- In this presentation we provide a modular training on SPWF04 Wi-Fi module
  - It can also be used to find answers to common questions on the module
- A number of laboratories have been conceived to facilitate the user to get familiar with SPWF04 module key features
  - Some HW and SW is needed to perform the training
- An evaluation environment is offered by ST:
  - The Nucleo X-Pansion i.e. X-NUCLEO-IDW04A1
    - For the evaluation of SPWF features plus other ST components of the Nucleo ecosystem

Lab Prerequisites
X-NUCLEO-IDW04A1

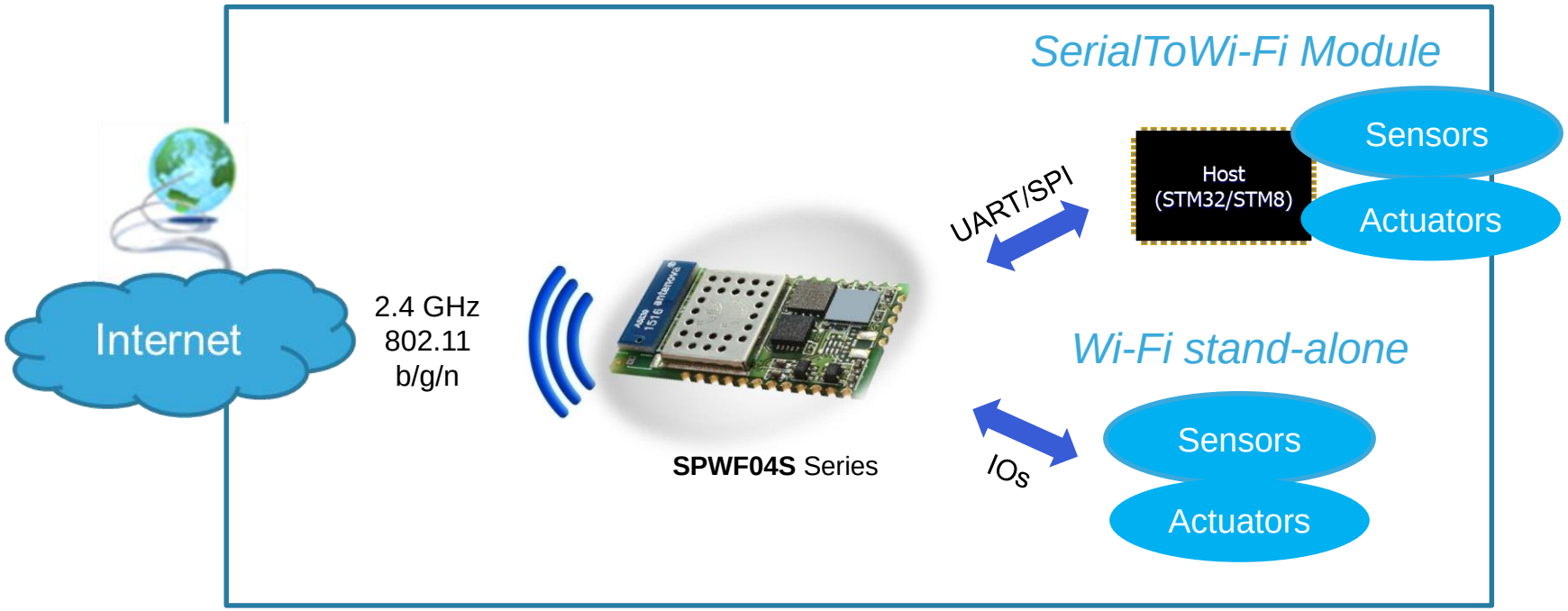
- Hardware
  - X-NUCLEO-IDW04A1, Wi-Fi expansion board based on SPWF04 module for STM32 Nucleo-64
  - NUCLEO-F401RE, NUCLEO-F411RE or NUCLEO-L476RG
- Utility software
  - STSW-WIFI004 FW package
    - www.st.com/stsw-wifi004
  - X-CUBE-WIFI1 SW package
    - www.st.com/x-cube-wifi1
  - Tera Term: terminal emulator
    - http://en.sourceforge.jp/projects/ttssh2/releases
  - Notepad++: text editor
    - https://notepad-plus-plus.org/download
  - Google Chrome: web browser
    - https://www.google.it/chrome

Hands on chapters

- Lab 0 : Module presentation
- Lab 1 : HW and SW setup
  - Lab 1.1 : UART Configuration
  - Lab 1.2 : SPI Configuration
  - Lab 1.3 : MicroPython Configuration
- Lab 2 : Set the SPWF variables
- Lab 3 : FW Upgrade
  - Lab 3.1 : Through UART
  - Lab 3.2 : Through SWD
  - Lab 3.3 : FOTA
  - Lab 3.4 : SFOTA
- Lab 4 : Used modes
  - Lab 4.1 : Mini AP
  - Lab 4.2 : Station
    - Lab 4.2.1 : Through UART
    - Lab 4.2.2 : Through WPS
    - Lab 4.2.3 : Through First Set Page
  - Lab 4.3 : Station/Mini AP Switcher
  - Lab 4.4 : IBSS mode
- Lab 5 : mDNS
- Lab 6 : Socket interface
  - Lab 6.1 : Socket Client
  - Lab 6.2 : Socket Server
  - Lab 6,3 : Broadcast
- Lab 7 : Websocket
- Lab 8 : HTTP Web Interface
  - Lab 8.1 : Web Client Mode
  - Lab 8.2 : Web Server Mode
    - Lab 8.2.1 : Web Server Feature
      - Lab 8.2.1.1 : List, Print a File
      - Lab 8.2.1.2 : Create, Append and Delete a File in RAM
      - Lab 8.2.1.3 : Create a File in Flash
        - Lab 8.2.1.3.1 : FS upgrade OTA
        - Lab 8.2.1.3.2 : FS upgrade over UART
        - Lab 8.2.1.3.3 : FS upgrade through SWD
      - Lab 8.2.1.4 : Create Dynamic Page
        - Lab 8.2.1.4.1 : Input Demo
        - Lab 8.2.1.4.2 : Output Demo
        - Lab 8.2.1.4.3 : Remote control of GPIO
    - Lab 8.2.2 : Web Server Usage
- Lab 9 : SMTP
- Lab 10 : MQTT
- Lab 11 : TFTP
  - Lab 11.1 : TFTP Client
  - Lab 11.2 : TFTP Server
- Lab 12 : ADC
- Lab 13 : Low Power Modes

Lab 0 : Module presentation

The SPWF04S is a “Cloud Compatible” Wi-Fi Module working either in Stand-alone or Serial-to-WiFi mode. These modules integrate free of charge FW supporting Security (TLS, WPS, WEP, WPA2 and WPA-Enterprise) and a robust IP Stack with HTTPS, MQTT, SMTP, WebSockets, IPv6 protocols



Integrated ST Technology

- Low-power CW1100 802.11 b/g/n certified SoC
- Cortex-M4-based STM32 microcontroller

Lab 0 : Module presentation

Integration

- 2.4 GHz IEEE **802.11 b/g/n** low power transceiver
- **STM32** ARM Cortex-M4 microcontroller
- 2MB **Integrated Flash** memory
- Integrated highly efficient antenna or U.FL connector
- Certified RF (**FCC, IC, CE / RED**)
- **Integrated TCP/IP** and Application Layer Functions

Flexibility and ease of use

- Easy interface to host Microcontroller through **UART/SPI** or standalone supported with **MicroPython script language**
- **TLS for End-to-End** security integrated in all modules
- Security: **WPS**, **WEP, WPA2**, **WPA-Enterprise**
- **HTTPS, MQTT, SMTP, WebSockets, IPv6** protocols and to easily connect applications to the cloud
- **Over The Air** firmware update
- System Modes: **mini-AP mode, IBSS** and **Station**

RF power

- Up to **+18.3 dBm** output power

Size and temperature

- **Small** form factor: 26.92 x 15.24 x 2.35 mm
- **Industrial** temperature range: -40 °C to +85 °C

SPWF04SA
Integrated antenna

SPWF04SC
Integrated U.FL connector

www.st.com/wifimodules

life.augmented

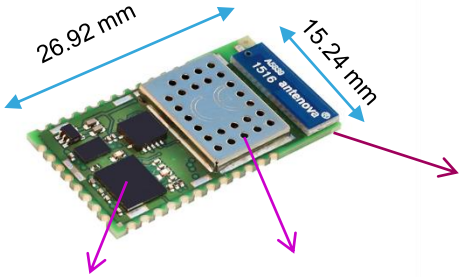
Lab 0 : Module presentation – Feature set

SPWF04 Features		February 2017 - FW1.0.0			November 2017 - FW 1.1.0		
		UART	SPI	µPython	UART	SPI	µPython
Wi-Fi modes	Station	x	x	x			
	miniAP	x	x	x			
	IBSS	x	x	x			
Pairing & Security	WPS	x	x	x	x	x	x
	WPA2	Station only	Station only	Station only	x	x	x
	WPA-E	x	x	x	x	x	x
	TLS Client	x	x	x	x	x	x
	TLS Server	x	x	x	x	x	x
Services	TCP/UDP	x	x	x	x	x	x
	IPv4 + IPv6	x	x	x	x	x	x
	WebSocket Client	x	x		x	x	x
	MQTT	x	x		x	x	x
	SMTP	x	x		x	x	x
	TFTP Server	x	x		x	x	x
	TFTP Client	x	x		x	x	x
	HTTP Server	x	x		x	x	x
	HTTP Client	x	x		x	x	x
Throughput		1Mbps	3Mbps		1Mbps	5Mbps	
MicroPython		REPL & Script		Script	REPL & Script		Script

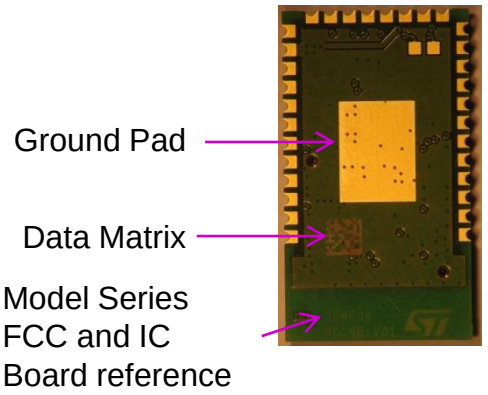
ECC supported

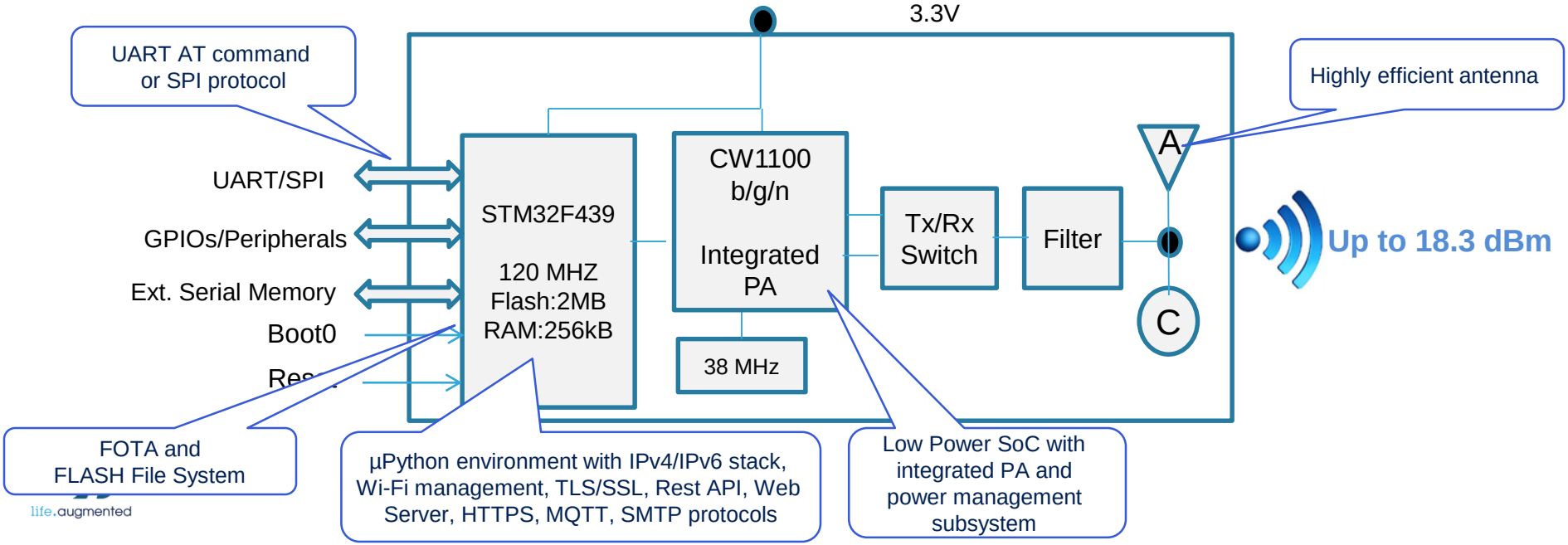
Lab 0 : Module presentation - HW

Front

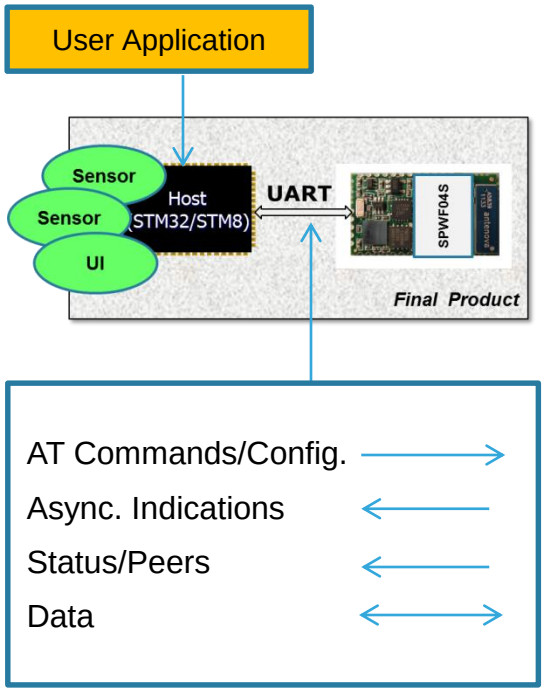


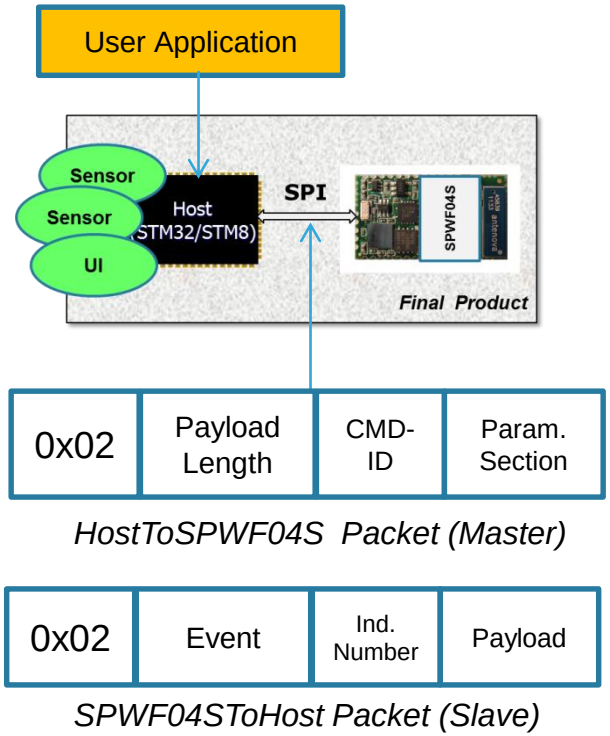
Bottom

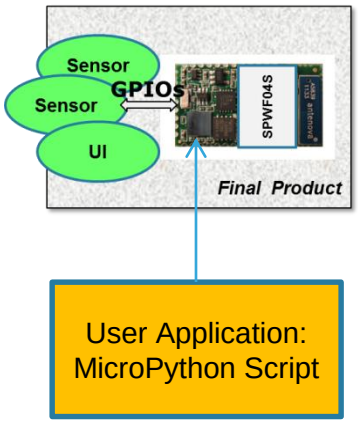




Lab 0 : Interface Modes







Plug&Play!

Host via UART

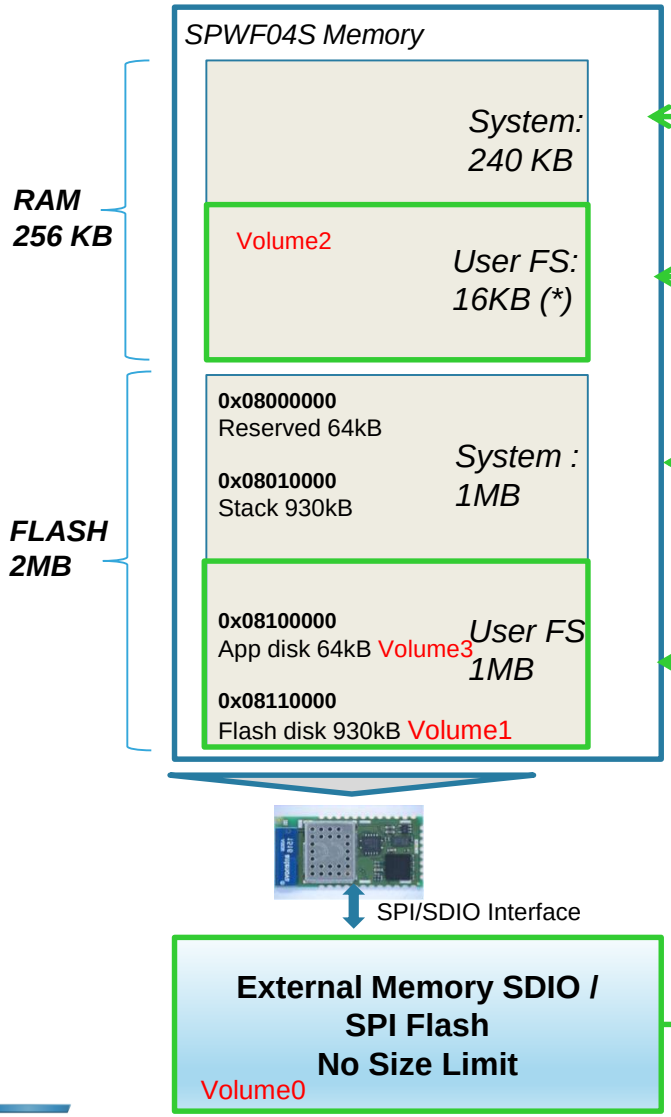
Performance!

Host via SPI

Cost Saving!

In-Module applications with MicroPython scripting

Lab 0 : File System Organization



Built-in Files

System RAM: leveraged also by the built-in Web Server to handle complex pages

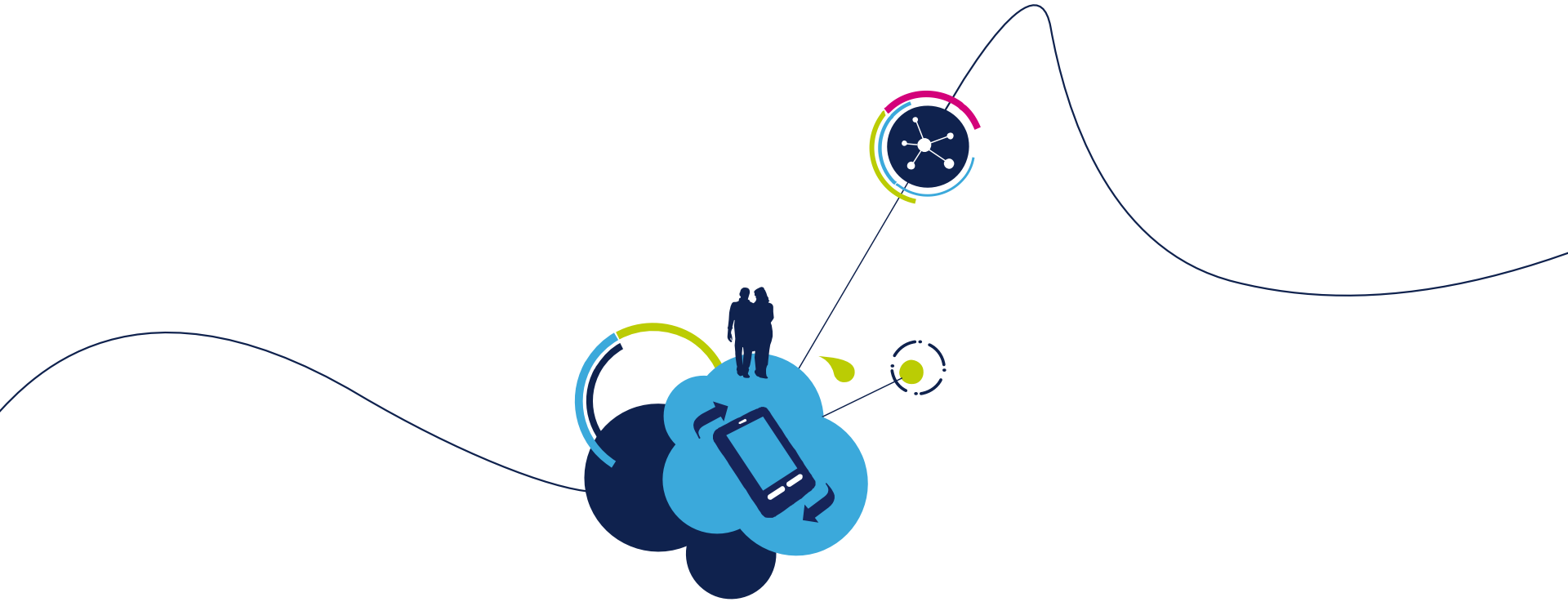
RAM . *Up to 32 files stored in the RAM can be generated and handled at run time (Tag D).*

Reserved 64KB organized in 4 files: Bootloader, TLS Certificates, Enterprise Certificates. ST reserved.

User Flash *1 MB of Flash is available to store proprietary files organized as:*
- App Disk (Tag I) e.g. uP scripts and
- Flash Disk (Tag E) for: pdf, webpages, docs

External Memory Volume. *(Tag X): Dedicated GPIOs are available to extend the storage capabilities with an SPI Flash or an SDIO (a variable is defined to select the option)*

(*) FW1.1.0 allows to define the amount of RAM for Volume number 2. Zero to turn it off.



Proceed to the next LAB!

Lab 1 : HW and SW setup

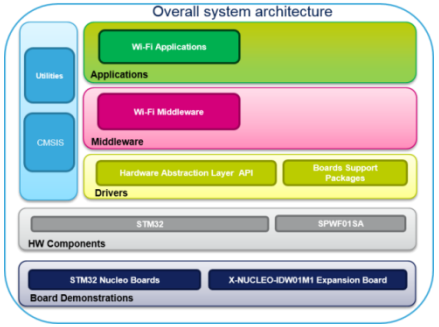
- Objective
  - Hardware set-up
  - Software set-up

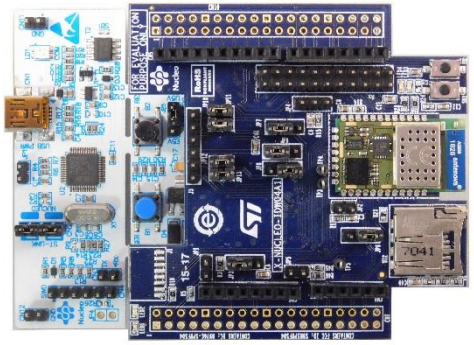
- Prerequisites
  - Work alone

Lab 1 : EVAL of the SPWF04 module X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

- X-NUCLEO-IDW04A1
  - X-NUCLEO-IDW04A1 is a Wi-Fi evaluation board based on SPWF04 module

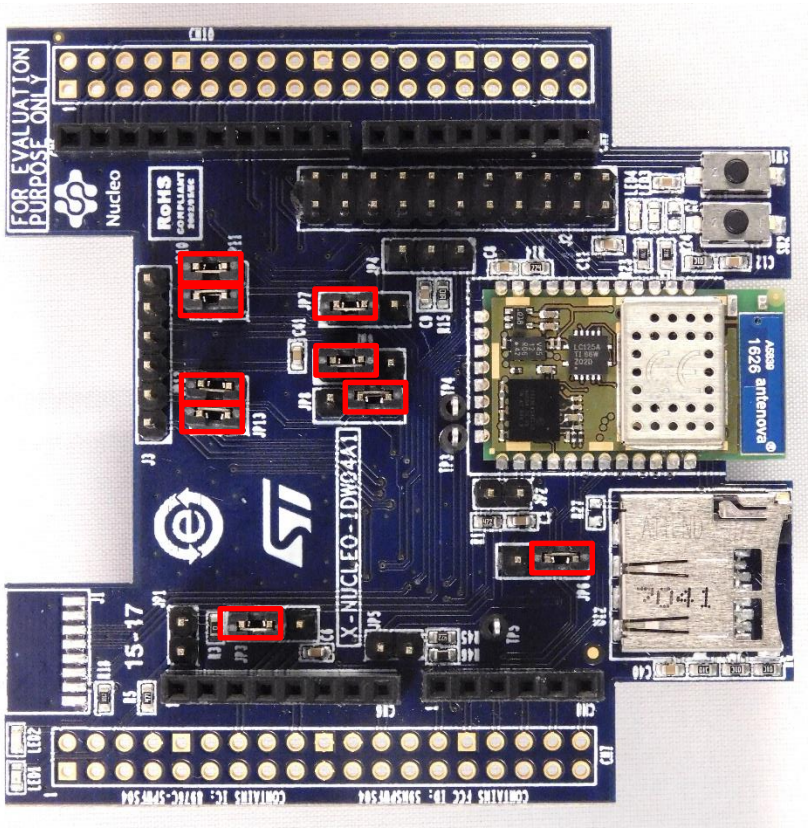
- X-CUBE-WIFI1
  - X-CUBE-WIFI1 SW package Link to X-CUBE-WIFI1





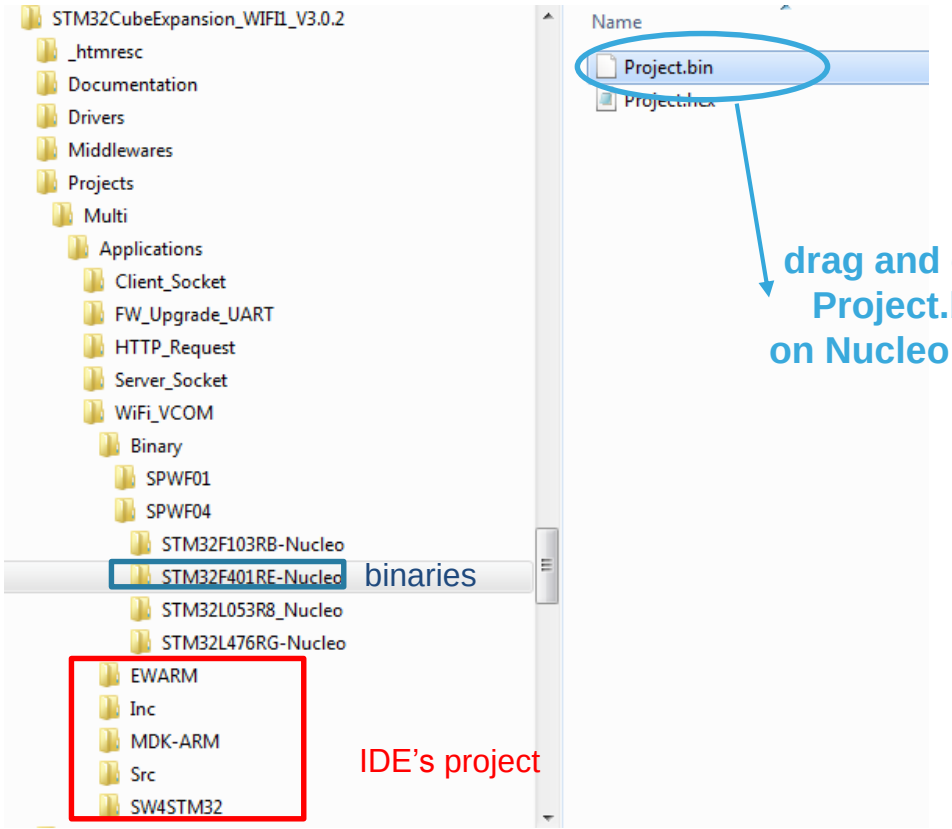
Lab 1.1 : UART Configuration
Set Vcom binary in Nucleo X-NUCLEO-IDW04A1 & X-CUBE-WIFI

1 Put the Jumpers in the right position



Lab 1.1 : UART Configuration
Set Vcom binary in Nucleo
X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

1 Flash UART VCOM binary into Nucleo drive



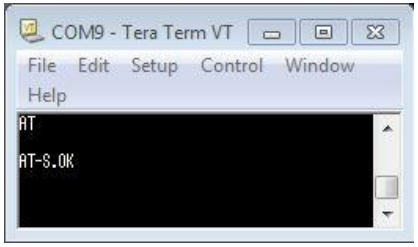
NUCLEO-F401RE,
F411,
L476RG

Lab 1.1 : Configuring the UART X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

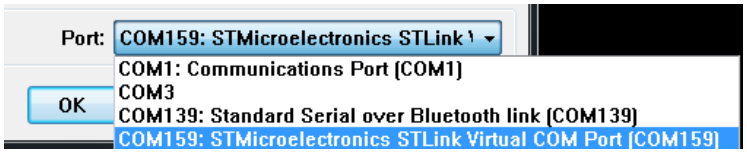
2 Setup TeraTerm window in order to send AT command to Wi-Fi module

Open Hyper terminal or TeraTerm

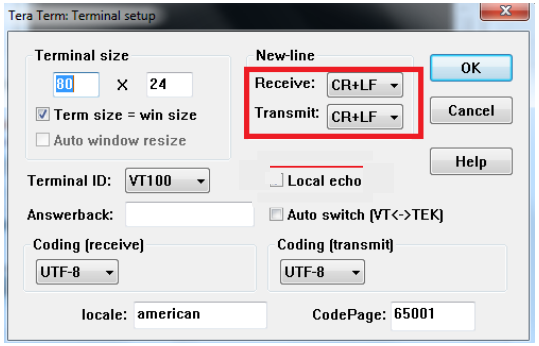
HyperTerminal or similar



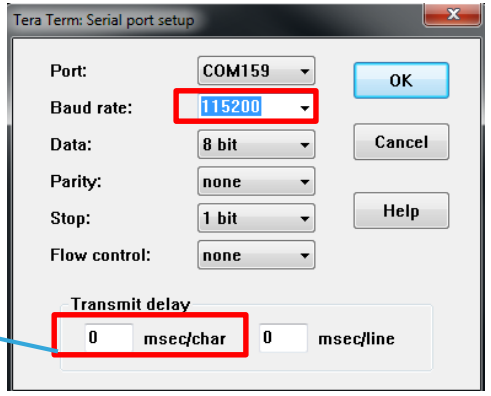
Select right COM port



Terminal setup



Serial port setup



1ms delay not needed anymore starting from X-CUBE-WIFI v3.02

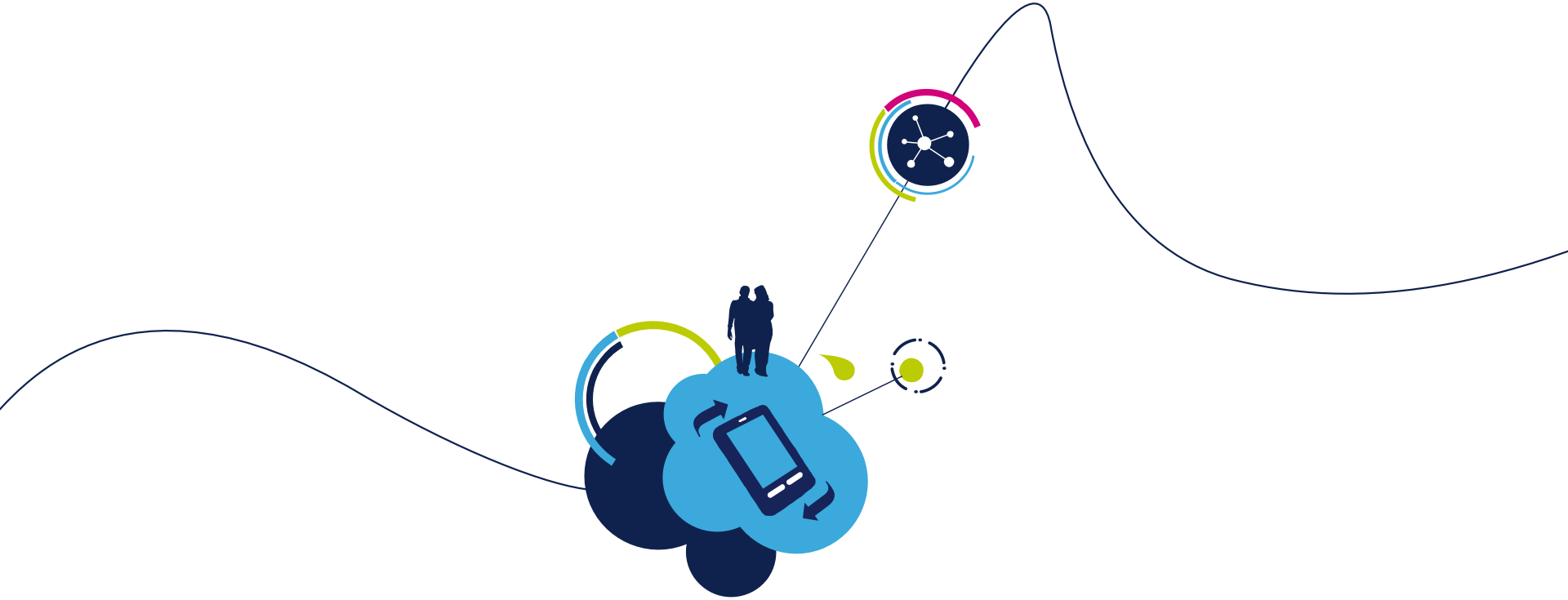
life.augmented

Lab 1.1 : Configuring the UART X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

- Open Tera Term
- Command Mode
  - Type **AT** followed by a carriage return (CR)

Tera Term output

AT
AT-S.OK

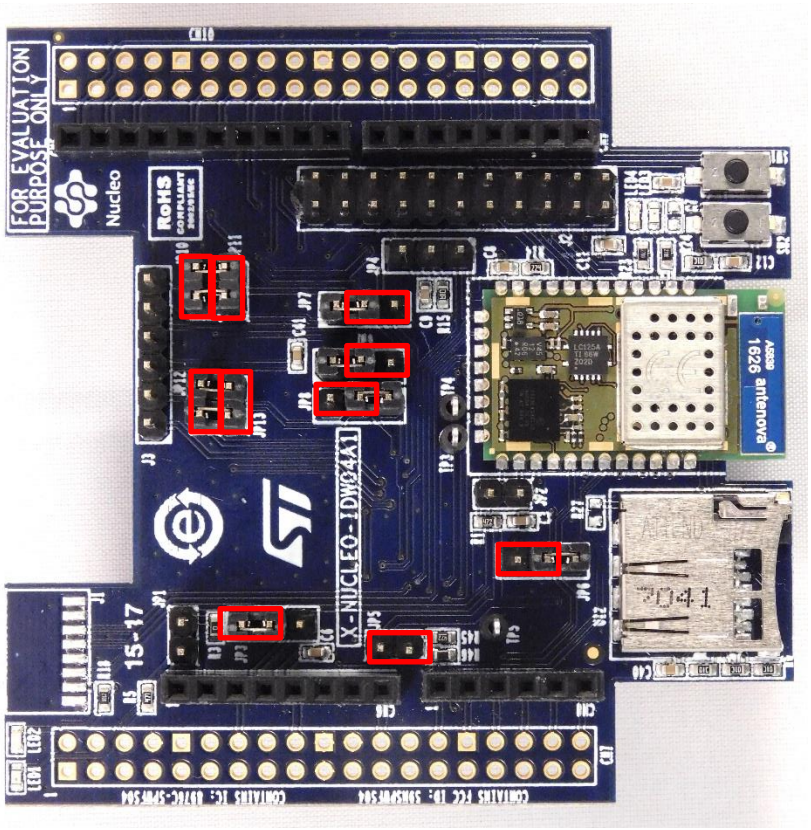


You are ready to use your Wi-Fi EVAL board!

life.augmented

Lab 1.2 : For SPI Configuration
Set Vcom binary in Nucleo
X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

1 Put the Jumpers in the right position



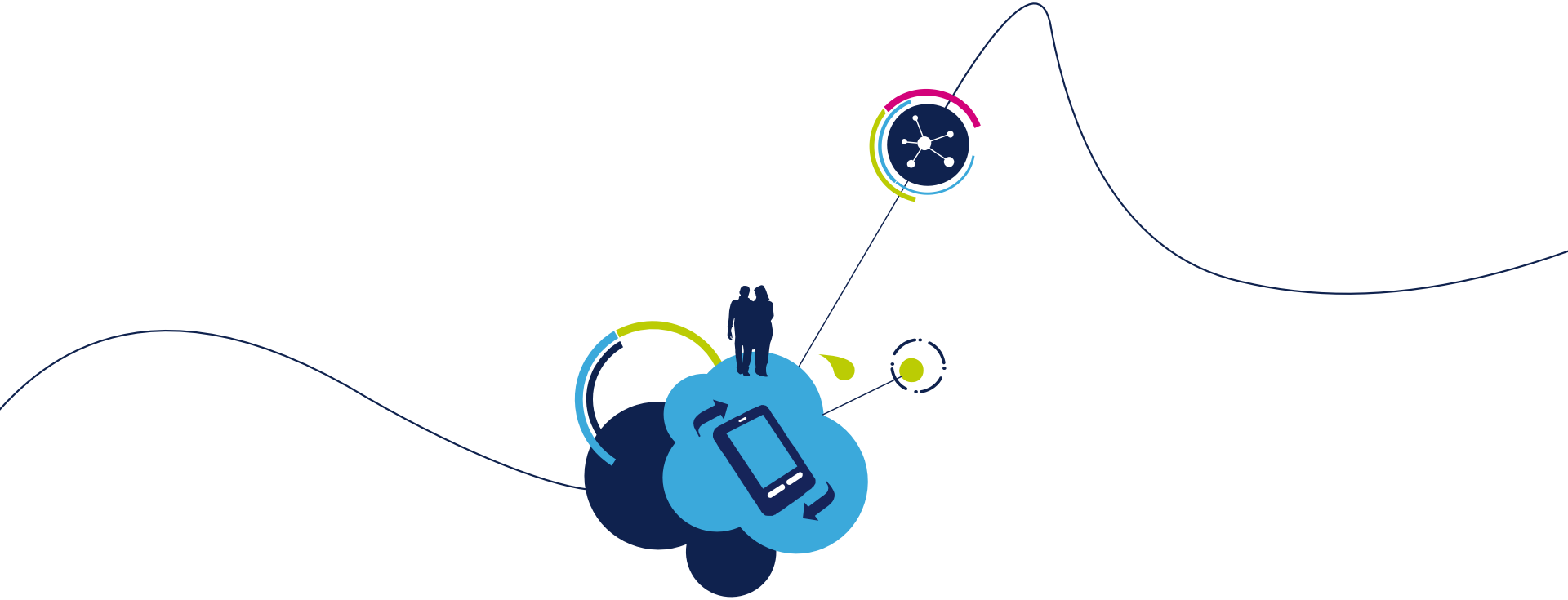
Lab 1.2 : Running some project in SPI

1. Some projects are available using SPI in X-CUBE-WIFI1

2. For example socket client project using STM32F401RE
   - In wifi_conf.h select SPI (comment line as per below)

```
#ifdef USE_STM32F4XX_NUCLEO
 //#define CONSOLE_UART_ENABLED
 //#define SPWF01
 #define SPWF04
 #endif
```

   - Compile and run available project



You are ready to use your Wi-Fi EVAL board!

life.augmented

Lab 1.3 : For MicroPython Configuration X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

1. Dedicated app note is available on st.com
AN4964 : MicroPython scripting language over SPWF04S

2. Below is very basic micro python example

```
from pyb import LED
import utime
l = LED(2)
l.on()
cnt=0
while True:
 l.toggle()
 cnt=cnt+1
 utime.sleep(1)
 print('Loop ', str(cnt))
 if cnt == 100:
   break
```

Lab 1.3 : For MicroPython Configuration X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

3 As stated in AN4964, below parameter must be set in order to enable micropython through console

AT+S.SCFG=console_enabled,2
AT+S.WCFG
AT+S.RESET

```
AT+S.SCFG=console_enabled,2
AT-S.OK

AT+S.WCFG
AT-S.OK

AT+S.RESET
+WIND:2:Reset
+WIND:1:Poweron:170216-fd39c59-SPWF04S
```

4 Here we will simply create & load this script in RAM (see chapter 8.2.1.2)

AT+S.FSC=blink_led.py,173

```
AT+S.FSC=blink_led.py,173
AT-S.OK
```

5 Confirm file is now created in RAM

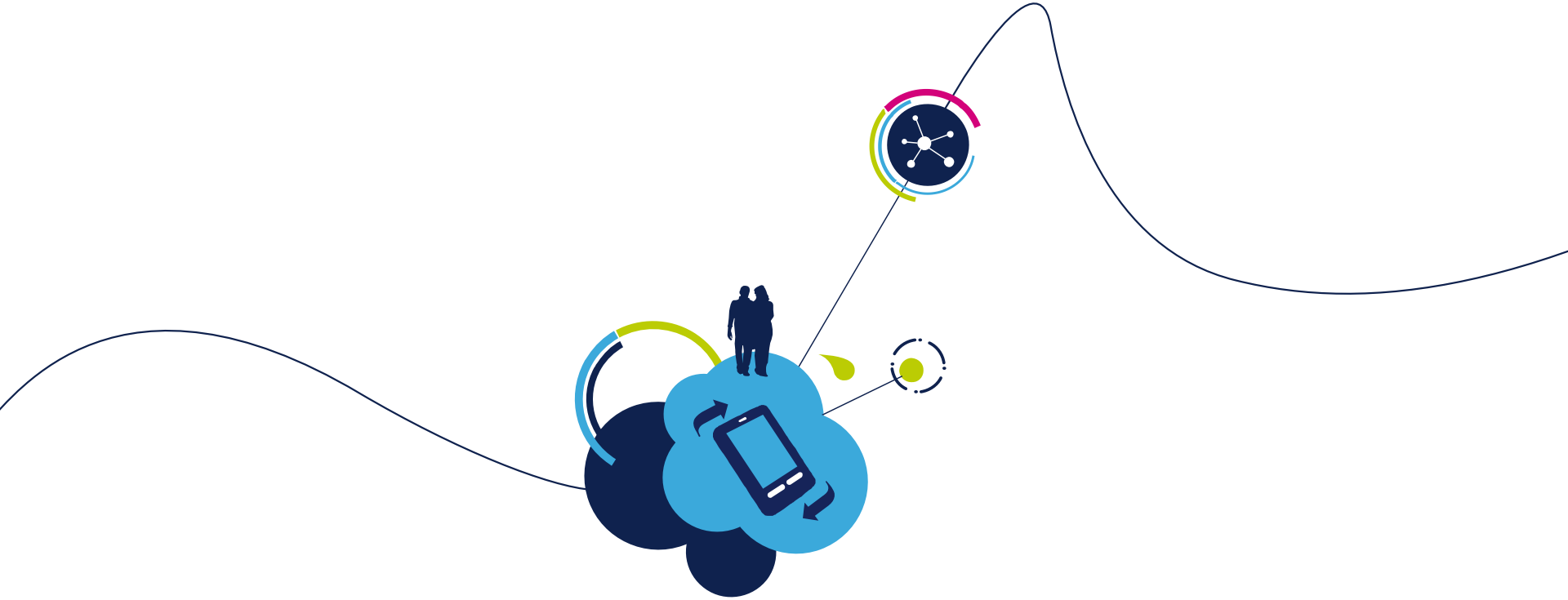
AT+S.FSL

```
at+s.fsl
AT-S.Free RAM Disk:14848
AT-S.File:D     173     blink_led.py
AT-S.File:I     4241    config.fhtml
AT-S.File:I     676     favicon.gz.ico
AT-S.File:I     697     firstset.gz.html
AT-S.File:I     401     index.gz.html
AT-S.File:I     252     input_demo.fhtml
AT-S.File:I     658     MULTI_CLIENT_SERVER.py
AT-S.File:I     290     output_demo.gz.html
```

Lab 1.3 : For MicroPython Configuration X-NUCLEO-IDW04A1 & X-CUBE-WIFI1

6 Execute micro python script

```
AT+S.PYTHON=blink_led.py
AT-S.Launching script:2:blink_led.py
AT-S.OK
Loop  1
Loop  2
Loop  3
Loop  4
Loop  5
Loop  6
Loop  7
Loop  8
Loop  9
Loop  10
```



You are ready to use your Wi-Fi EVAL board!

life.augmented

Lab 2 : Set the SPWF variables

- Objective
  - Run a command
  - Get the default configuration dump
  - Set host name
  - Set static IP parameters
  - Reset the module
- Prerequisites
  - Work alone

Lab 2 : Run a command

Run a command - Syntax

- **AT+S.[Command]<CR>**

AT Command Prefix

Not case sensitive

Utilities	
AT	Attention
AT+S.HELP	Display Help Text
AT+S.FWUPDATE	Perform a firmware update
AT+S.WIFI	Enable/Disable WiFi device
AT+S.RESET	Comm Function (Reset)

Configuration	
AT+S.GCFG	Get configuration value
AT+S.SCFG	Set configuration value
AT+S.SSIDTXT	Set a textual SSID
AT+S.STS	Display all configuration values
AT+S.FCFG	Restore factory default settings
AT+S.WCFG	Save current settings

Network	
AT+S.PING	Send a ping to a specified host
AT+S.SCAN	Channels Scan
AT+S.HTTPGET	Issue an HTTP GET

File Management	
AT+S.FSC	Create a file
AT+S.FSA	Append to an existing file
AT+S.FSD	Delete an existing file
AT+S.FSL	List existing filename(s)
AT+S.FSP	Print the contents of an existing file
AT+S.HTTPDFSUPDATE	Update static HTTPD filesystem

GPIO	
AT+S.GPIOC	Configure General Purpose I/O
AT+S.GPIOR	Query General Purpose Input
AT+S.GPIOW	Set General Purpose Output

Response – Syntax

- Optional «AT-S.Output» to monitor command execution, followed by
- «AT-S.OK»

- «AT-S.ERROR:Number:Reason»

Both monitoring and error verbosity level and can be set by proper configuration variables

Lab 2 : Get the SPWF variables

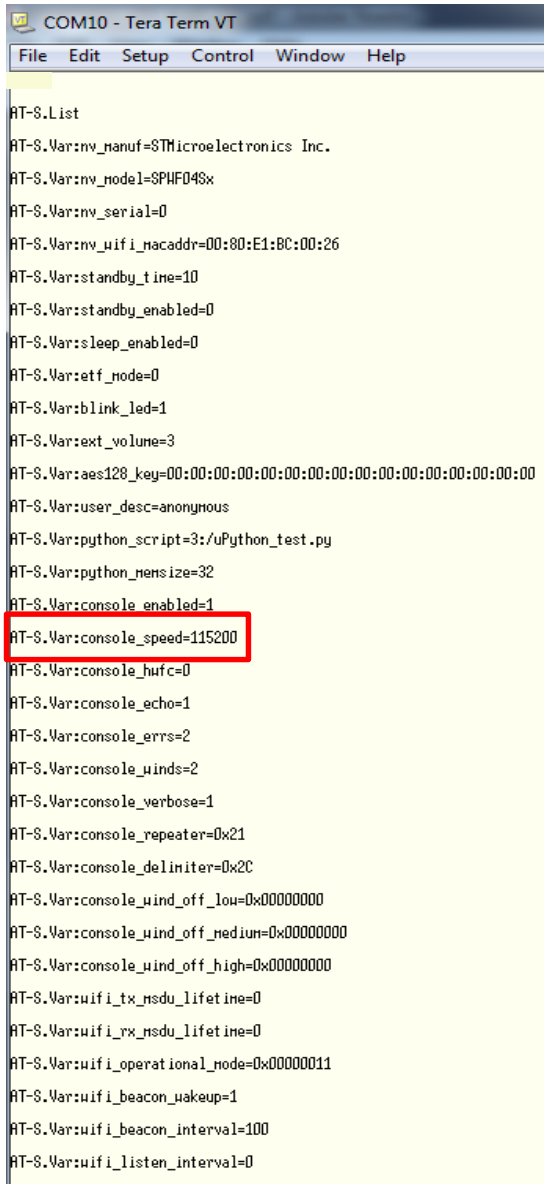
- Get the default configuration dump
  - Type **AT+S.GCFG**
- Get a variable value
  - Type **AT+S.GCFG=console_speed**

Config dump frame

Variable Name	Variable Value

How to change the variable value?

1. AT+S.SCFG=[var name],xxxx
2. AT+S.WCFG



Lab 2 : Set the SPWF variables

- Get the default configuration dump
- Set host name

When you're going to change the radio settings, we advise to you to turn off the wifi during the whole configuration and to turn on again at the end.

- Type **AT+S.SCFG=ip_hostname,xxxxxxxx**

Up to 31 characters (case sensitive), "spacebar" is allowed

Tera Term output

AT+S.SCFG=ip_hostname,xxxxxxxx
AT-S.OK

Lab 2 : Set the SPWF variables

- Get the default configuration dump
- Set host name
- Set IP address, IP default gateway, IP DNS and IP netmask (for static usage)
  - Type **AT+S.SCFG=ip_ipaddr,192.168.0.1**
  - Type **AT+S.SCFG=ip_gw,192.168.0.1**
  - Type **AT+S.SCFG=ip_dns1,192.168.0.1**
  - Type **AT+S.SCFG=ip_netmask,255.255.255.0**

Tera Term output

```
AT+S.SCFG=ip_ipaddr,192.168.0.1
AT-S.OK

AT+S.SCFG=ip_gw,192.168.0.1
AT-S.OK

AT+S.SCFG=ip_dns1,192.168.0.1
AT-S.OK

AT+S.SCFG=ip_netmask,255.255.255.0
AT-S.OK
```

Lab 2 : Set the SPWF variables

- Get the default configuration dump
- Set host name
- Set IP address, IP default gateway, IP DNS and IP netmask
- Save settings on the flash memory
  - Type **AT+S.WCFG**
- Reset the module
  - Type **AT+S.RESET**

Tera Term output

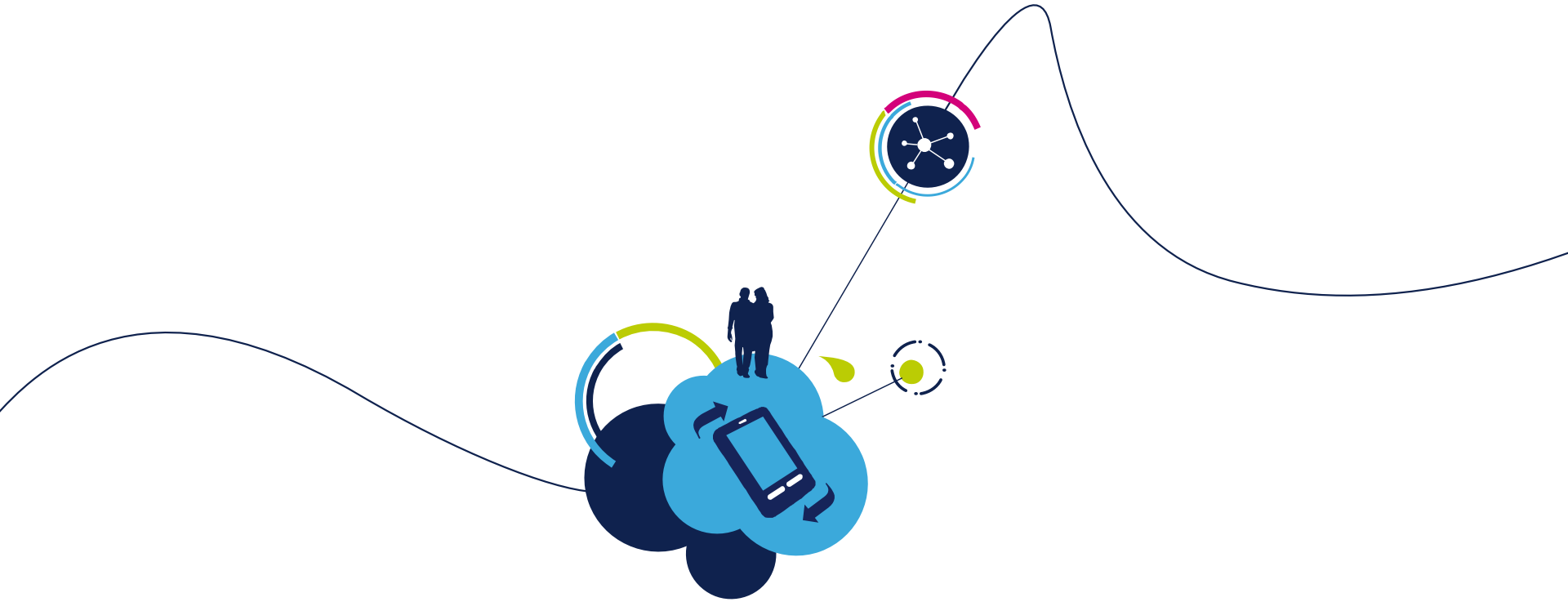
AT+S.WCFG
AT-S.OK

AT+S.RESET
+WIND:2:Reset

Lab 2 : Set the SPWF variables

- Get the default configuration dump
- Set host name
- Set IP address, IP default gateway, IP DNS and IP netmask
- Save settings on the flash memory (mandatory after a variable change) and reset the module
- Check the new configuration dump
  - Type **AT+S.GCFG**

```
AT-S.Var:ip_ipaddr=192.168.0.1
AT-S.Var:ip_netmask=255.255.255.0
AT-S.Var:ip_gw=192.168.0.1
AT-S.Var:ip_dns1=192.168.0.1
```



Proceed to the next LAB!

Lab 3 : Firmware Upgrade

- Objective
  - Upgrade the Firmware using 4 different methods
- Prerequisites
  - Getting latest Firmware from ST

Lab 3 : Firmware Upgrade

- FW Upgrade : why ?
  - As for any supplier providing chipset with integrated FW, user must ensure possibility to perform FW upgrade.
  - Each new FW version is developed with objective of bringing maturity or new features while keeping backward compatibility with previous FW.

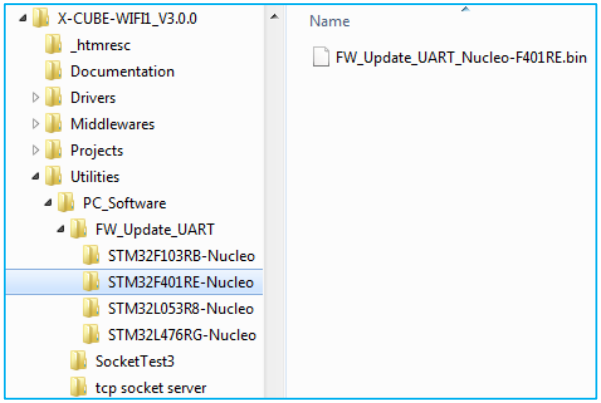
FW upgrade is a must to ensure customer homogeneous production.

Lab 3.1 : Upgrade through UART

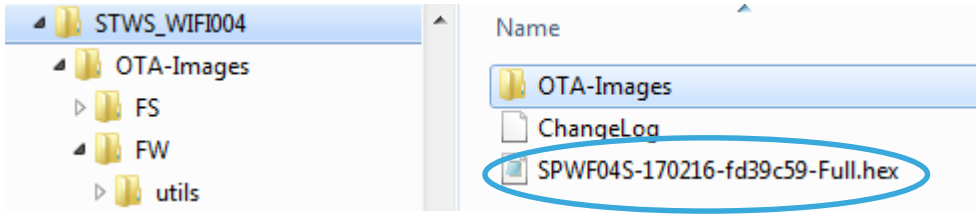
- Objective
  - Upgrade the FW using UART
- Prerequisites
  - HEX file (provided in the STSW-WIFI004 FW package)
    - http://www.st.com/content/st_com/en/products/embedded-software/wireless-connectivity-software/stsw-wifi004.html

Lab 3.1 : Upgrade through UART

- A dedicated FW must be set Nucleo board
- Flash Nucleo with binary available in SW package X-CUBE-WIFI1.

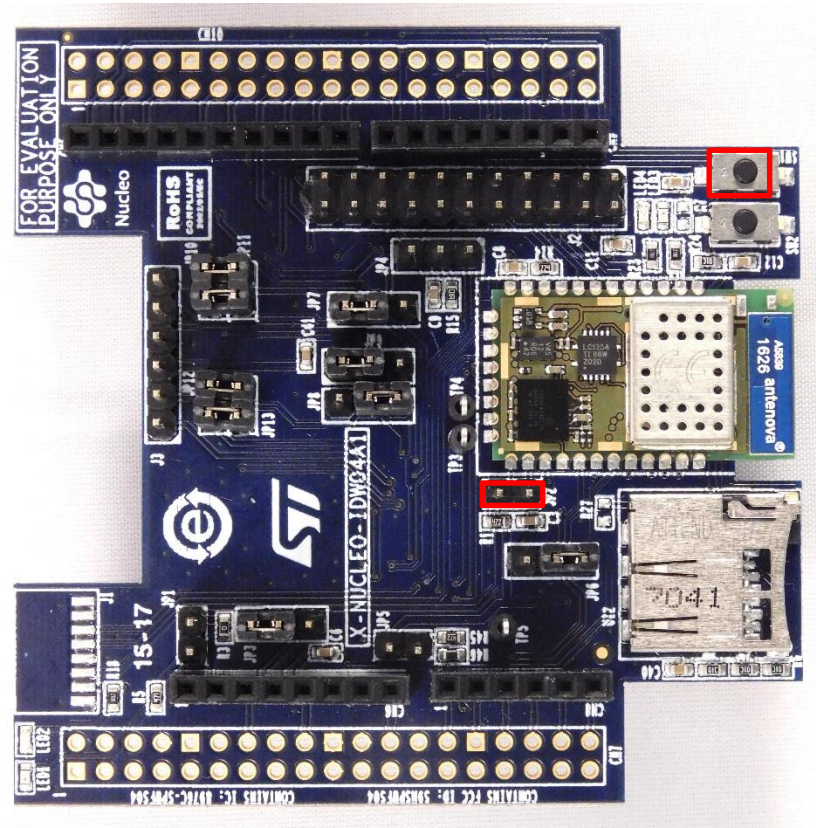


- Get hex file from latest STWS_WIFI004 package.



Lab 3.1 : Upgrade through UART

- Preparing the X-NUCLEO-IDW04A1 Board

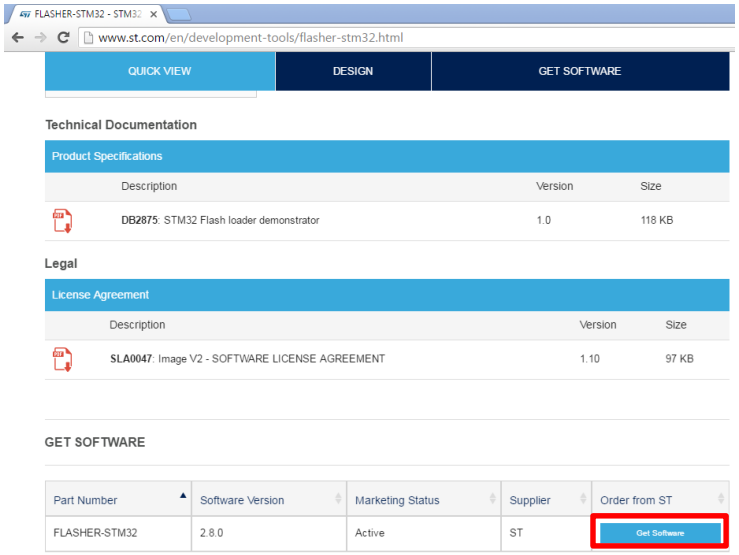


- Set a jumper on JP2 as shown on picture. This will pull the BOOT0 pin high (force bootmode on Wi-Fi module)
- Connect X-NUCLEO-IDW04A1 and NUCLEO and connect Nucleo to PC through USB
- RESET both boards (press SW1 on X-NUCLEO-IDW04A1 and the B2 on the NUCLEO)

Lab 3.1 : Upgrade through UART

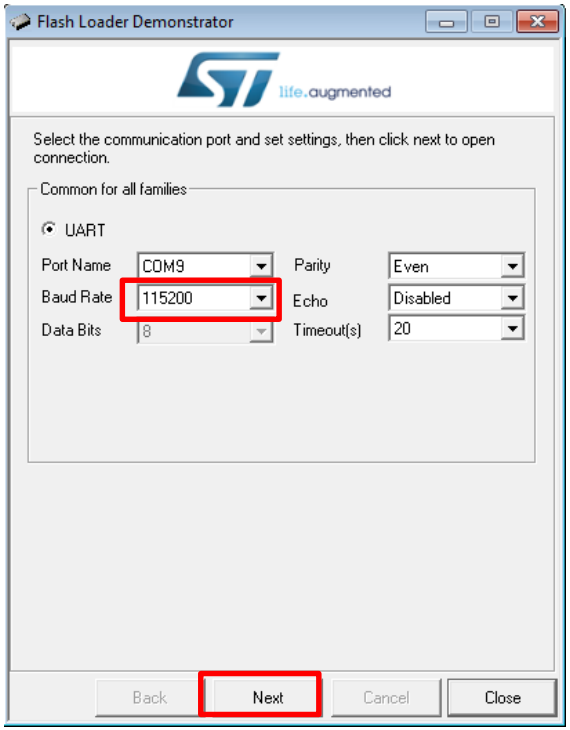
- Downloading the Flash Loader tool

- www.st.com/web/catalog/tools/FM147/CL1794/SC961/SS1743/PF257525



Lab 3.1 : Upgrade through UART

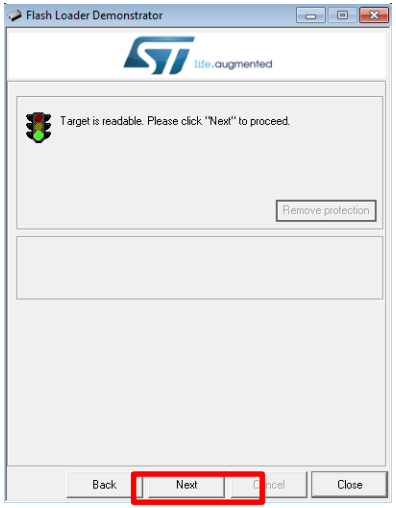
- Run flash loader with X-NUCLEO IDW04A1 board

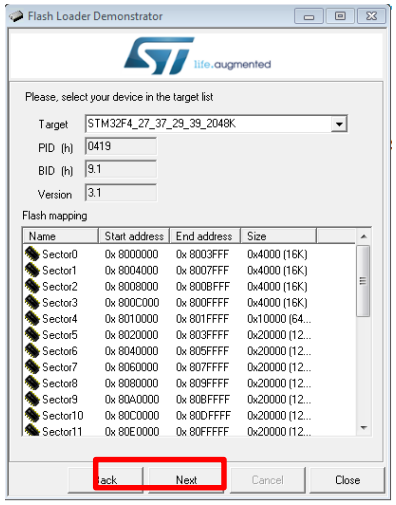


- Select COM port
  - (If not, use the PC's Device Manager to load the device driver. The USB to UART bridge should be in the list of "Ports (COM & LPT)" devices.)
- Set correct settings
  - Baud Rate =115200
  - Parity = Even
  - Echo Disabled
  - Timout 20
- Click the "Next" button.

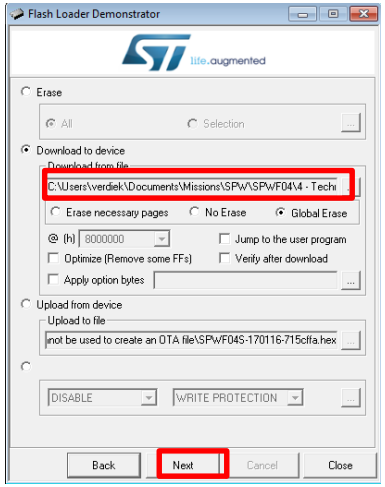
Lab 3.1 : Upgrade through UART

- X-NUCLEO-IDW04A1 board is ready for programming.



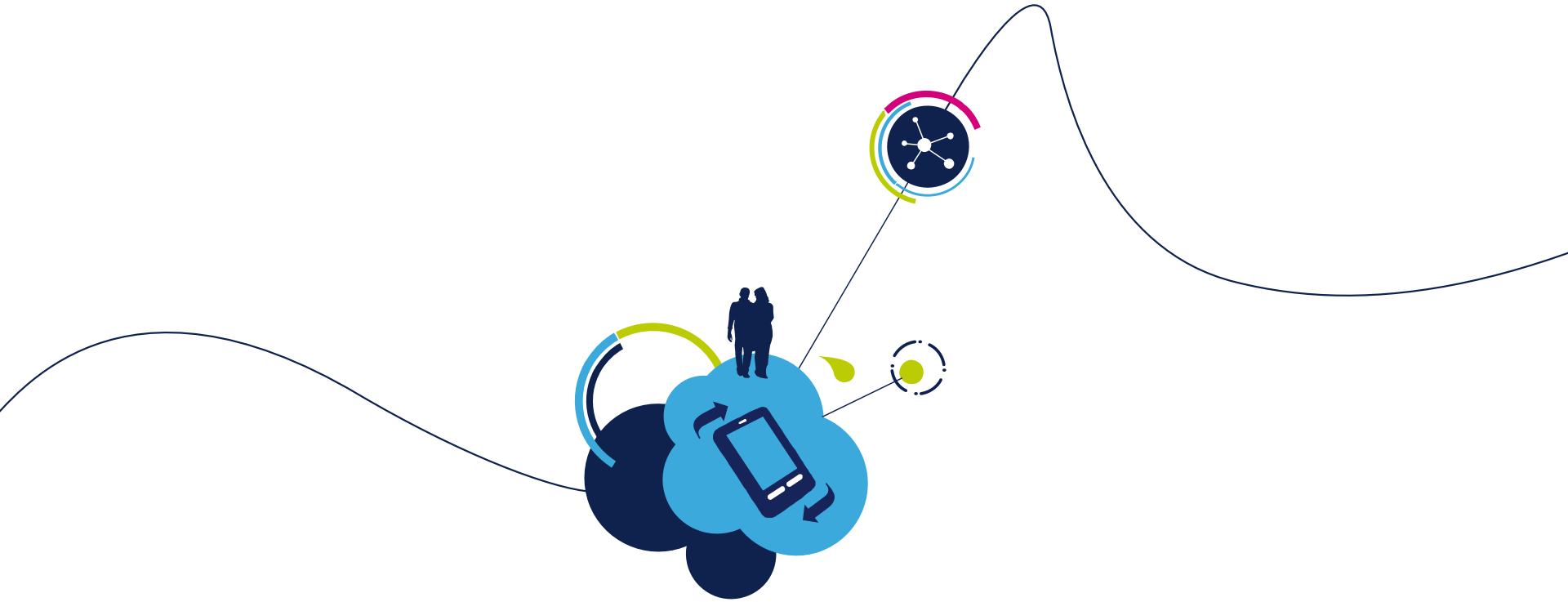


- If communication is OK click the “Next” button.



- Select FW hex file and click on « next ».

- At the end of FW upgrade , remove JP2 jumper and press RESET button SW1 on X-NUCLEO-IDW04A1 board.



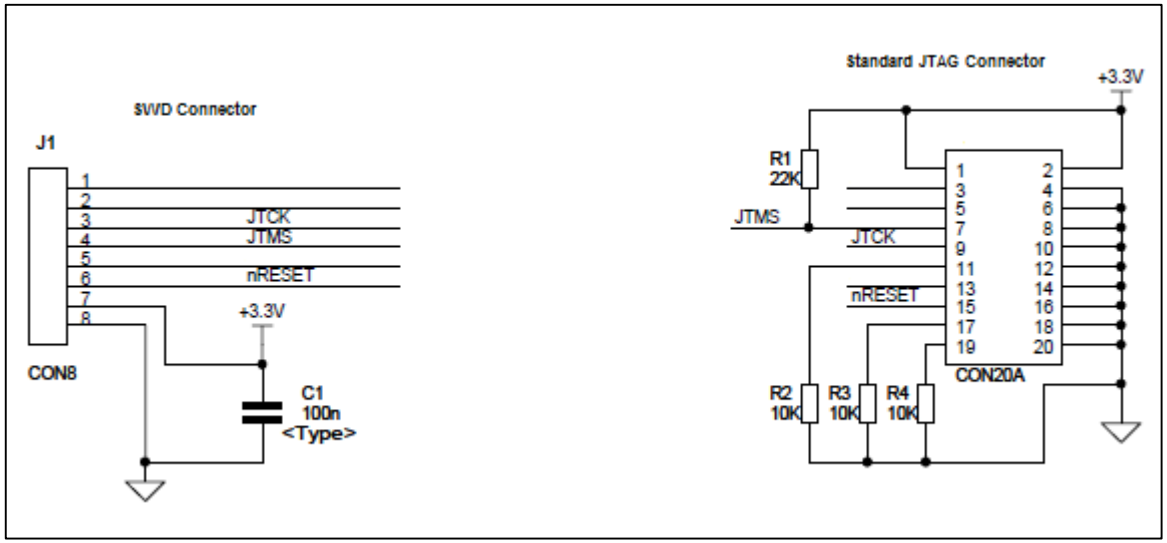
Proceed to the next LAB!

Lab 3.2 : Upgrade through SWD

- Objective
  - Upgrade the FW using SWD
- Prerequisites
  - HEX file (provided in the SPWF04S FW package)

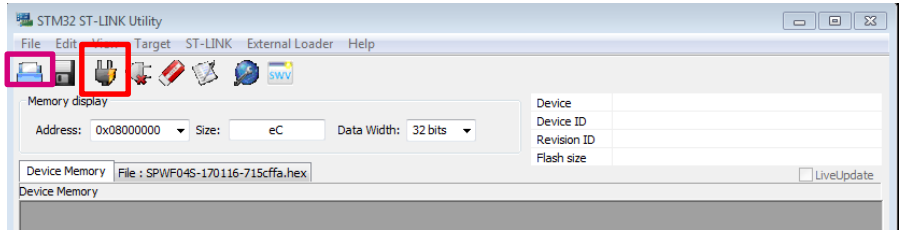
Lab 3.2 : Upgrade through SWD

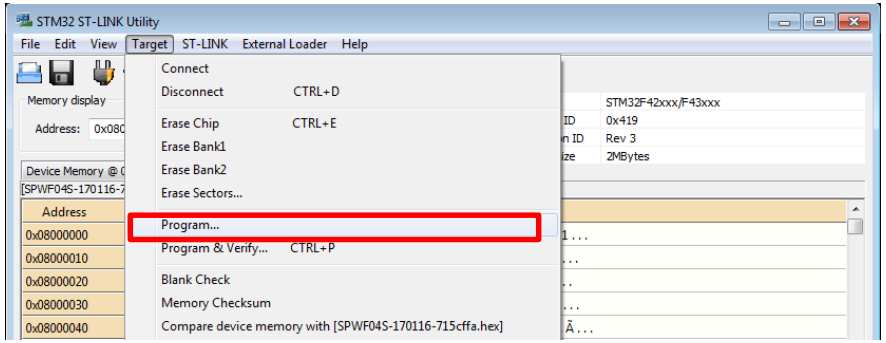
- X-NUCLEO-IDW04A1 J1 Connector details

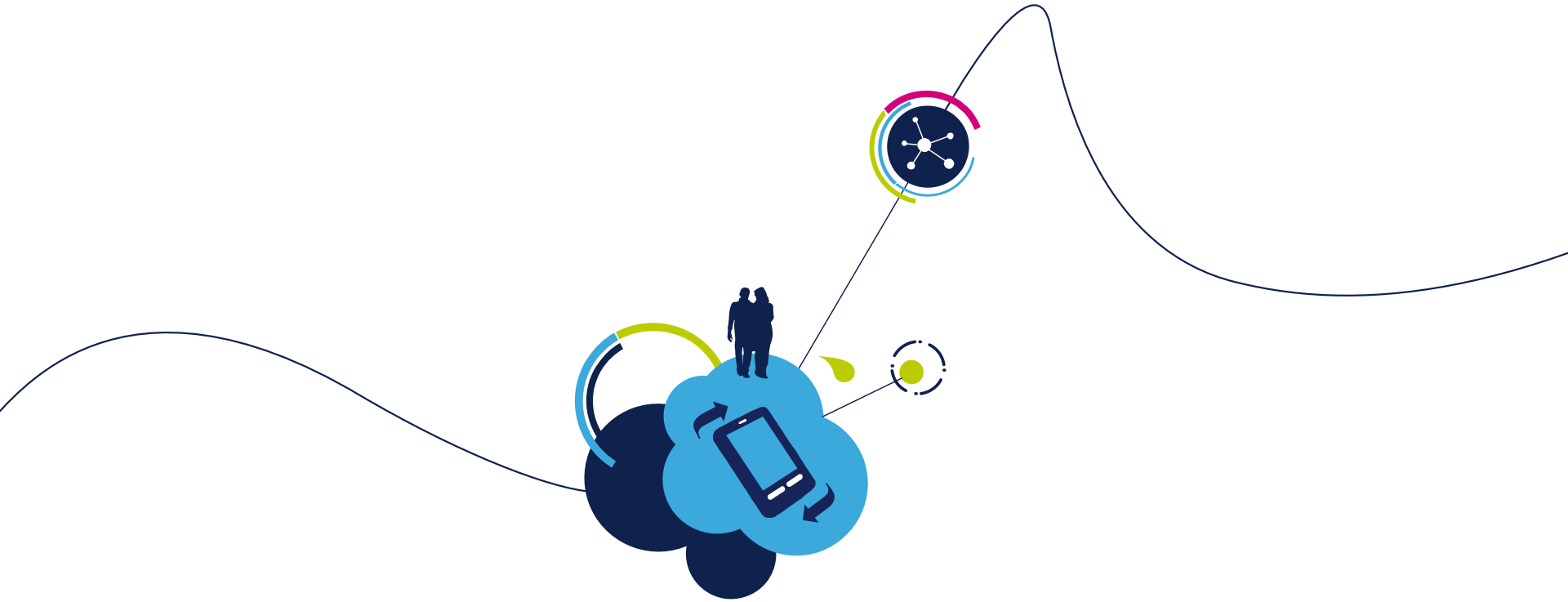


Lab 3.2 : Upgrade through SWD

- Download & install ST-Link Utility tool
  - http://www.st.com/content/st_com/en/products/embedded-software/development-tool-software/stsw-link004.html
- Program SPWF04 through SWD
  - Click on Target button
  - Click on open button
  - Select hex file
  - Program SPWF04







Proceed to the next LAB!

Lab 3.3 : FOTA update

- Objective
  - Upgrade the Firmware using the FOTA file
- Prerequisites
  - FOTA file (provided in the SPWF04S FW package)
  - External web server (i.e. Apache web server running on PC)

Lab 3.3 : FOTA update

- The SPWF04S module allows performing a Firmware Over-the-air update via a single HTTP (or HTTPS) GET.
- The SPWF04S will validate the firmware image it downloads, load it into a staging area, then prompt the user to issue a reset command in order to complete the update.

Lab 3.3 : FOTA update

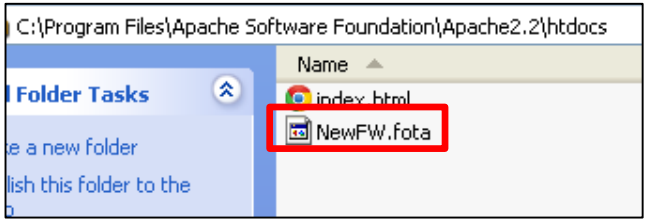
- **Apache Web Server** will be used in this LAB

(Apache Web Server is available at this link:

http://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.25-win32-x86-openssl-0.9.8y.msi)

Note: please check that the local firewall is disabled or properly set. It can block the connection from module to Apache.

- Copy the OTA file (i.e. SPWF04S-xxxxxxx-yyyyyyy-Release.fota) in the Apache 2.2 htdocs folder



Lab 3.3 : FOTA update

The FWUPDATE command allows to perform a Firmware Over-the-air update via a single HTTP (or HTTPS) GET.

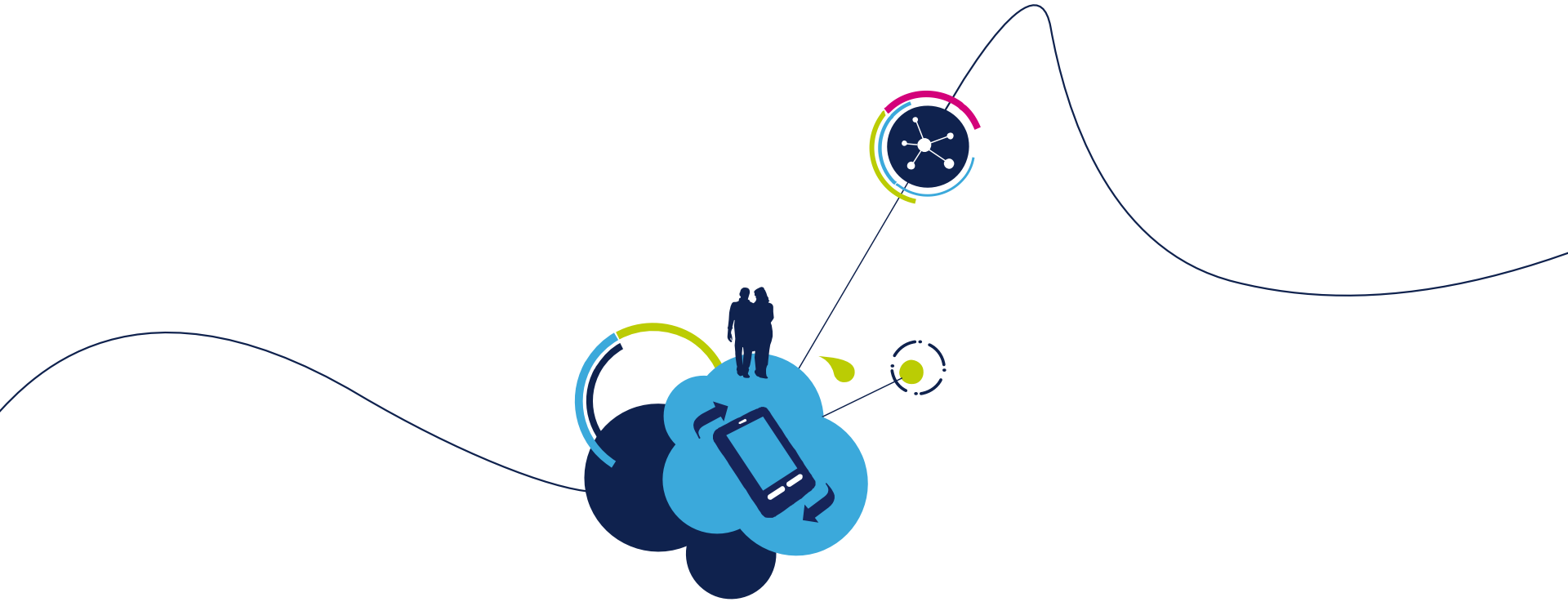
- Syntax
  - AT+S.FWUPDATE=e,<hostname>,[<path&queryopts>],[<port>],[<TLS>],[<username>],[<passwd>]<cr>
- Configuration parameters
  - <hostname> Target host. DNS resolvable name or IP address.
  - <path&queryopts> Default: /fw.ota. Document path and optional query arguments. If a secure FOTA is required, the extension of the file needs to be ".sfota".
  - <port> Default 80 (if TLS=0) or 443 (if TLS>0).
  - <TLS> Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS
  - <username> Default: none.
  - <passwd> Default: none.

Lab 3.3 : FOTA update

- The module and the Apache Web server must be connected to the same network
- In Tera Term: type **AT+S.FWUPDATE=e,192.168.x.yyy,\SPWF04S-xxxxxxx-yyyyyyy-Release.fota,,,,**
- Reset the module to apply the new FW
  - Type **AT+S.RESET**
- Restore factory default settings (mandatory)
  - Type **AT+S.FCFG**

Tera Term output

```
AT+S.FWUPDATE=e,192.168.x.yyy,\SPWF04S-xxxxxxx-yyyyyyy-Release.fota,,,,
AT-S.Write chunk:2048:8110000
AT-S.Write chunk:2048:8110800
[…]
AT-S.Write chunk:2048:81F0000
AT-S.Write chunk:740:81F0800:920292
AT-S.OK
AT+S.RESET
+WIND:2:Reset
+WIND:17:Boot:1.0
+WIND:17:Performing F/W update
+WIND:17:Completed F/W update
+WIND:17:Cleanup
AT+S.FCFG
AT-S.OK
```



Proceed to the next LAB!

Lab 3.4 : SFOTA update

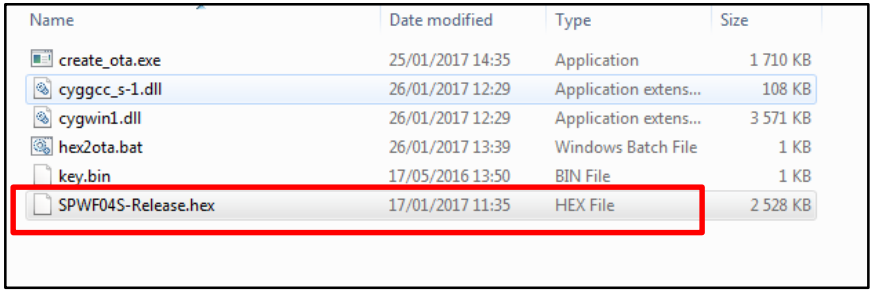
- Objective
  - Upgrade the Firmware using the SFOTA file
- Prerequisites
  - SFOTA Creation Folder
  - HEX file
  - Key.bin file
  - External web server (i.e. Apache web server running on PC)

Lab 3.4 : SFOTA update

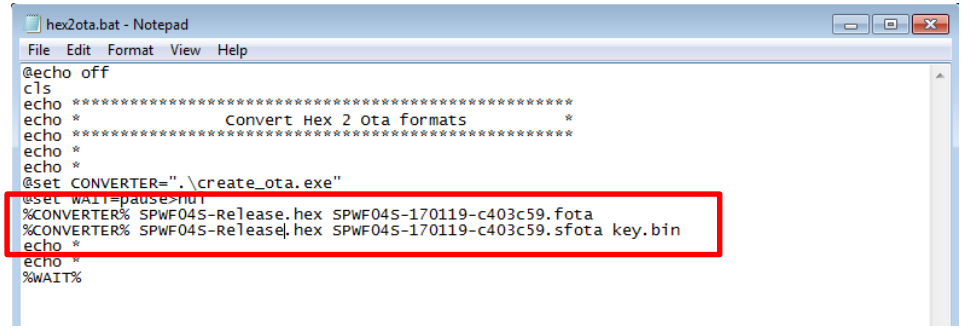
- The SPWF04S module allows performing a Secure Firmware Over-the-air update via a single HTTP (or HTTPS) GET.
- The SPWF04S will validate the firmware image it downloads, load it into a staging area, decrypt it, and then prompt the user to issue a reset command in order to complete the update.

Lab 3.4 : SFOTA update

- Drag and drop SPWF04 FW HEX file to the SFOTA Creation Folder

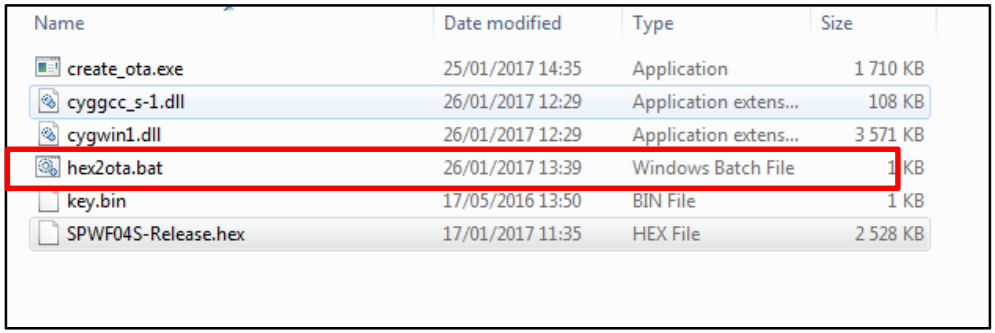


- Edit the hex2ota.bat file and set name of your HEX file (here SPWF04S-Release.hex) accordingly.

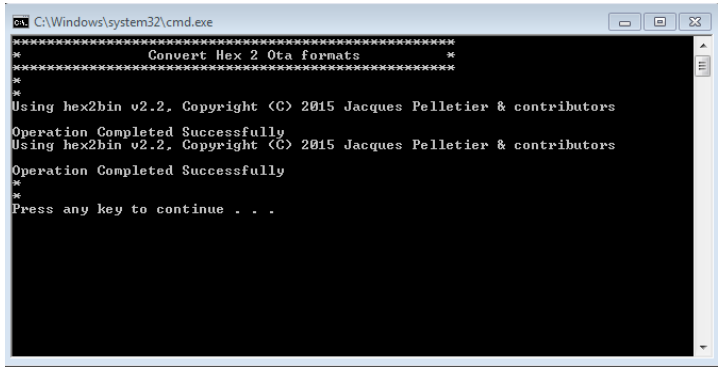


Lab 3.4 : SFOTA update

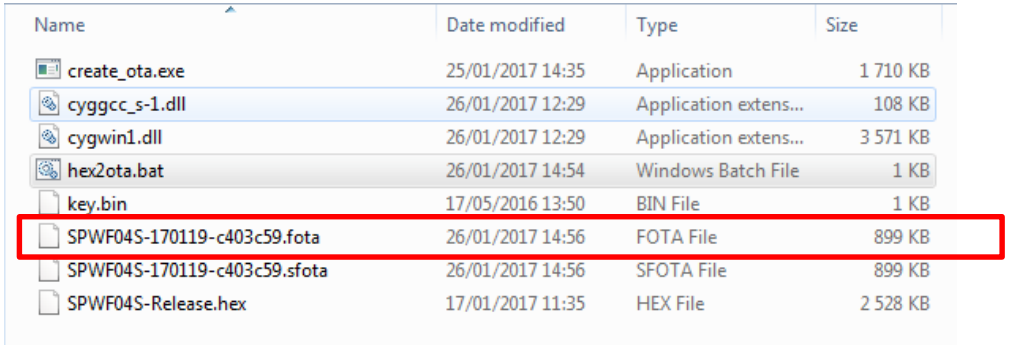
- Launch the bat file



Name	Date modified	Type	Size
create_ota.exe	25/01/2017 14:35	Application	1 710 KB
cyggcc_s-1.dll	26/01/2017 12:29	Application extens...	108 KB
cygwin1.dll	26/01/2017 12:29	Application extens...	3 571 KB
hex2ota.bat	26/01/2017 13:39	Windows Batch File	1 KB
key.bin	17/05/2016 13:50	BIN File	1 KB
SPWF04S-Release.hex	17/01/2017 11:35	HEX File	2 528 KB



- OTA file is created



Name	Date modified	Type	Size
create_ota.exe	25/01/2017 14:35	Application	1 710 KB
cyggcc_s-1.dll	26/01/2017 12:29	Application extens...	108 KB
cygwin1.dll	26/01/2017 12:29	Application extens...	3 571 KB
hex2ota.bat	26/01/2017 14:54	Windows Batch File	1 KB
key.bin	17/05/2016 13:50	BIN File	1 KB
SPWF04S-170119-c403c59.fota	26/01/2017 14:56	FOTA File	899 KB
SPWF04S-170119-c403c59.sfota	26/01/2017 14:56	SFOTA File	899 KB
SPWF04S-Release.hex	17/01/2017 11:35	HEX File	2 528 KB

Lab 3.4 : SFOTA update

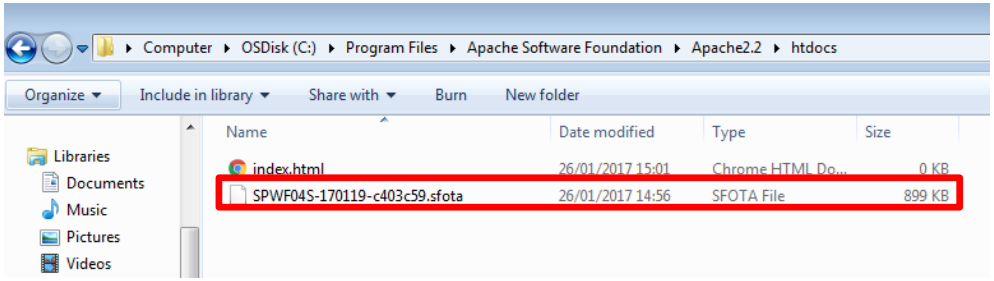
The **Apache Web Server** will be used

(Apache Web Server is available at this link:

http://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.25-win32-x86-openssl-0.9.8y.msi)

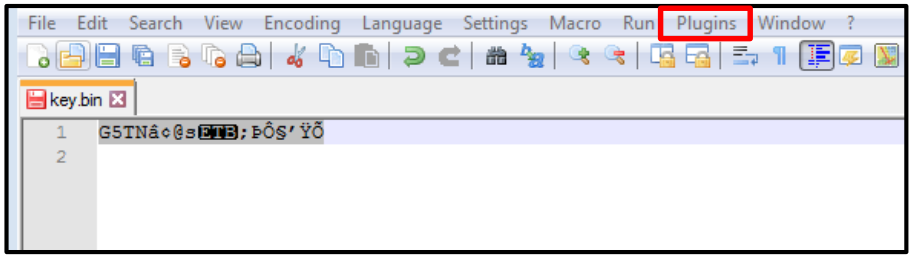
Note: please check that the local firewall is disabled or properly set. It can block the connection from module to Apache.

- Copy the SFOTA file (i.e. SPWF04S-xxxxxxx-yyyyyyy-Release.sfota) in the Apache 2.2 htdocs folder

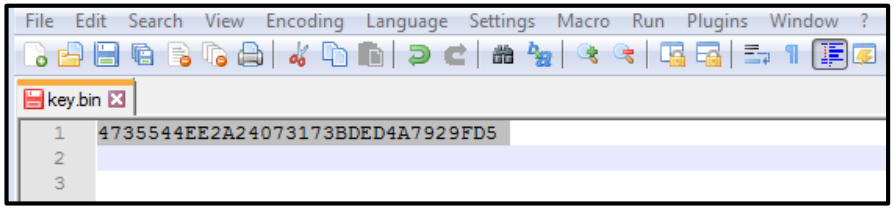


Lab 3.4 : SFOTA update

- Open the key.bin file, in the FOTA Creation Folder, with Notepad ++ for example



- In the Plugins tab, choose ASCII -> HEX in Converter



- This is the key you have to enter in the aes128_key variable of the module, look how do that in the next slide

Lab 3.4 : SFOTA update

- Type **AT+S.SCFG=**aes128_key,xx:yy:xx:yy:xx:yy:xx:yy:xx:yy:xx:yy:xx:yy:xx:yy
- type **AT+S.WCFG**

Tera Term output

AT+S.SCFG=aes128_key,xx:yy:xx:yy:xx:yy:xx:yy:xx:yy:xx:yy:xx:yy:xx:yy
AT-S.OK

AT+S.WCFG
AT-S.OK

Lab 3.4 : SFOTA update

The FWUPDATE command allows to perform a Secure Firmware Over-the-air update via a single HTTP (or HTTPS) GET.

- Syntax
  - AT+S.FWUPDATE=e,<hostname>,[<path&queryopts>],[<port>],[<TLS>],[<username>],[<passwd>]<cr>
- Configuration parameters
  - <hostname> Target host. DNS resolvable name or IP address.
  - <path&queryopts> Default: /fw.ota. Document path and optional query arguments. If a secure FOTA is required, the extension of the file needs to be ".sfota".
  - <port> Default 80 (if TLS=0) or 443 (if TLS>0).
  - <TLS> Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS
  - <username> Default: none.
  - <passwd> Default: none.

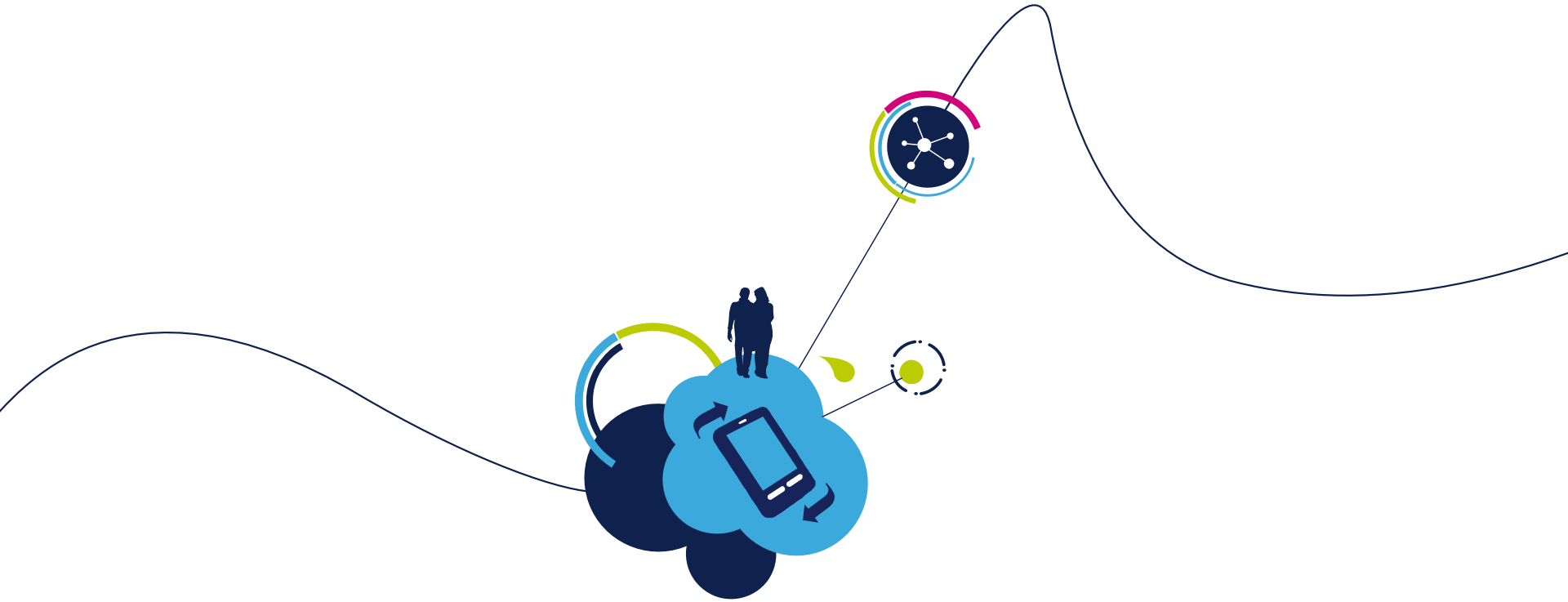
Lab 3.4 : SFOTA update

The module and the Apache Web server must be connected to the same network

- In Tera Term: type **AT+S.FWUPDATE=e,192.168.x.yyy,\SPWF04S-xxxxxxx-yyyyyyy-Release.sfota,,,,**
- Reset the module to apply the new FW
  - Type **AT+S.RESET**
- Restore factory default settings (mandatory)
  - Type **AT+S.FCFG**

Tera Term output

```
AT+S.FWUPDATE=e,192.168.x.yyy,\SPWF04S-xxxxxxx-yyyyyyy-Release.sfota,,,,
AT-S.Write chunk:2048:8110000
AT-S.Write chunk:2048:8110800
[…]
AT-S.Write chunk:2048:81F0000
AT-S.Write chunk:740:81F0800:920292
AT-S.OK
AT+S.RESET
+WIND:2:Reset
+WIND:17:Boot:1.0
+WIND:17:Performing F/W update
+WIND:17:Completed F/W update
+WIND:17:Cleanup
AT+S.FCFG
AT-S.OK
```



Proceed to the next LAB!

Lab 4 : Used modes

- Objective
  - Create a direct connection between the module and a device or an Access Point

- Prerequisites
  - Work alone

Lab 4.1 : MiniAP mode

- Objective
  - Create a direct connection between the module and an end device

- Prerequisites
  - Work alone

Lab 4.1 : Configuring the module in MiniAP mode

- Type **AT+S.WIFI=0**

Set the SSID

- Type **AT+S.SSIDTXT=SPWF04_AP**

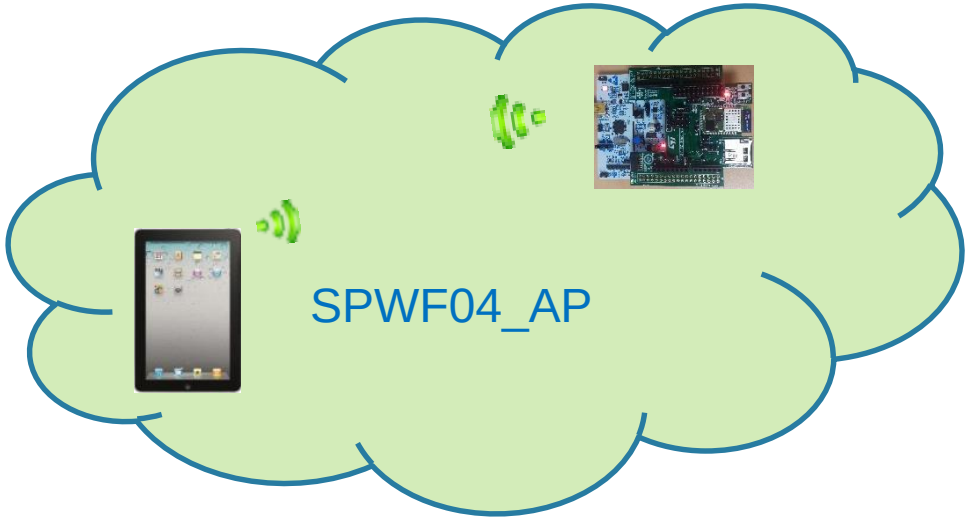
- Set the network privacy mode (0=OPEN, 1=WEP, 2=WPA)
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,3**
  - Type **AT+S.WIFI=1**

WPA2 available in MiniAP from FW1.1

Tera Term output

```
AT+S.WIFI=0
AT-S.OK
AT+S.SSIDTXT=SPWF04_AP
AT-S.OK
AT+S.SCFG=wifi_priv_mode,0
AT-S.OK
AT+S.SCFG=wifi_mode,3
AT-S.OK
AT+S.WIFI=1
AT-S.OK
```

Lab 4.1 : Configuring the module in MiniAP mode



Tera Term output

+WIND:32:WiFi Hardware Started
+WIND:26:WiFi Started AP:SPWF04_AP
+WIND:24:WiFi Up: 192.168.0.1

Lab 4.1 : Configuring the module in MiniAP mode (WEP Key)

Configure the module using the WEP key (2 possible combinations available)

Sample table:

AP configuration	AT command to be used
Security Mode: WEP WEP Key Length: 64 bit (10 hex digits) Authentication: Open Wep Key 1: 1234567890	AT+S.WIFI=0 AT+S.SSIDTXT=SPWF04_AP AT+S.SCFG=wifi_wep_keys[0],1234567890 AT+S.SCFG=wifi_wep_key_lens,05 AT+S.SCFG=wifi_auth_type,0 AT+S.SCFG=wifi_priv_mode,1 AT+S.SCFG=wifi_mode,3 AT+S.WIFI=1

AP configuration	AT command to be used
Security Mode: WEP WEP Key Length: 128 bit (26 hex digits) Authentication: Open Wep Key 1: 12345678901234567890123456	AT+S.WIFI=0 AT+S.SSIDTXT=SPWF04_AP AT+S.SCFG=wifi_wep_keys[0],1234567890123456789012345 AT+S.SCFG=wifi_wep_key_lens,0D AT+S.SCFG=wifi_auth_type,0 AT+S.SCFG=wifi_priv_mode,1 AT+S.SCFG=wifi_mode,3 AT+S.WIFI=1

Lab 4.1 : Configuring the module in MiniAP mode (WEP Key)

Notes:

- “wifi_wep_key_lens” variable values: 05 and 0D
- It’s possible to enter any text string as WEP key. It have to be converted into a hexadecimal key using the ASCII values of the characters. A maximum of 5 text characters can be entered for 64 bit keys, and a maximum of 13 characters for 128 bit keys.
  In this case, it needs to manually convert your ASCII password to HEX and complete the wifi_wep_keys[0] variable with the HEX value.

- i.e. WEP key: **test1**
  ASCII to HEX: **74:65:73:74:31**
  So, the AT command is: **AT+S.SCFG=wifi_wep_keys[0],7465737431**

Lab 4.1 : Customizing the MiniAP address (optional)

- Type **AT+S.WIFI=0**

- Set the SSID
  - Type **AT+S.SSIDTXT=SPWF04_AP**

- Set the network privacy mode (0=OPEN or 1=WEP are supported, WPA2 supported from FW1.1)
  - Type **AT+S.SCFG=wifi_priv_mode,0**

- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,3**

- Set the MiniAP address
  - Type **AT+S.SCFG=ip_ipaddr,192.168.0.1**
  - Type **AT+S.WIFI=1**

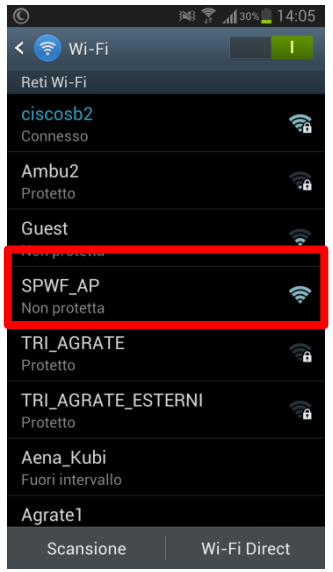
Tip: the MiniAP will assign sequential addresses to the client
i.e. 1° client: 192.168.0.2, 2° client: 192.168.0.3

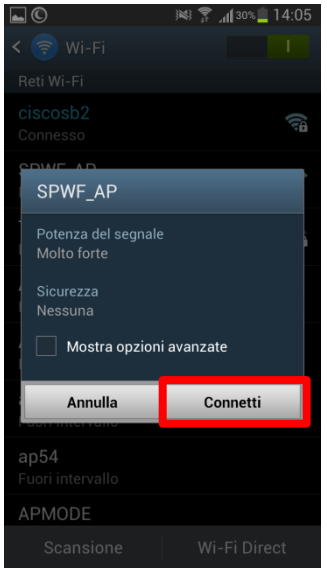
Tera Term output

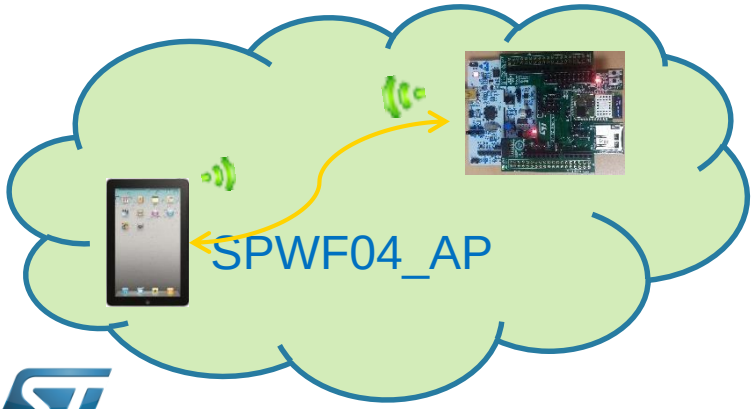
```
AT+S.WIFI=0
AT-S.OK
AT+S.SSIDTXT=SPWF04_AP
AT-S.OK
AT+S.SCFG=wifi_priv_mode,0
AT-S.OK
AT+S.SCFG=wifi_mode,3
AT-S.OK
AT+S.SCFG=ip_ipaddr,192.168.0.1
AT-S.OK
AT+S.WIFI=1
AT-S.OK
+WIND:32:WiFi Hardware Started
+WIND:26:WiFi Started AP with network 'SPWF04_AP'
+WIND:24:WiFi Up:192.168.0.1
```

Lab 4.1 : Mini AP mode

- Associate your end device to the SPWF04_AP network
  - Find the SPWF_AP network and connect the end device to the module



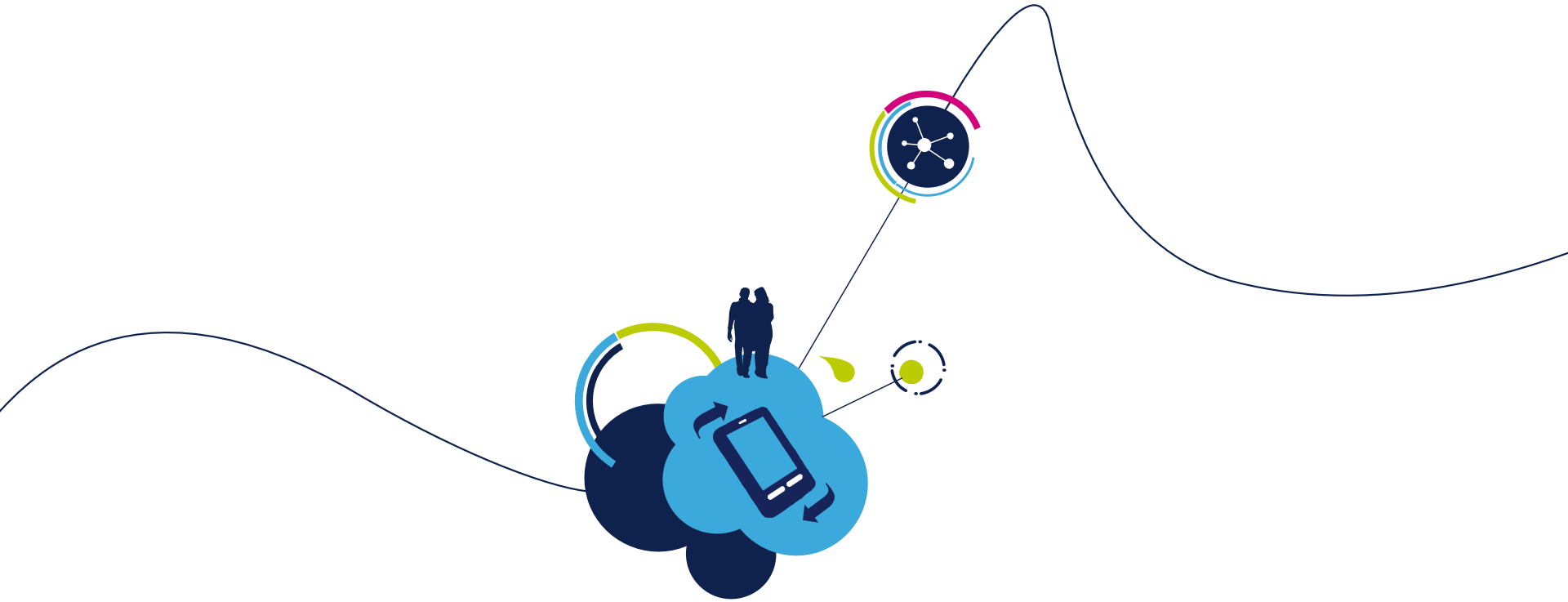




Tera Term output

```
+WIND:28:Station Associated:90:18:7C:96:0D:0B:0
+WIND:29:DHCP reply:192.168.0.2
```

life.augmented



Proceed to the next LAB!

Lab 4.2 : Station mode

- Objective
  - Connect the SPWF04 Module to an Access Point
- Prerequisites
  - USB dongle and computer are set up as described in Lab 2
  - Work alone

Lab 4.2.1 : Access point connection through UART

- Objective
  - Scan for available networks
  - Join a network
  - Check the status/statistics variables

- Prerequisites
  - Work alone

Lab 4.2.1 : Scan for available networks

The SCAN command performs an immediate scan for available networks. Infrastructure (AP) and IBSS (Ad-Hoc) networks are both reported. Network type, Channel, BSSID, SSID, Signal strength (RSSI), and 802.11 capabilities are all reported.

- Type **AT+S.SCAN=d,ScanResult** — Do a scan without any filter. Put the scans result in the file named ScanResult, if no “ScanResult” file is given, output will be show on console
- Type **AT+S.FSP=ScanResult,,** — Show the contains of the file ScanResult

Tera Term output

```
AT+S.SCAN=d,ScanResult
AT-S.Parsing Networks:3
AT-S.OK

AT+S.FSP=ScanResult,,
1:  BSS 14:D6:4D:24:36:00 CHAN: 01 RSSI: -28 SSID: 'ENG-WPA' CAPS: 0431 WPA WPA2
2:  BSS 00:18:0A:31:EA:78 CHAN: 11 RSSI: -82 SSID: 'ZyckoItalyWireless' CAPS: 0531 WPA WPA2
3:  BSS 06:18:0A:31:E7:E2 CHAN: 11 RSSI: -85 SSID: 'ZyckoItalyGuest' CAPS: 0531 WPA WPA2
```

Network type | Network MAC | Network channel | Network RSSI | Network SSID | Network capabilities

Lab 4.2.1 : Joining a network (WPA Key)

If the device isn't WPS compliant, the AP parameters setting is needed.

- Type **AT+S.WIFI=0**

- Set the SSID
  - Type **AT+S.SSIDTXT=ENG_WPA**

- Set the password
  - Type **AT+S.SCFG=wifi_wpa_psk_text,helloworld**

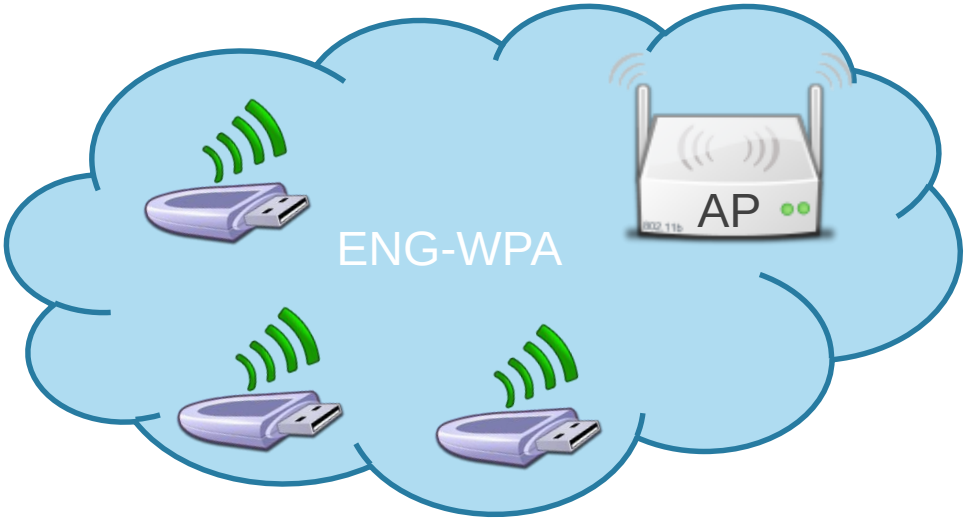
- Set the network privacy mode (0=none, 1=WEP, 2=WPA-Personal (TKIP/AES) or WPA2-Personal (TKIP/AES))
  - Type **AT+S.SCFG=wifi_priv_mode,2**
    - N.B. wifi_auth_type must be set to 0 → AT+S.SCFG=wifi_auth_type,0

- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,1**
  - Type **AT+S.WIFI=1**

Tera Term output

All
AT-S.OK

Lab 4.2.1 : Joining a network (WPA Key)



Tera Term output

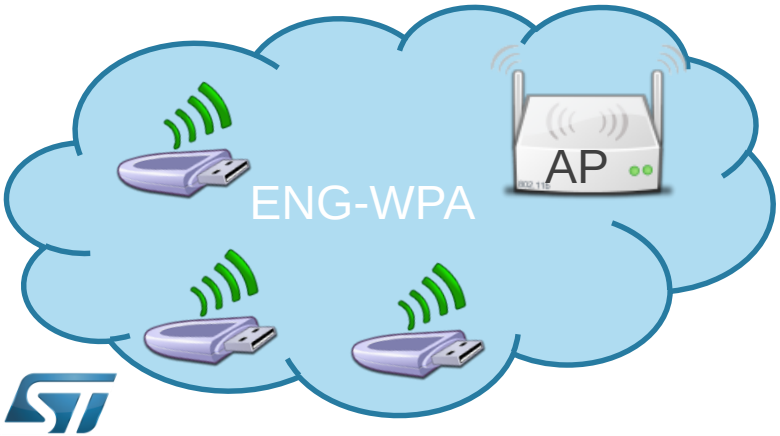
```
+WIND:46:WPA Crunching
PSK:mypassword:10
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join:14:D6:4D:24:36:00
+WIND:25:WiFi Association with 'ENG-
WPA' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up:192.168.0.1xx
```

Lab 4.2.1 : Joining a network (WPA Key)

- Check the status/statistics variables
  - Type **AT+S.STS**

```
AT-S.Var:ip_ipaddr=192.168.1.2
AT-S.Var:ip_netmask=255.255.255.0
AT-S.Var:ip_gw=192.168.1.1
AT-S.Var:ip_dns1=192.168.1.1
AT-S.Var:ip_dns2=0.0.0.0
```

- Send a ping to the gateway (ip_gw)
  - Type **AT+S.PING=,,192.168.0.1**



Tera Term output

```
AT+S.PING=,,192.168.0.1
AT-S.OK
```

Lab 4.2.1 : Joining a network (WEP Key)

Configure the module using the WEP key (4 possible combinations available)

Sample table:

AP configuration	AT command to be used
Security Mode: WEP WEP Key Length: 64 bit (10 hex digits) Authentication: Open Wep Key 1: 1234567890	AT+S.WIFI=0 AT+S.SSIDTXT=ENG-WEP AT+S.SCFG=wifi_wep_keys[0],1234567890 AT+S.SCFG=wifi_wep_key_lens,05 AT+S.SCFG=wifi_auth_type,0 AT+S.SCFG=wifi_priv_mode,1 AT+S.SCFG=wifi_mode,1 AT+S.WIFI=1
Security Mode: WEP WEP Key Length: 64 bit (10 hex digits) Authentication: Shared Key Wep Key 1: 1234567890	AT+S.WIFI=0 AT+S.SSIDTXT=ENG-WEP AT+S.SCFG=wifi_wep_keys[0],1234567890 AT+S.SCFG=wifi_wep_key_lens,05 AT+S.SCFG=wifi_auth_type,1 AT+S.SCFG=wifi_priv_mode,1 AT+S.SCFG=wifi_mode,1 AT+S.WIFI=1

AP configuration	AT command to be used
Security Mode: WEP WEP Key Length: 128 bit (26 hex digits) Authentication: Open Wep Key 1: 12345678901234567890123456	AT+S.WIFI=0 AT+S.SSIDTXT=ENG-WEP AT+S.SCFG=wifi_wep_keys[0],12345678901234567890123456 AT+S.SCFG=wifi_wep_key_lens,0D AT+S.SCFG=wifi_auth_type,0 AT+S.SCFG=wifi_priv_mode,1 AT+S.SCFG=wifi_mode,1 AT+S.WIFI=1
Security Mode: WEP WEP Key Length: 128 bit (26 hex digits) Authentication: Shared Key Wep Key 1: 12345678901234567890123456	AT+S.WIFI=0 AT+S.SSIDTXT=ENG-WEP AT+S.SCFG=wifi_wep_keys[0],12345678901234567890123456 AT+S.SCFG=wifi_wep_key_lens,0D AT+S.SCFG=wifi_auth_type,1 AT+S.SCFG=wifi_priv_mode,1 AT+S.SCFG=wifi_mode,1 AT+S.WIFI=1

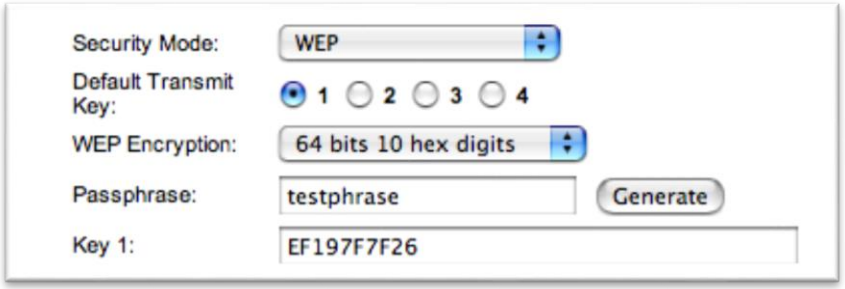
Lab 4.2.1 : Joining a network (WEP Key)

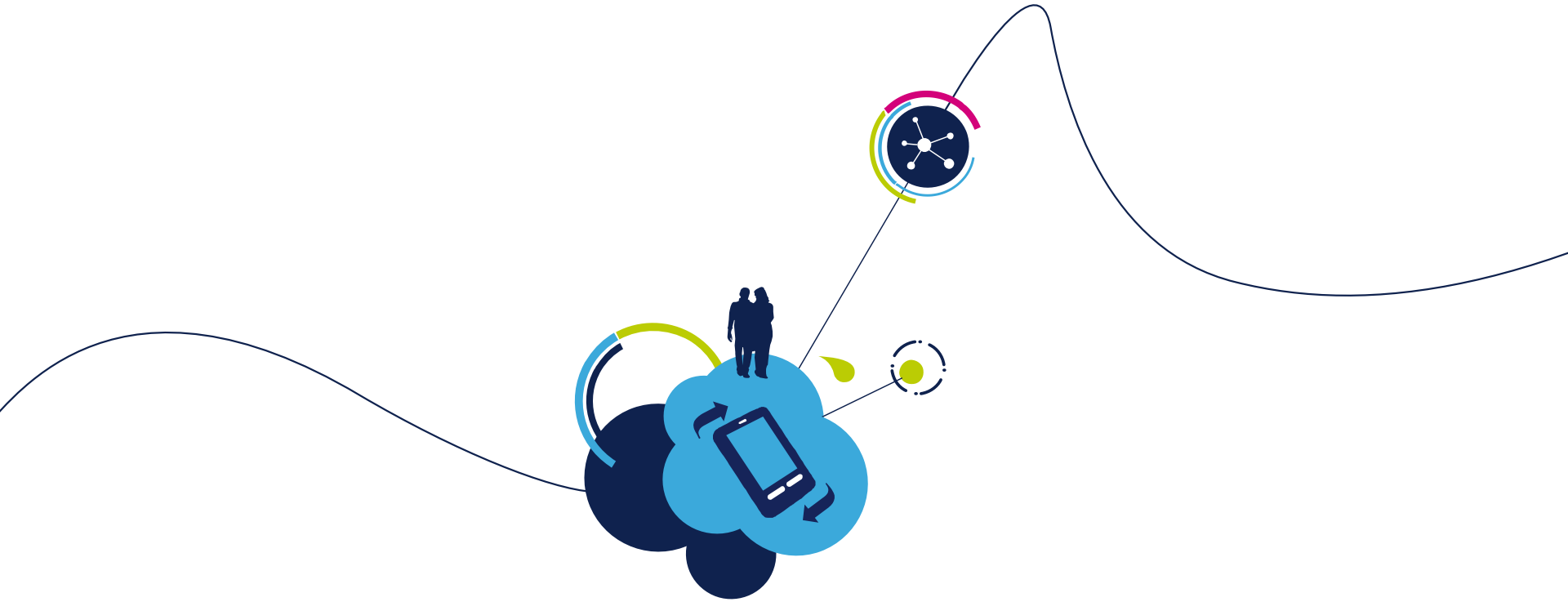
Notes:

- “wifi_wep_key_lens” variable values: 05 and 0D
- It’s possible to enter any text string into a WEP key box in the AP, in which case it will be converted into a hexadecimal key using the ASCII values of the characters. A maximum of 5 text characters can be entered for 64 bit keys, and a maximum of 13 characters for 128 bit keys.
  In this case, it needs to manually convert your ASCII password to HEX and complete the wifi_wep_keys[0] variable with the HEX value.

- i.e. AP WEP key: **test1**
  ASCII to HEX: **74:65:73:74:31**
  So, the AT command is: **AT+S.SCFG=wifi_wep_keys[0],7465737431**

- Some APs allow user to insert a passphrase and then the AP automatically generates the hex keys.
  In this scenario, user have not to perform the ASCII to HEX conversion because the AP already gives it the hex value.





Proceed to the next LAB!

Lab 4.2.2 : Access point connection through WPS

- Objective
  - Create a direct connection between the module and a generic AP with WPS option

- Prerequisites
  - Work alone

Lab 4.2.2 : Joining a network through WPS

In order to be connected to an available Wifi network, the AP parameters setting was needed, but now, with the WPS, it isn't longer the case.

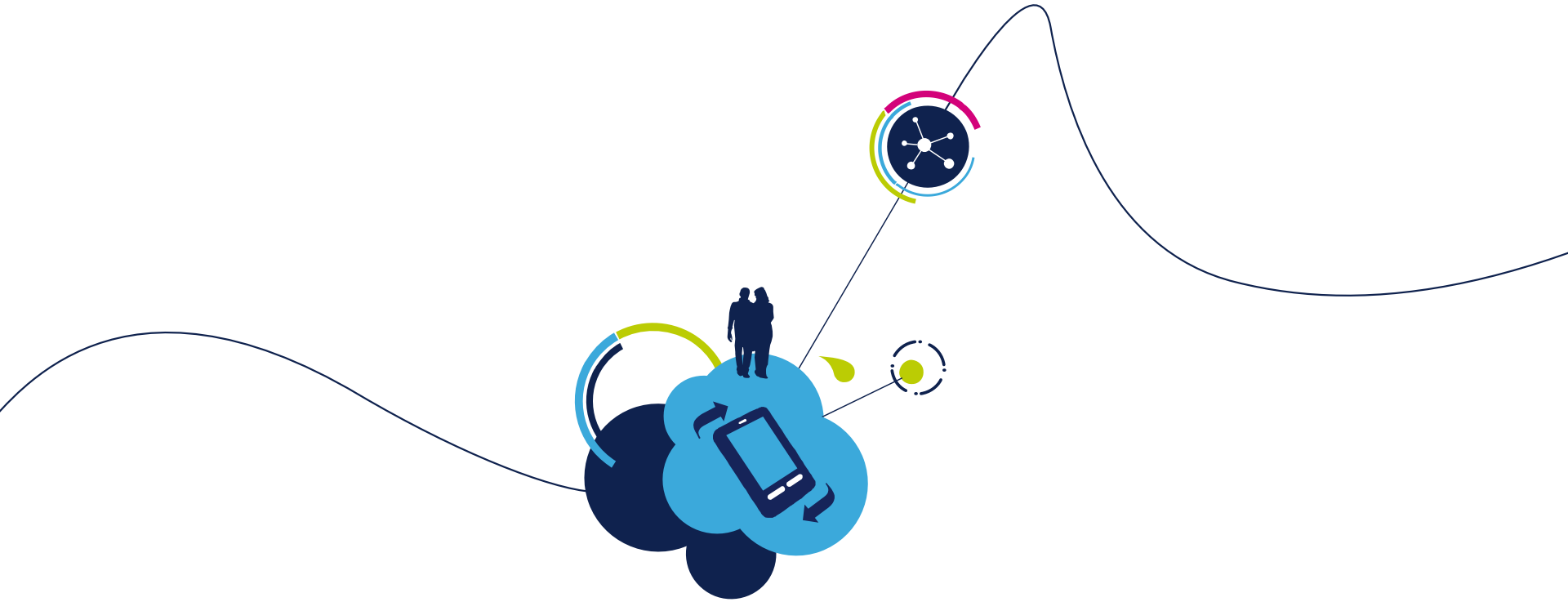
- **There are 2 options**

  - **Hardware : Press the WPS button of the AP and the SW2 of the SPWF04**

  - **Software : Press the WPS button of the AP and Type AT+S.WPS=0 in Tera Term**

Tera Term output

```
AT+S.WPS=0
AT-S.OK
+WIND:46:WPA Crunching
PSK:mypassword:10
+WIND:25:WiFi Association successful:AP
```



Proceed to the next LAB!

Lab 4.2.3 : MiniAP mode for the first set

- Objective
  - Create a direct connection between the module and an end device
  - First set of the module in order to enable the connection between the module and a generic AP

- Prerequisites
  - Work alone

Lab 4.2.3 : Configuring the module in MiniAP mode

- Type **AT+S.WIFI=0**

Set the SSID

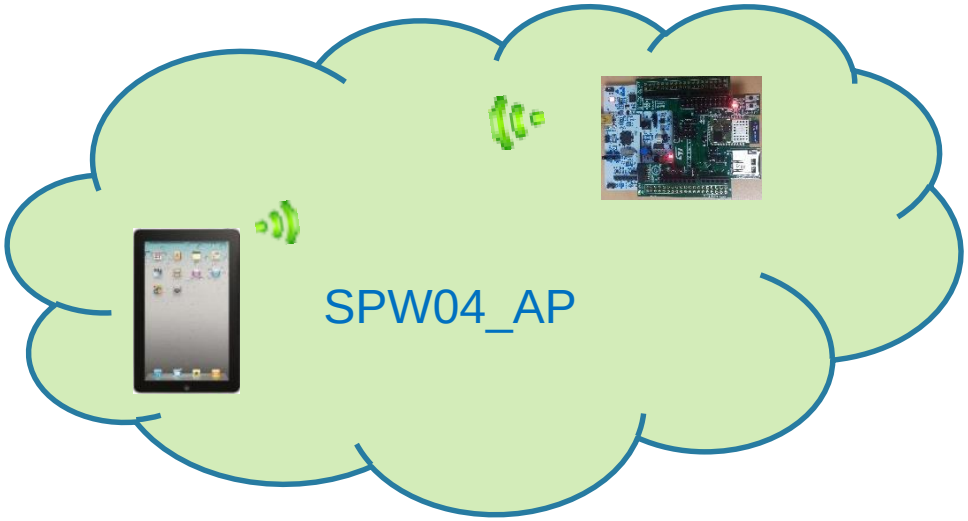
- Type **AT+S.SSIDTXT=SPWF04_AP**

- Set the network privacy mode (0=OPEN or 1=WEP are supported)
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,3**
  - Type **AT+S.WIFI=1**

Tera Term output

```
AT+S.WIFI=0
AT-S.OK
AT+S.SSIDTXT=SPWF04_AP
AT-S.OK
AT+S.SCFG=wifi_priv_mode,0
AT-S.OK
AT+S.SCFG=wifi_mode,3
AT-S.OK
AT+S.WIFI=1
AT-S.OK
```

Lab 4.2.3 : Configuring the module in MiniAP mode

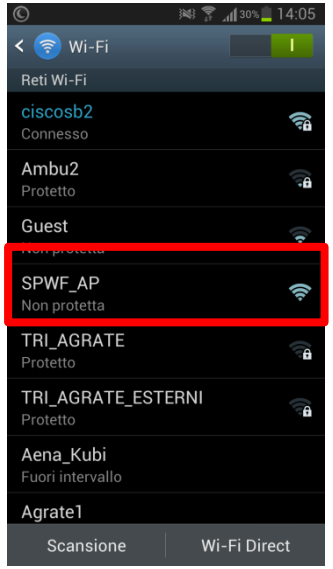


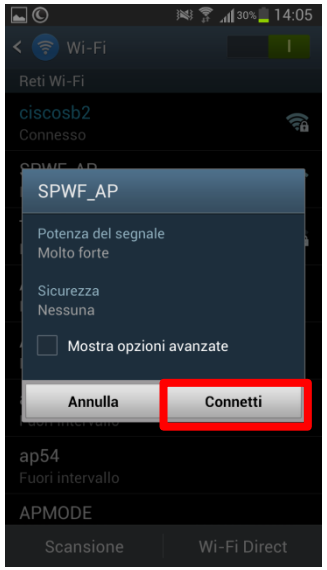
Tera Term output

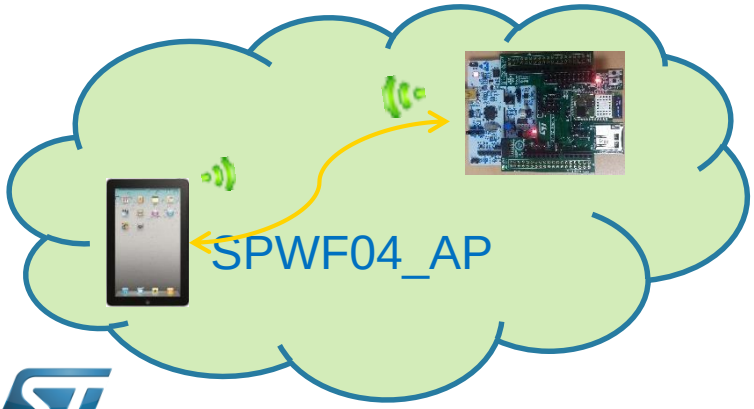
```
+WIND:32:WiFi Hardware Started
+WIND:26:WiFi Started
AP:SPWF04_AP
+WIND:24:WiFi Up:192.168.0.1
```

Lab 4.2.3 : Mini AP mode

- Associate your end device to the SPWF04_AP network
  - Find the SPWF04_AP network and connect the end device to the module







Tera Term output

+WIND:28:Station Associated:90:18:7C:96:0D:0B:0
+WIND:29:DHCP reply:192.168.0.2

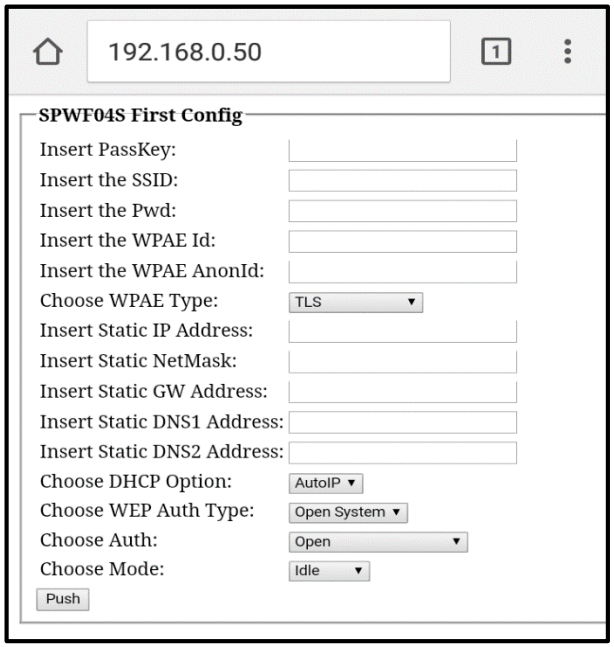
life.augmented

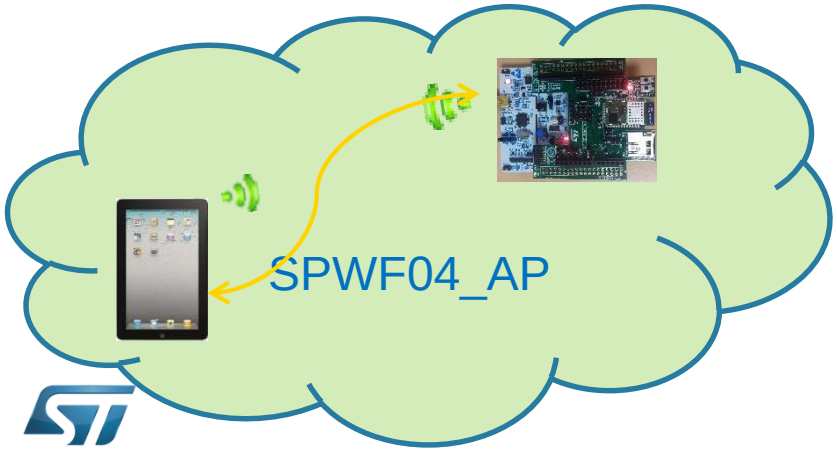
Lab 4.2.3 : Mini AP mode

- Open your web browser

- In the address bar, type the value of the variable ip_ipaddr or capitveportal.net

Tip: If the AP domain name is not quickly opened, it's suggested to turn off an eventual proxy server (check the connection settings or browser preferences)





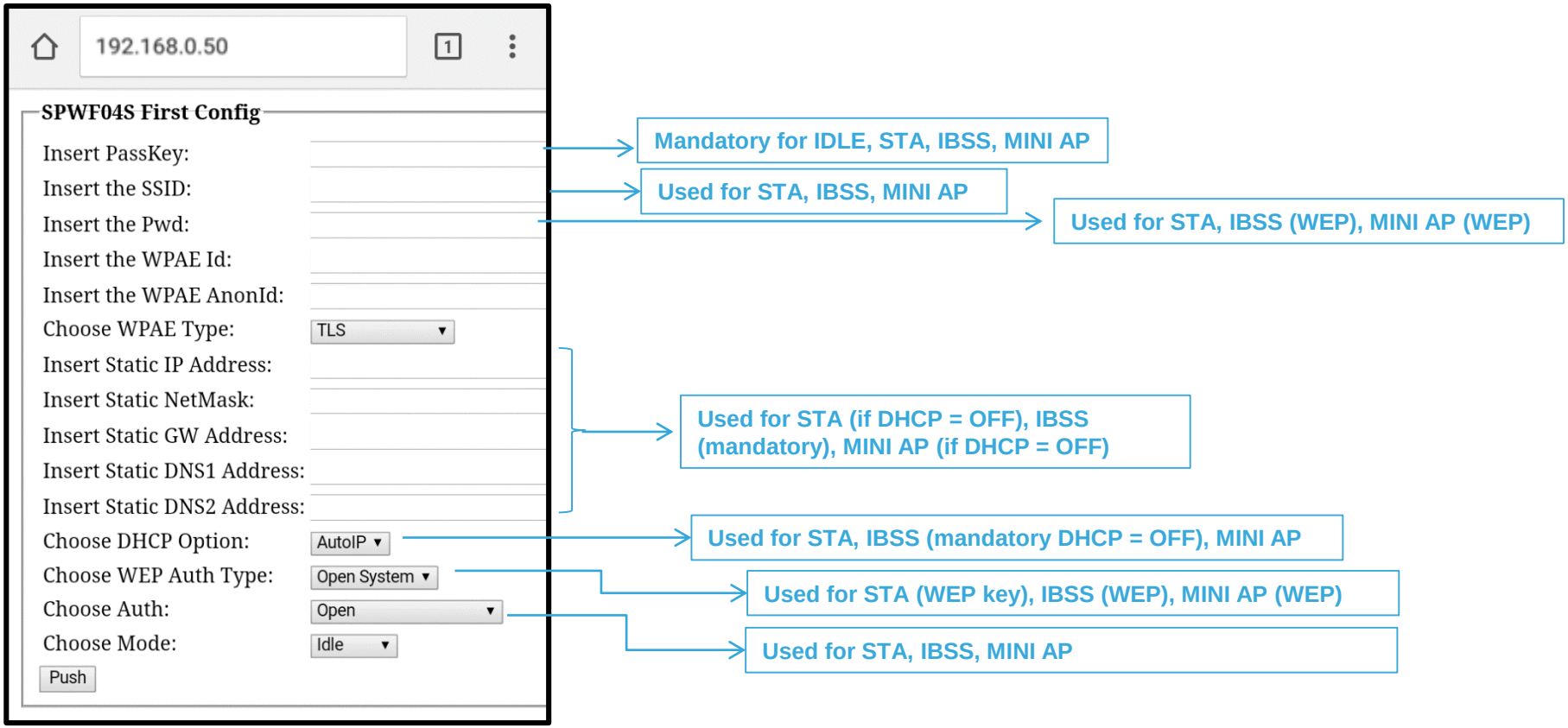
Tip: The Mini AP domain can be set using the variable ip_apdomainname.
The default value is "captiveportal.net".

The Mini AP default homepage can be set using the variable ip_apredirect.
The default value is "firstset.html".

life.augmented

Lab 4.2.3 : First Set Page

- The first set page allows to configure the module in IDLE mode, STATION mode, IBSS mode and MINI AP mode.

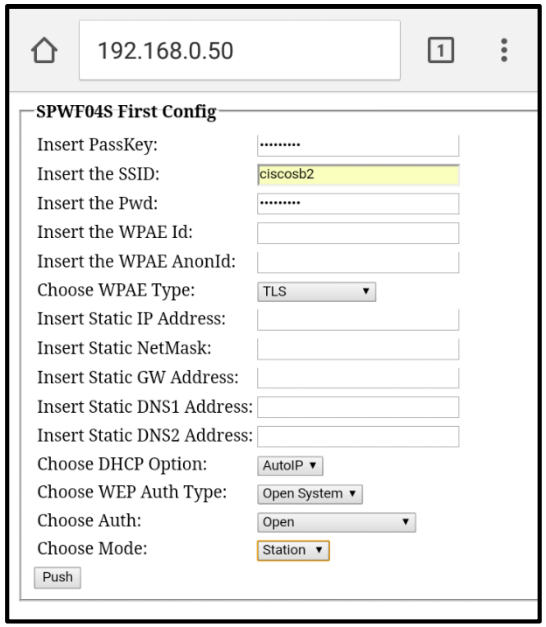


Lab 4.2.3 : Mini AP mode

- Set all the parameters required in order to enable the connection between the module and a generic AP

  i.e. AP configured in WPA/WPA2 mode

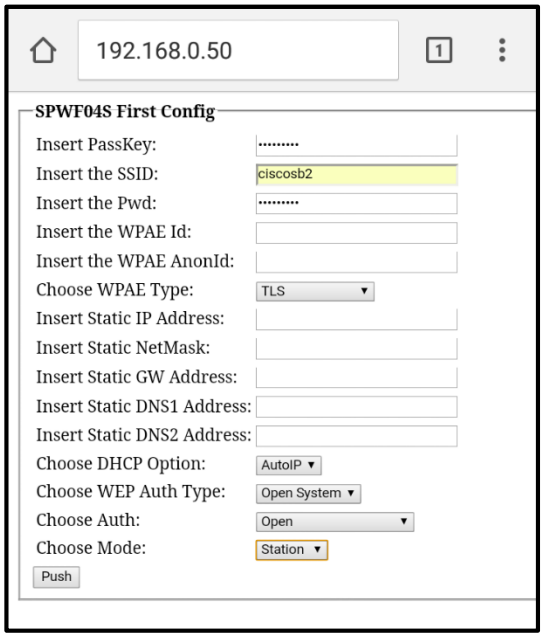
  - MiniAP PassKey: anonymous (by default)
  - SSID of the access point
  - Password of the access point
  - Authentication type of the access point
  - Use mode of the module

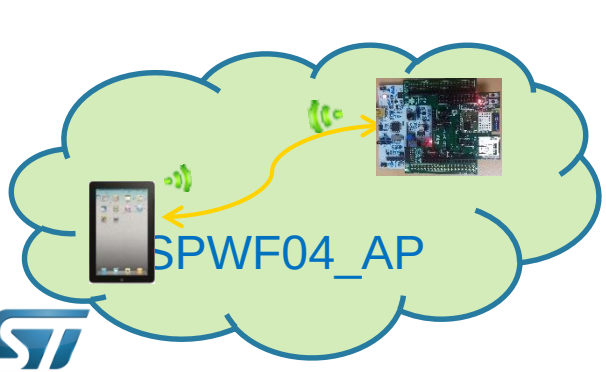


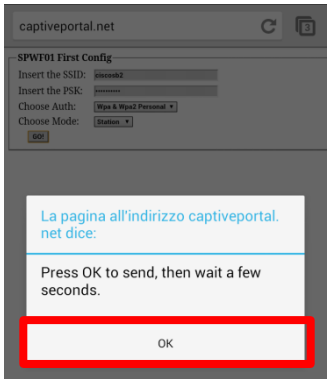
Tip: The Mini AP PassKey can be set using the variable "user_desc".
The default value is "anonymous".

Lab 4.2.3 : Mini AP mode

- Click on the Push button and then send the parameters confirming with OK



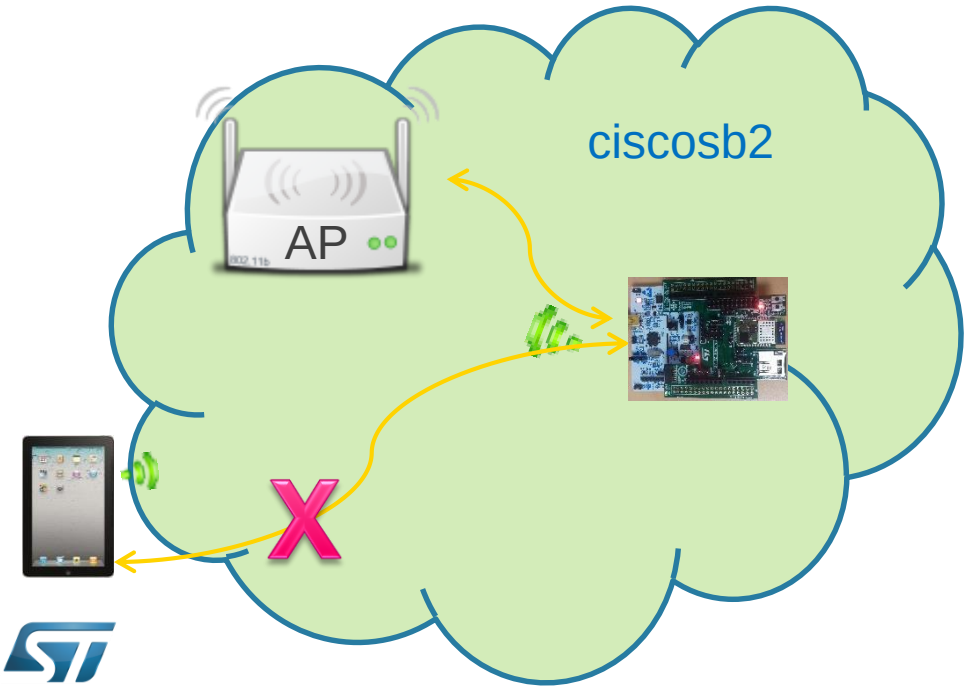




life.augmented

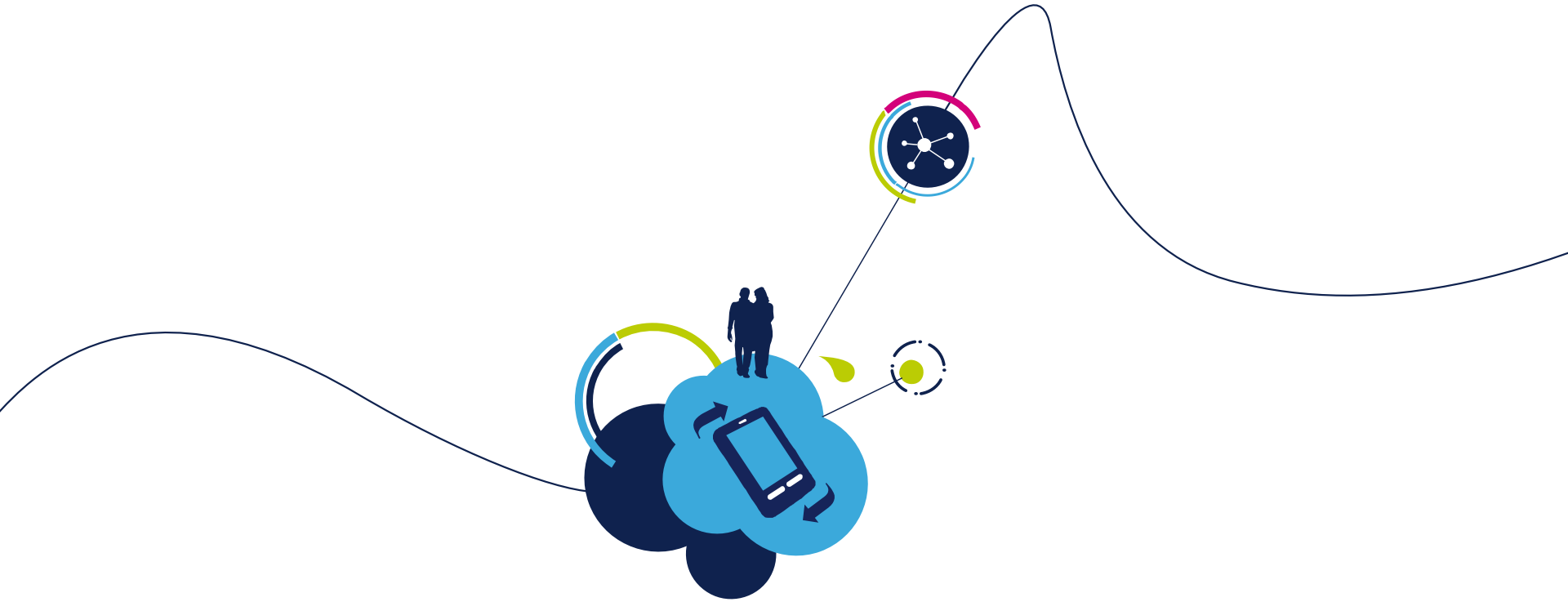
Lab 4.2.3 : Mini AP mode

- The module will receive the parameters and will automatically connect to the access point required



Tera Term output

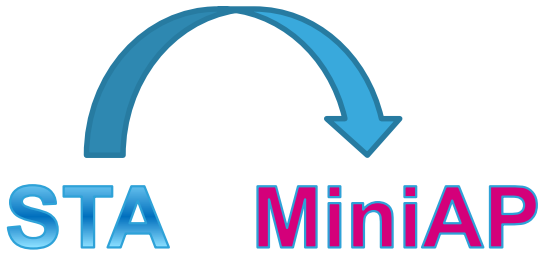
```
+WIND:57:Received SSID is ciscosb2
+WIND:57:Received PWD is **********
+WIND:57:Received Auth mode is 2
+WIND:57:Received Mode is 1
+WIND:2:Reset
+WIND:1:Poweron:xxxxxxx-yyyyyyy-SPWF04S
+WIND:13:Copyright (c) 2012-2017 STMicroelectronics,
Inc.  All rights Reserved:SPWF04Sx
+WIND:3:Watchdog Running
+WIND:0:Console active
+WIND:46:WPA Crunching PSK:mypassword:10
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join:02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'ciscosb2' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up:192.168.1.106
```



Proceed to the next LAB!

Lab 4.3 : STA/MINI AP switcher

- Objective
  - Hardware “Station to Mini AccessPoint” switcher
- Prerequisites
  - Work alone



Lab 4.3 : STA/MINI AP switcher

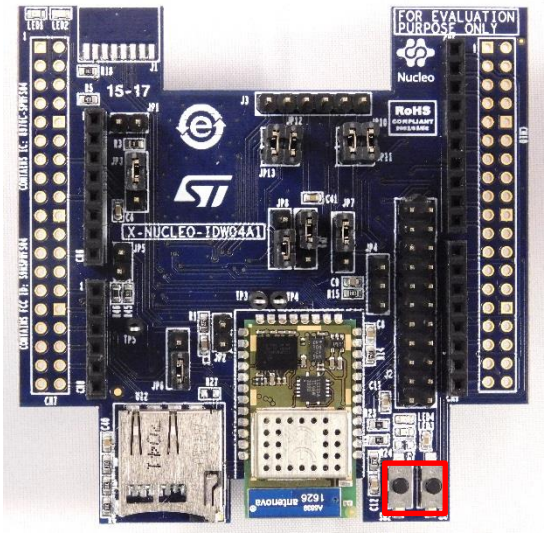
This feature allows to force the module in Mini AP mode starting from a preexistent state.

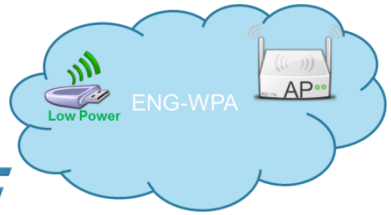
“Recovery Mode”: this functionality could be useful to lead the module in a known state and to reconfigure it (i.e. using the firstset page).

The GPIO7 will be used to drive this feature.

Lab 4.3 : STA/MINI AP switcher

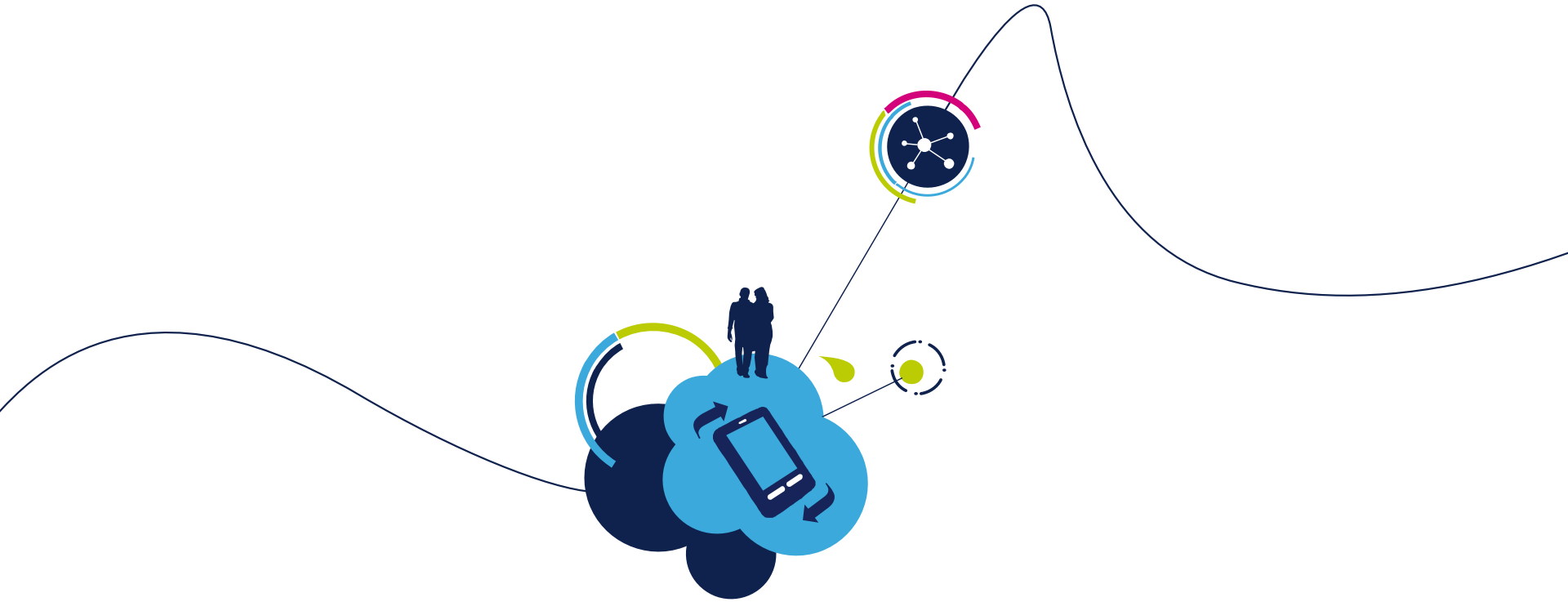
- Press and hold the SW1 button on the EVAL to perform a reset and click the SW2 button
- Release the SW1 button
- The MiniAP mode will be started and the module is discoverable with the following SSID:
  - i**wm-XX-YY-ZZ** where XX-YY-ZZ are the last six digits of module's MAC ADDRESS





Tera Term output

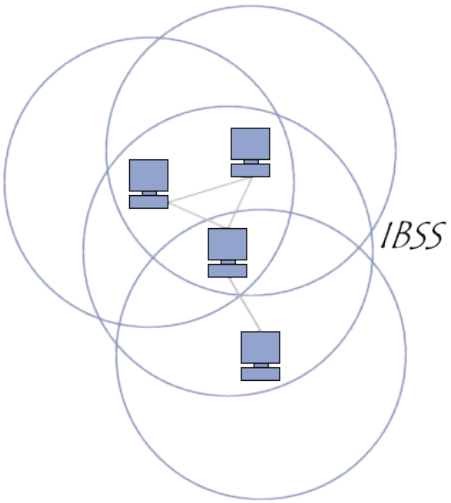
```
+WIND:1:Poweron:xxxxxxx-yyyyyyy-SPWF04S
+WIND:13:Copyright (c) 2012-2017 STMicroelectronics, Inc.  All rights Reserved:SPWF04Sx
+WIND:39:HW in miniAP mode
+WIND:0:Console active
+WIND:3:Watchdog Running
+WIND:32:WiFi Hardware Started
+WIND:26: Started AP: iwm-XX-YY-ZZ
+WIND:24:WiFi Up:192.168.0.1
```



Proceed to the next LAB!

Lab 4.4 : IBSS connection

- Objective
  - Create an IBSS network
- Prerequisites
  - Work alone



Lab 4.4 : Create an IBSS network

In an IBSS network, the SSID is chosen by the client device that starts the network.

Module settings to create an IBSS network:

Type **AT+S.WIFI=0**

- Set the IBSS SSID0
  - Type **AT+S.SSIDTXT=ADHOC**
- Set the network privacy mode (0=OPEN or 1=WEP are supported)
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (2 = IBSS)
  - Type **AT+S.SCFG=wifi_mode,2**

Tera Term output

```
AT+S.WIFI=0
AT-S.OK
AT+S.SSIDTXT=ADHOC
AT-S.OK
AT+S.SCFG=wifi_priv_mode,0
AT-S.OK
AT+S.SCFG=wifi_mode,2
AT-S.OK
```

Lab 4.4 : Create an IBSS network

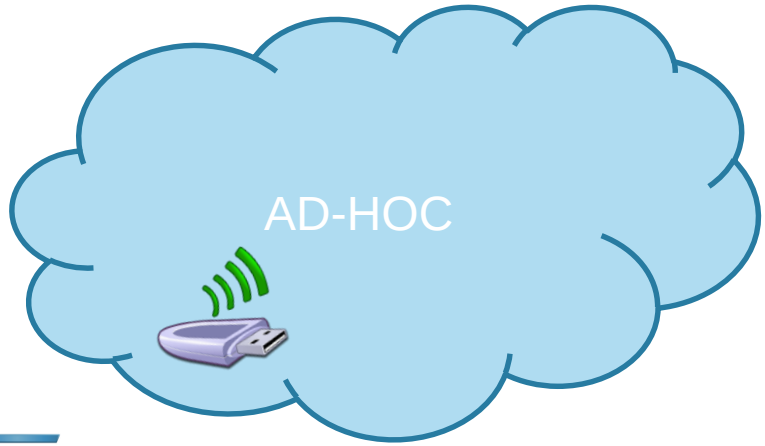
- Set IP address, IP default gateway, IP DNS and IP netmask
  - Type **AT+S.SCFG=ip_ipaddr,192.168.y.1xx**
  - Type **AT+S.SCFG=ip_gw,192.168.y.1**
  - Type **AT+S.SCFG=ip_dns1,192.168.y.1**
  - Type **AT+S.SCFG=ip_netmask,255.255.255.0**
- Turn off the DHCP
  - Type **AT+S.SCFG=ip_use_dhcpc,0**

Tera Term output

```
AT+S.SCFG=ip_ipaddr,192.168.0.1
AT-S.OK
AT+S.SCFG=ip_gw,192.168.0.1
AT-S.OK
AT+S.SCFG=ip_dns1,192.168.0.1
AT-S.OK
AT+S.SCFG=ip_netmask,255.255.255.0
AT-S.OK
AT+S.SCFG=ip_use_dhcp,0
AT-S.OK
```

Lab 4.4 : Create an IBSS network

- Save the settings on the flash memory and reset the module
  - Type **AT+S.WIFI=1**



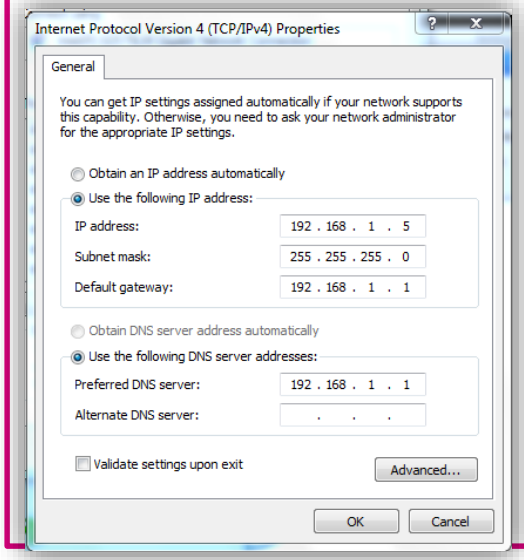
Tera Term output

```
AT-S.Var:ip_ipaddr=192.168.0.1
AT-S.Var:ip_netmask=255.255.255.0
AT-S.Var:ip_gw=192.168.0.1
AT-S.Var:ip_dns1=192.168.0.1
```

Lab 4.4 : Create an IBSS network

- Associate the device with the ADHOC network (iOS > 8 could not support the IBSS mode)

Tip: manual configuration of static TCP/IPv4 parameters is suggested using a PC
I.e. PC TCP/IPv4 properties



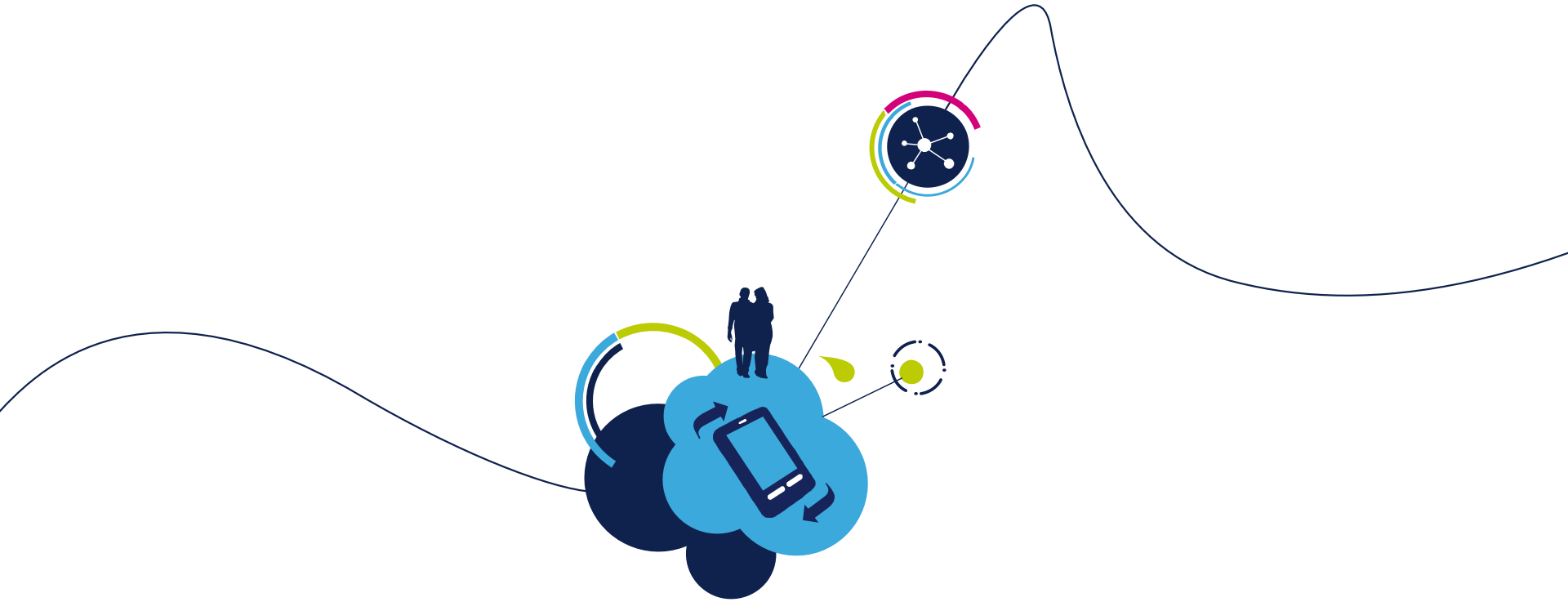
life.augmented

Lab 4.4 : Create an IBSS network

- Find your IP address
  - Type **AT+S.STS**

```
AT-S.Var:ip_ipaddr=192.168.1.2
AT-S.Var:ip_netmask=255.255.255.0
AT-S.Var:ip_gw=192.168.1.1
AT-S.Var:ip_dns1=192.168.1.1
AT-S.Var:ip_dns2=0.0.0.0
```

- Open Safari web browser
- In the address bar, type <SPWF IP address>/index.html
  - Type 192.168.1.3/index.html



Proceed to the next LAB!

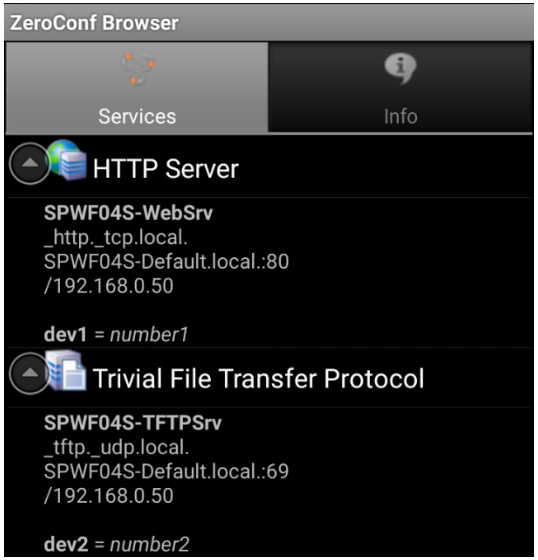
Lab 5 : mDNS

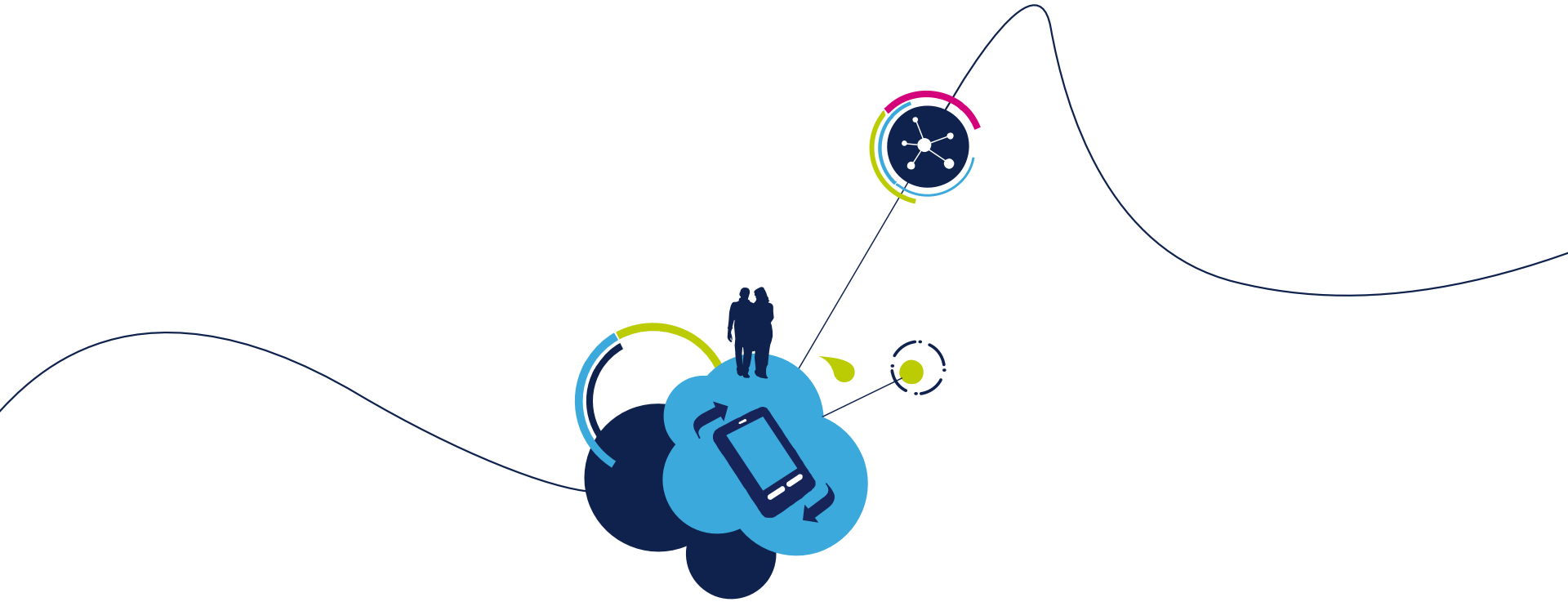
- Objective
  - Successfully see the module and see its capabilities via the mDNS protocol
- Prerequisites
  - Work alone

Lab 5 : mDNS

- Press and hold the SW1 button on the EVAL to perform a reset and click the SW2 button
- Release the SW1 button. You are, now, in MiniAP mode.
- Download and open ZeroConf Browser app on your smartphone, activate wifi and notice this :

Thanks to mDNS support you are able to discover service capabilities



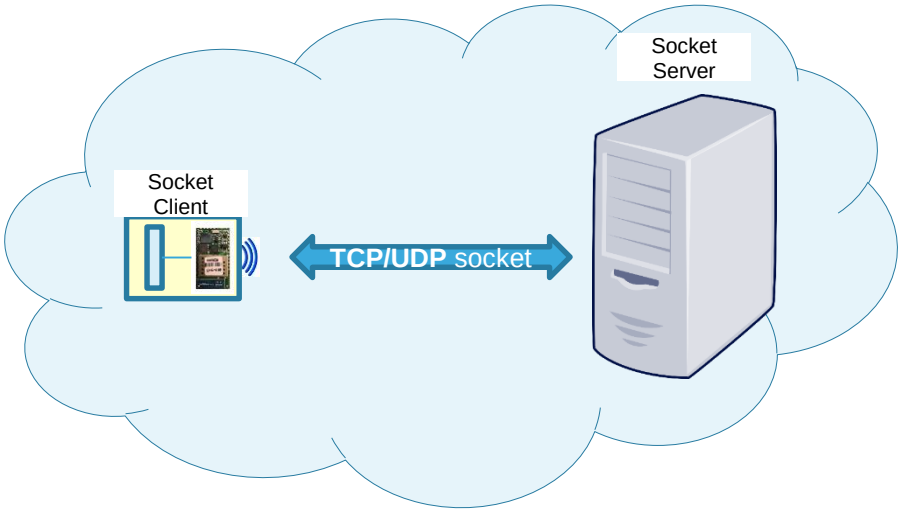


Proceed to the next LAB!

Lab 6 : Socket interface

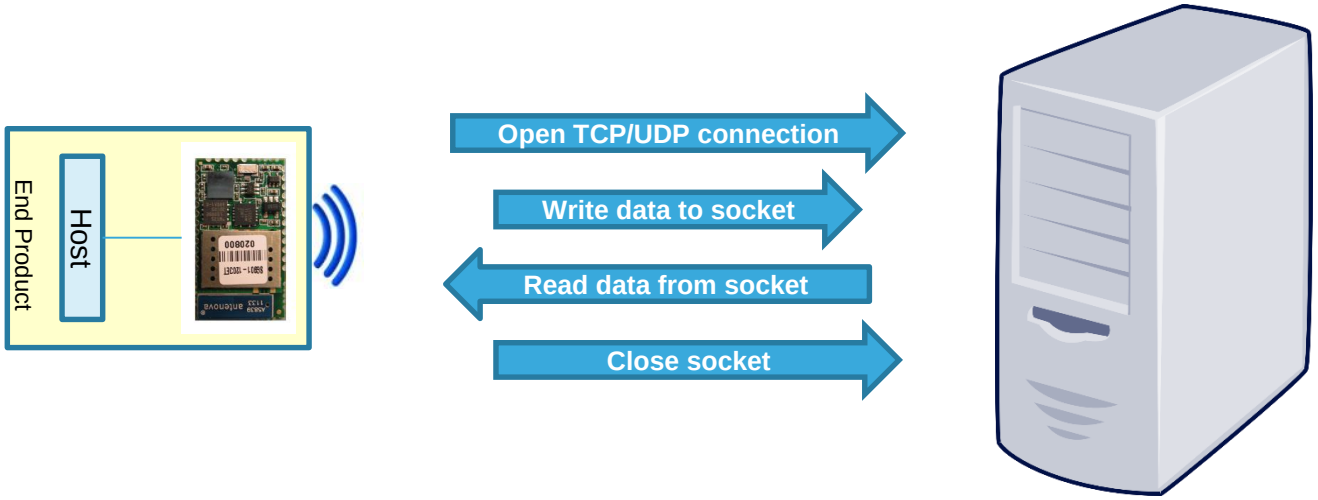
- Objective
  - Open TCP/UDP connection
  - Write data to socket
  - Read data from socket
  - Query socket
  - List socket
  - Close socket
  - Broadcast

- Prerequisites
  - Work alone



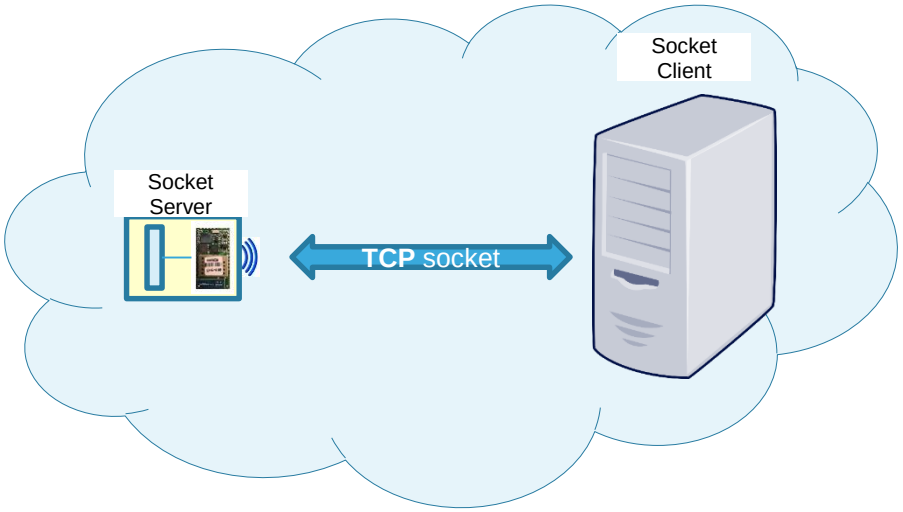
Lab 6 : TCP/UDP/Secure socket interface

The Socket interface allows communication via TCP, UDP or secure connection. The SPWF04 can be both a client and a server socket. The SPWF can be in whatever mode (MiniAP or Station Mode). For secure socket, please refer to Security_on_SPWF04S Application Notes.



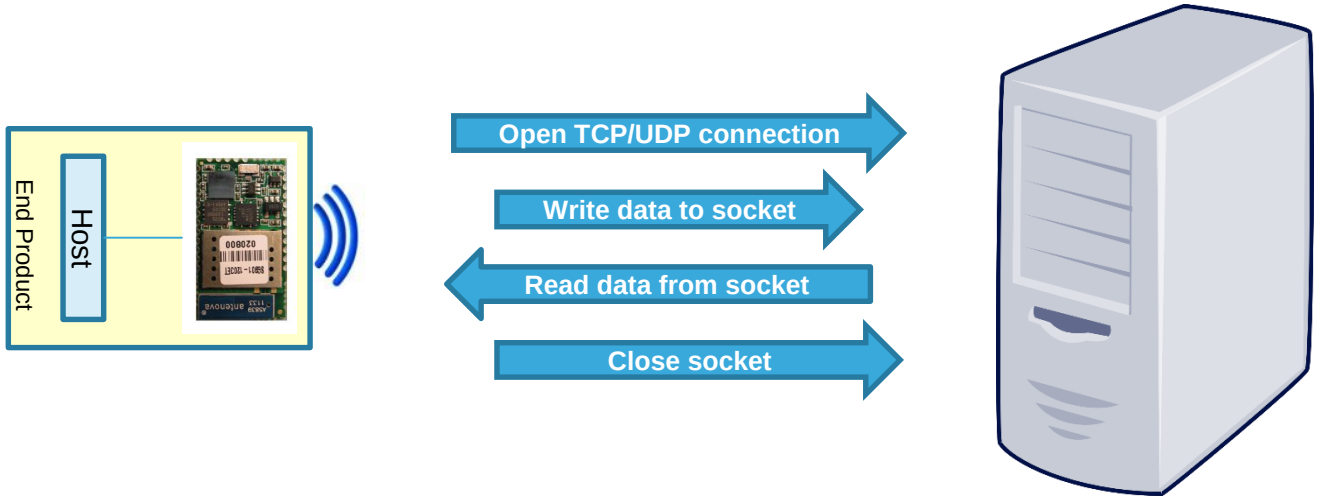
Lab 6.1 : Socket Client

- Objective
  - Open a TCP or UDP connection from the module to a socket server
- Prerequisites
  - module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode
  - PC to be used as socket server



Lab 6.1 : Socket Client

The Socket interface allows communication via TCP, UDP or secure connection. The SPWF is both a client and a server socket. In this LAB, will detail the socket client feature.



Lab 6.1 : Open Socket Client connection

The SOCKON command allows to open a TCP/UDP connection to a specified host (up to 8 socket connections at same time).

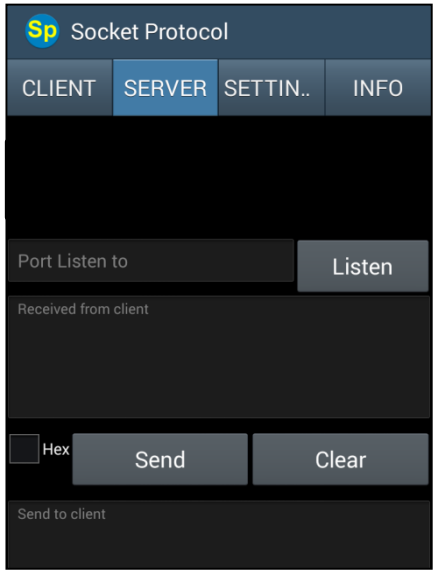
- Syntax
  - AT+S.SOCKON=<hostname>,<port>,,<kind><cr>
- Configuration parameters
  - <hostname> Remote Server. DNS resolvable name or IP address.
  - <port> TCP/UDP socket port
  - <kind> This parameter can assume the values:
    - t->tcp
    - u->udp
    - s->TLS socket using <Hostname> as domain name

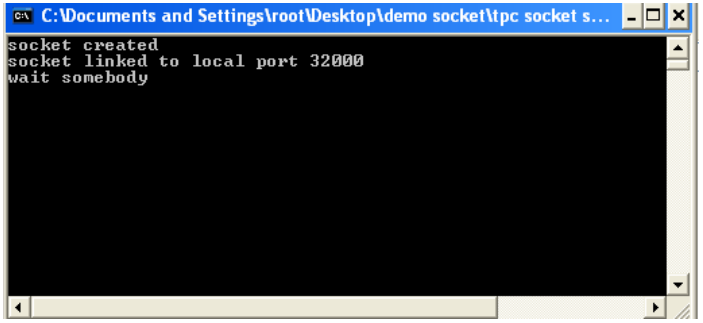
      TLS Server Domain Name: Common Name of the server (URL or the CN field reported into server certificate) for TLS socket

Lab 6.1 : Open Socket Client connection

- Open the TCP socket server (disable the firewall to properly run it)
  - Folder ../hands_on_wifi/tcp socket server/server.exe

- The TCP server listens for incoming connections on the port 32000. It sends back all data received.

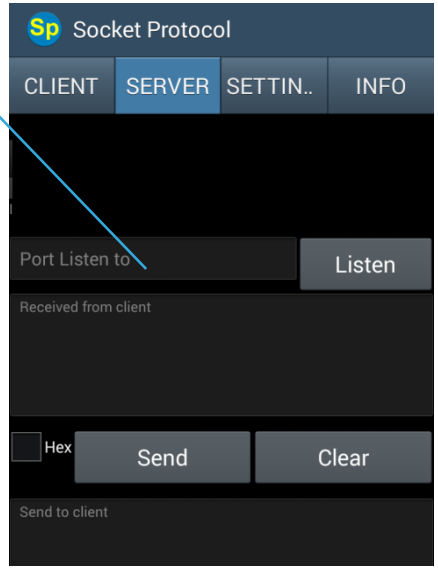




Lab 6.1 : Open Socket Client connection

- Using an Android platform, the socket server can be opened using a specific APP (Socket Protocol, available on Play Store)

- The TCP socket server can be configured to listen for incoming connections on the port 32000.
  - Type 32000 on «Port Listen to» box
  - Click on Listen

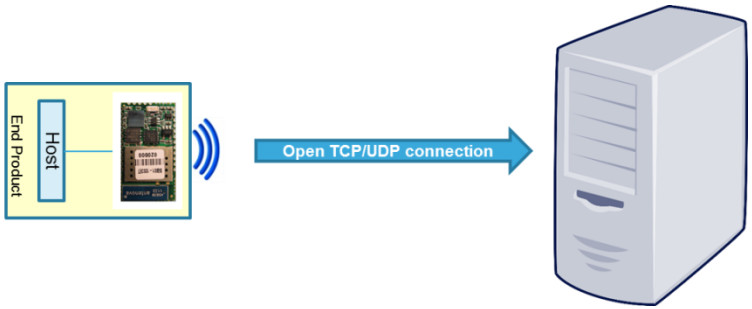


Lab 6.1 : Open Socket Client connection

- Type **AT+S.SOCKON=<hostname>,32000,,t**

  Hostname could be DNS resolvable name or IP

- The client and the server use the socket identifier (ID) displayed



Tera Term output

```
AT+S.SOCKON=<hostname>,32000,,t
AT-S.On:hostIPadress:0
AT-S.OK
```

Lab 6.1 : To query a socket client for length of pending data

The SOCKQ command allows to read data from socket.

- Syntax
  - AT+S.SOCKQ=<ID>
- Configuration parameters
  - <ID>: socket identifier

Tera Term output

AT+S.SOCKQ=<ID>
AT-S.Query:Lengthofdata
AT-S.OK

Lab 6.1 : Write data to socket

The SOCKW command allows to write data to the specified ID socket.

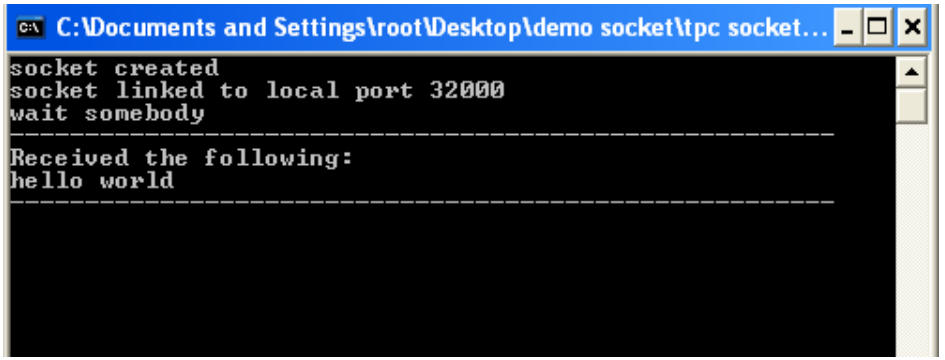
This command accepts data after the <cr> at the end of the command line.

- Syntax
  - AT+S.SOCKW=<ID>,<len>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: data length to send

Lab 6.1 : Write data to socket

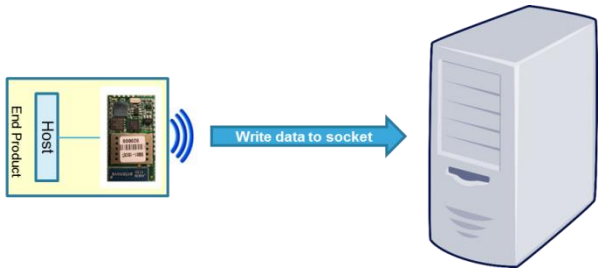
- Write data
  - Type **AT+S.SOCKW=0,11**

Note: the module is waiting 13 bytes to be written to the socket. As soon as 13 bytes (or characters) are sent from the terminal, the module is going to write them to the socket and will be ready to receive new commands.

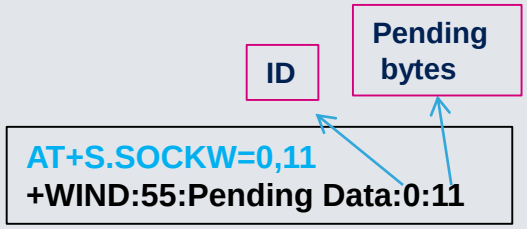


  - Type **hello world<CR>**

- SPWF shows that there are <pending data> and their amount



Tera Term output



life.augmented

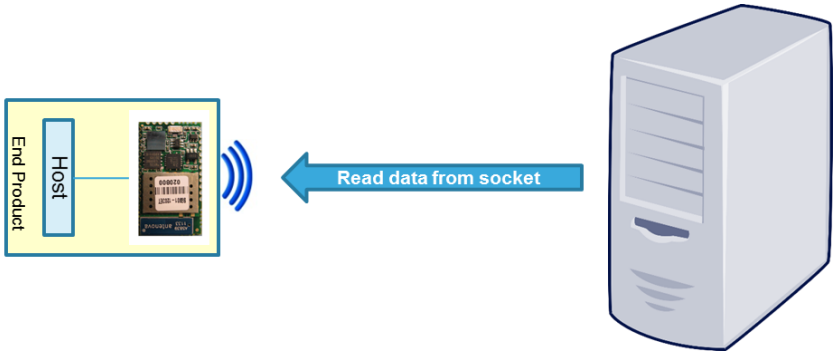
Lab 6.1 : Read data from socket

The SOCKR command allows to read data from socket.

- Syntax
  - AT+S.SOCKR=<ID>,<len>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: Defaut:0. Length (in bytes) of the buffer to read. The value 0 indicates to read the full buffer

Lab 6.1 : Read data from socket

- Read data
  - Type **AT+S.SOCKR=0,**



Tera Term output

AT+S.SOCKR=0,
hello world
AT-S.OK

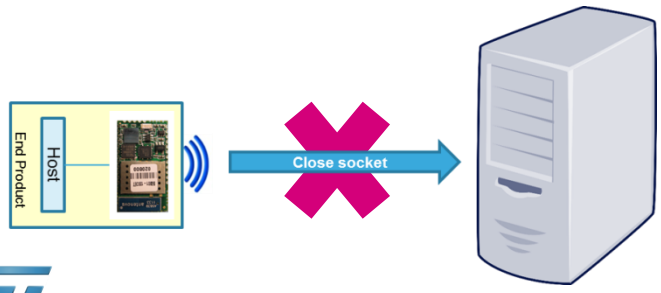
Lab 6.1 : Close socket

The SOCKC command allows to close socket.

- Syntax
  - AT+S.SOCKC=<ID>
- Configuration parameters
  - <ID>: socket identifier

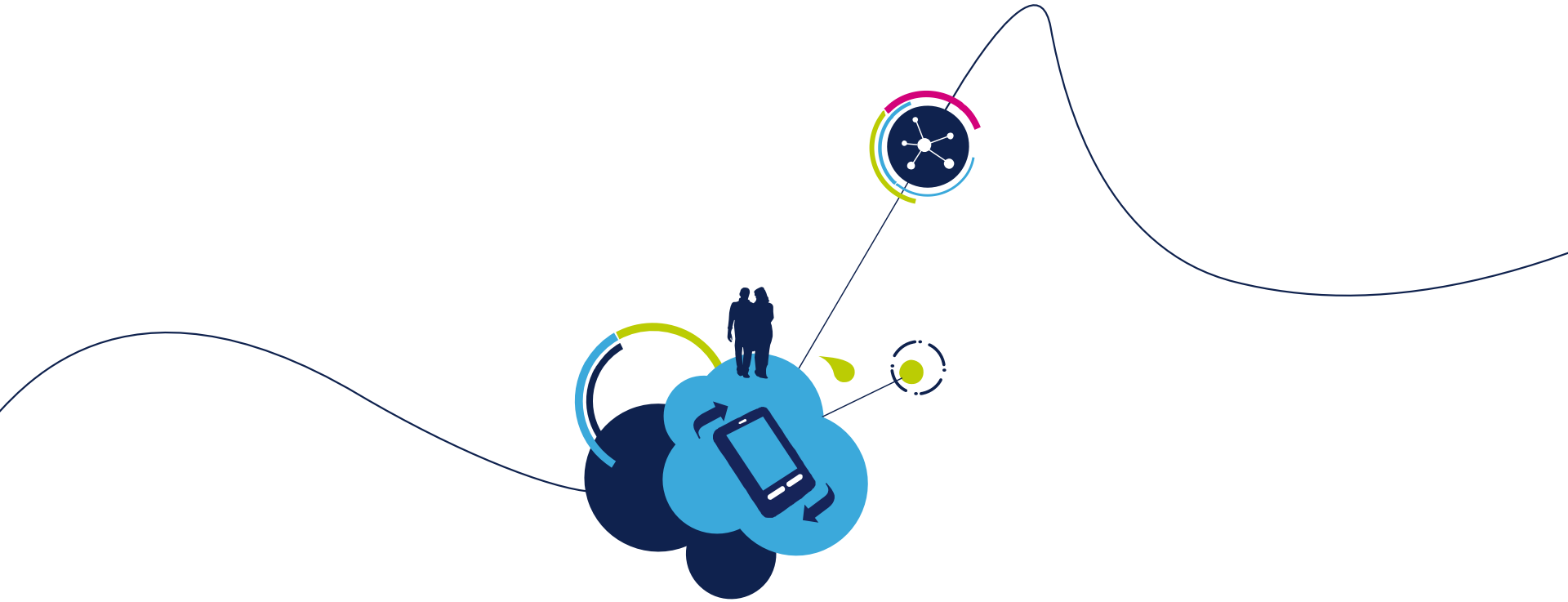
Lab 6.1 : Close socket

- Close socket
  - Type **AT+S.SOCKC=0**



Tera Term output

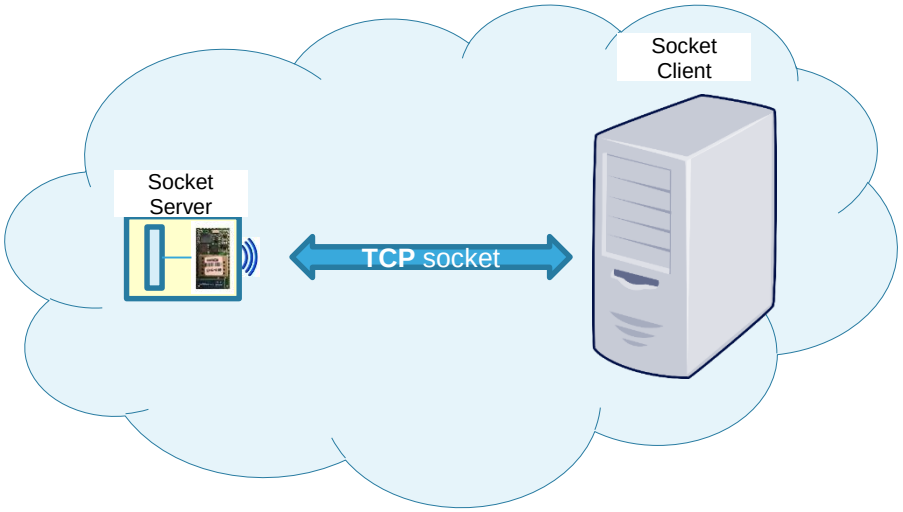
```
AT+S.SOCKC=0
AT-S.OK
```



Proceed to the next LAB!

Lab 6.2 : Socket Server functionality

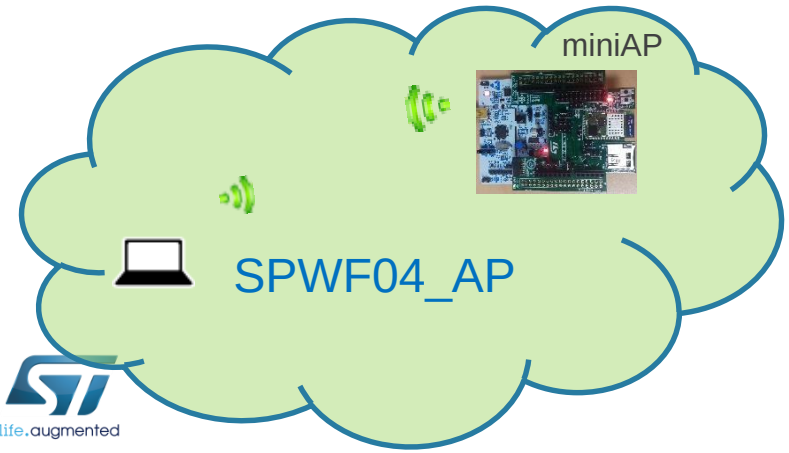
- Objective
  - Configure a Server Listening Port
  - Open a TCP or UDP connection from a socket client to the module
- Prerequisites
  - module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode
  - PC to be used as socket client

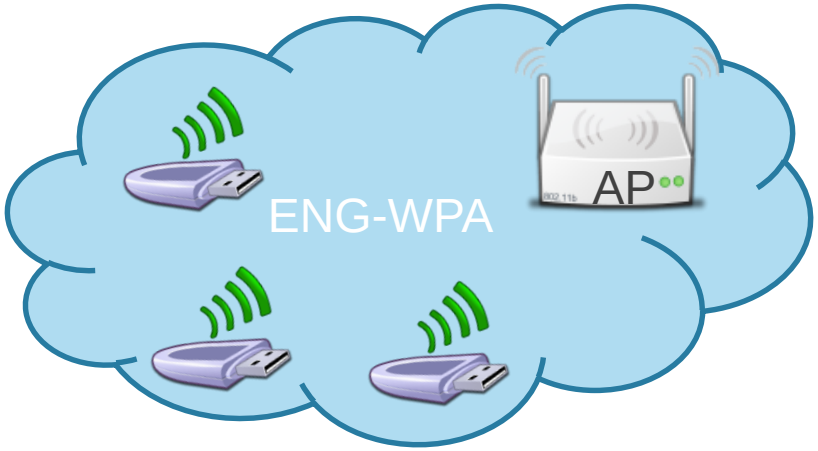


Lab 6.2 : Socket Server

This feature allows to enable the socket server mode. The module can be able to listen for an incoming connection on the specified port.

The module should be connected to the AP or should be configured in Mini AP mode. We will detailed the socket server in station mode.

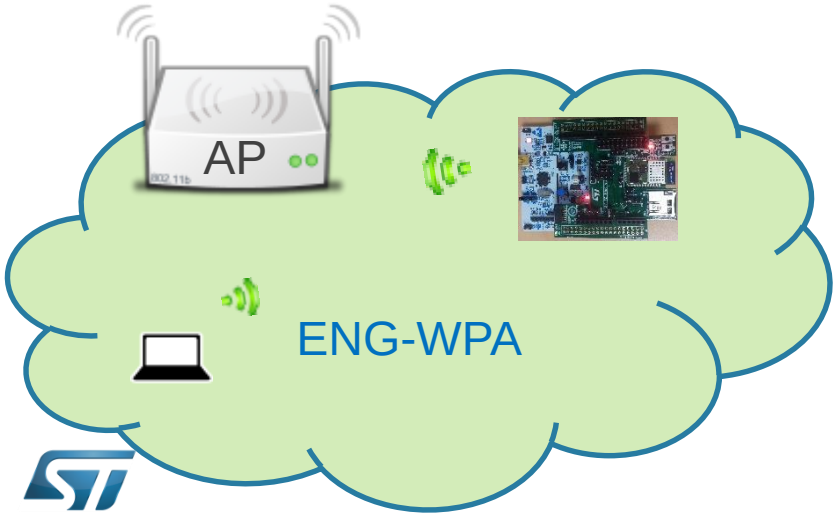




Lab 6.2 : Socket Server

- Socket server: turn ON the TCP Socket Server (user must specify the server listening port)
  - Type **AT+S.SOCKDON=32000,t**

(**AT+SOCKDON=32000,u** for UDP socket server or **AT+SOCKDON=32000,s1** for one-way and **AT+SOCKDON=32000,s2** for mutual secure socket server)

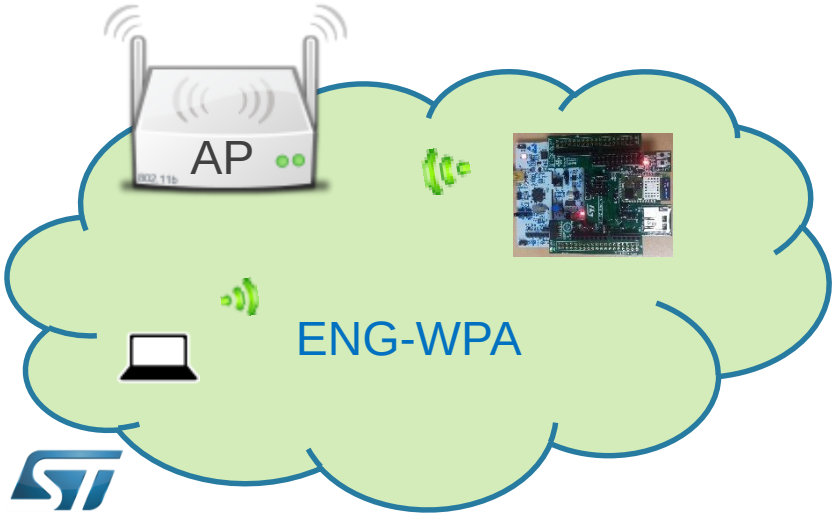


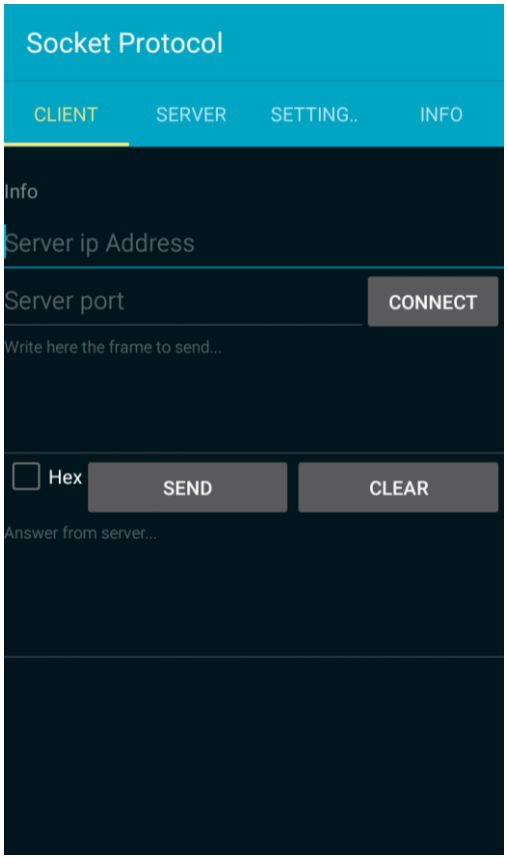
Tera Term output

```
AT+S.SOCKDON=32000,t
AT-S.On:0
AT-S.OK
```

Lab 6.2 : Socket Server

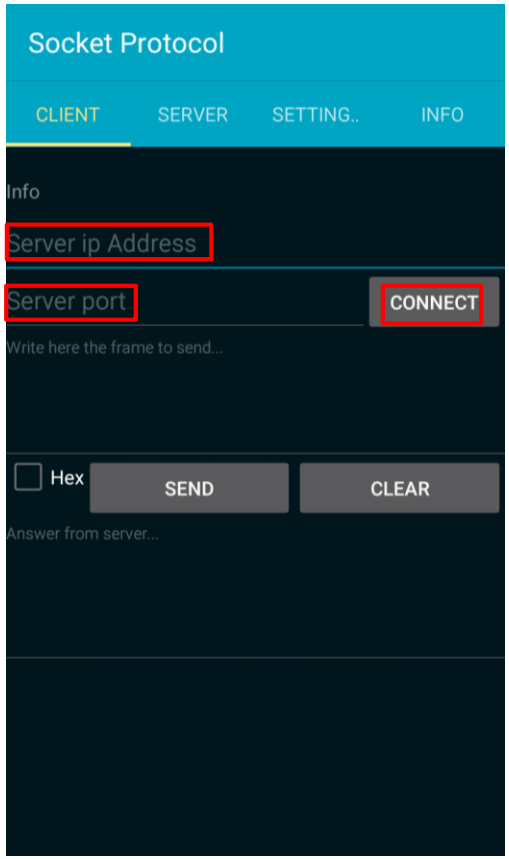
- Socket client: can be used a simple socket client in order to test the communication (Socket Protocol App – available on Play Store)





Lab 6.2 : Socket Server

- Socket client:
  - Insert the **module's IP Address** and the **port**
  - Click on the **Connect** button

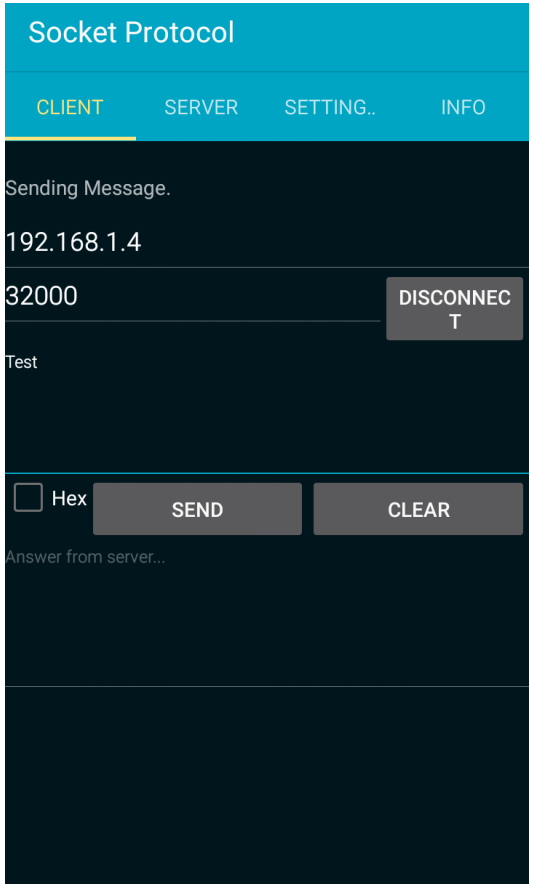


Tera Term output

+WIND:61:Incoming Socket Client:192.168.1.XX:12345:0:0

Lab 6.2 : Socket Server

- Try to send and receive data from the module



Tera Term output

```
+WIND:55:Pending
Data:0:0:4:4
```

Lab 6.2 : List bound socket client

The SOCKDL command allows to list bound socket client.

- Syntax
  - **AT+S.SOCKDL**
  - **AT+S.SOCKDL=**<sid><cr> to list bound clients on the specified server

- Configuration parameters
  - <sid> socket server identifier

Lab 6.2 : Write data to a socket client

The SOCKDW command allows to read data from socket client.

- Syntax
  - **AT+S.SOCKDW=**<sid>,<cid>,<len><cr><data>
- Configuration parameters
  - <sid> socket server identifier
  - <cid> socket client identifier
  - <len>: Length (in bytes) of the buffer to write that is sent after the command.

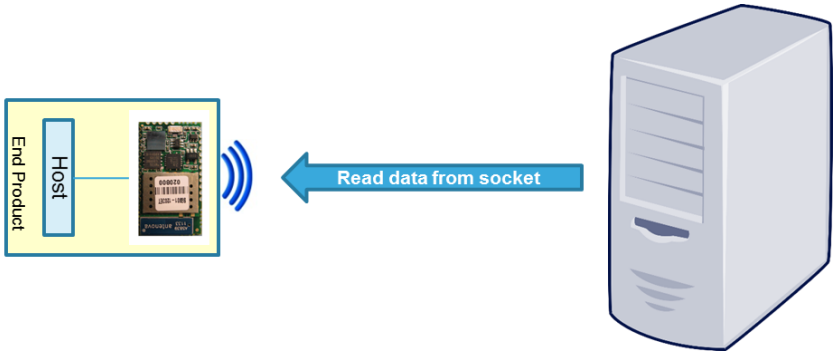
Lab 6.2 : Read data from socket

The SOCKDR command allows to read data from socket client.

- Syntax
  - AT+S.SOCKDR=<sid>,<cid>,<len><cr>
- Configuration parameters
  - <sid> socket server identifier
  - <cid> socket client identifier
  - <len>: Defaut:0. Length (in bytes) of the buffer to read. The value 0 indicates to read the entire buffer

Lab 6.2 : Read data from socket

- Read data
  - Type **AT+S.SOCKDR=0,0,**



Tera Term output

AT+S.SOCKDR=0,0,
AT-S.Reading:5:5
Test
AT-S.OK

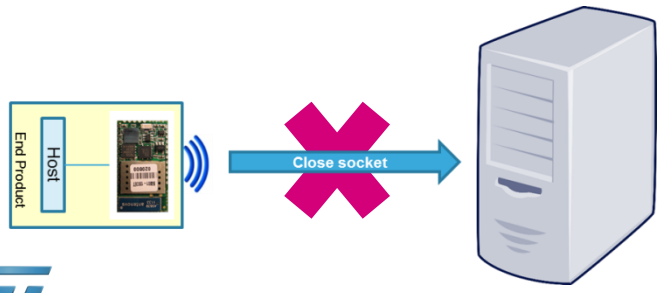
Lab 6.2 : Close socket

- Syntax
  - AT+S.SOCKDC=<sid> Disconnect all clients, and turn off the server
  - AT+S.SOCKDC=<sid>,<cid> Disconnect specific client, and keep the port open

- Configuration parameters
  - <sid> socket server identifier
  - <cid> socket client identifier

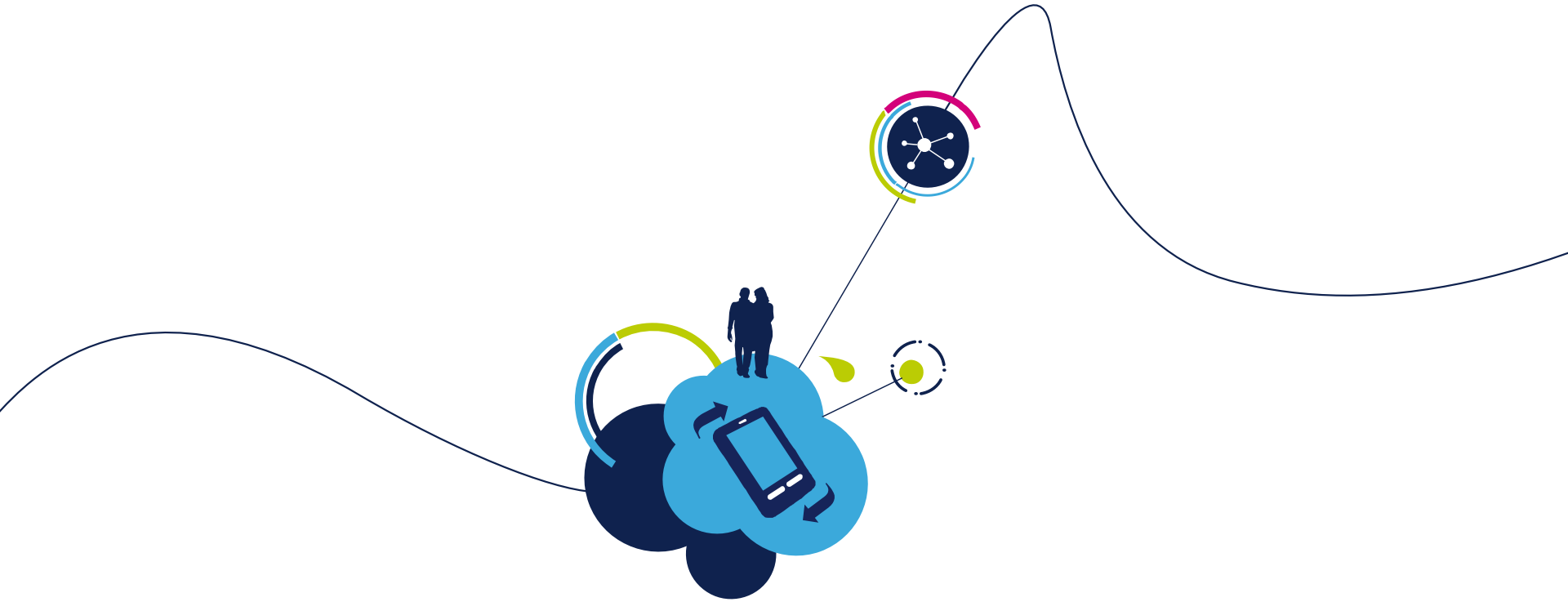
Lab 6.2 : Close socket

- Close socket
  - Type **AT+S.SOCKDC=0**



Tera Term output

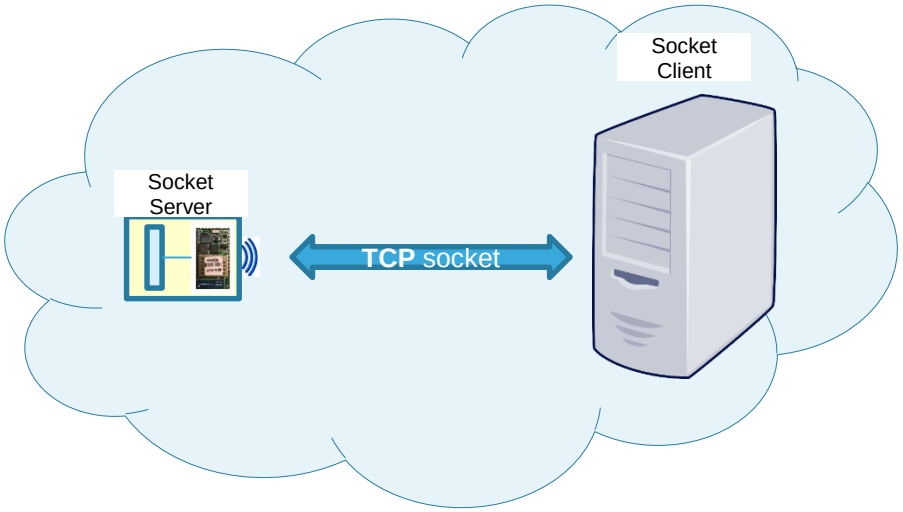
AT+S.SOCKDC=0
AT-S.OK



Proceed to the next LAB!

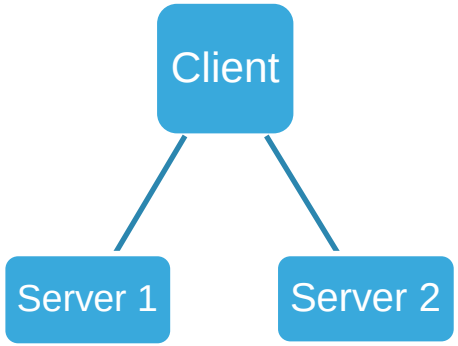
Lab 6.3 : Broadcast examples

- Objective
  - Broadcast with socket server configuration
  - Broadcast between socket client only
- Prerequisites
  - module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode
  - PC to be used as socket client



Lab 6.3 : First Case, one client to 2 servers

- Open socket client
  - Type **AT+S.SOCKON=192.168.1.255,32000,,u**
- If other modules are also used as socket server, open socket servers (up to 8 socket client connection are supported)
  - Type **AT+S.SOCKDON=32000,u**
- Write data on your socket client
  - Type **AT+S.SOCKW=0,5<CR>hello**



Tera Term output of socket client

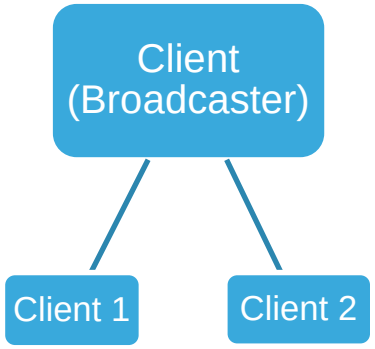
```
AT+S.SOCKON=192.168.1.255,32000,,u
AT-S.On:192.168.1.255:0
AT-S.OK
AT+S.SOCKW=0,5
AT-S.OK
```

Tera Term output of sockets server

```
AT+S.SOCKDON=32000,u
AT-S.On:0
AT-S.OK
+WIND:80:UDP Broadcast Received:192.168.1.9:5
hello
```

Lab 6.3 : Second Case, sockets client only

- Open a socket client (the «Broadcaster»)
  - Type **AT+S.SOCKON=192.168.1.255,32000,,u**
- Open other sockets client (the receivers)
  - Type **AT+S.SOCKON=192.168.1.***,32000,,u** (*** corresponding to your Broadcaster)
- Write data on the «Broadcaster»
  - Type **AT+S.SOCKW=0,5<CR>hello**



Tera Term output of Broadcaster

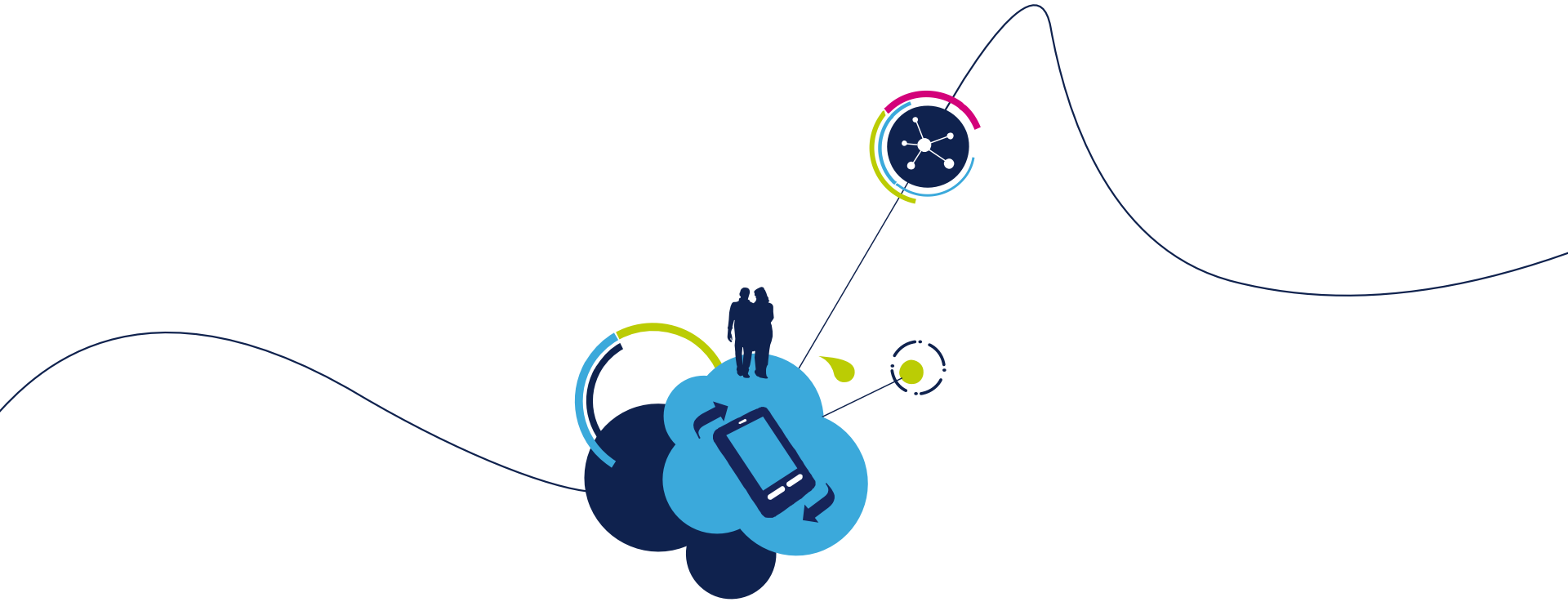
```
AT+S.SOCKON=192.168.1.255,32000,,u
AT-S.On:192.168.1.255:0
AT-S.OK
AT+S.SOCKW=0,5
AT-S.OK
```

Tera Term output of sockets server

```
AT+S.SOCKON=192.168.1.***,32000,,u
AT-S.On:0
AT-S.OK
+WIND:55:Pending Data::0:5:5
hello
```

Lab 6.3 : Broadcast as socket server

UDP broadcast as socket server not available at the moment, coming soon.



Proceed to the next LAB!

Lab 7 : Websocket

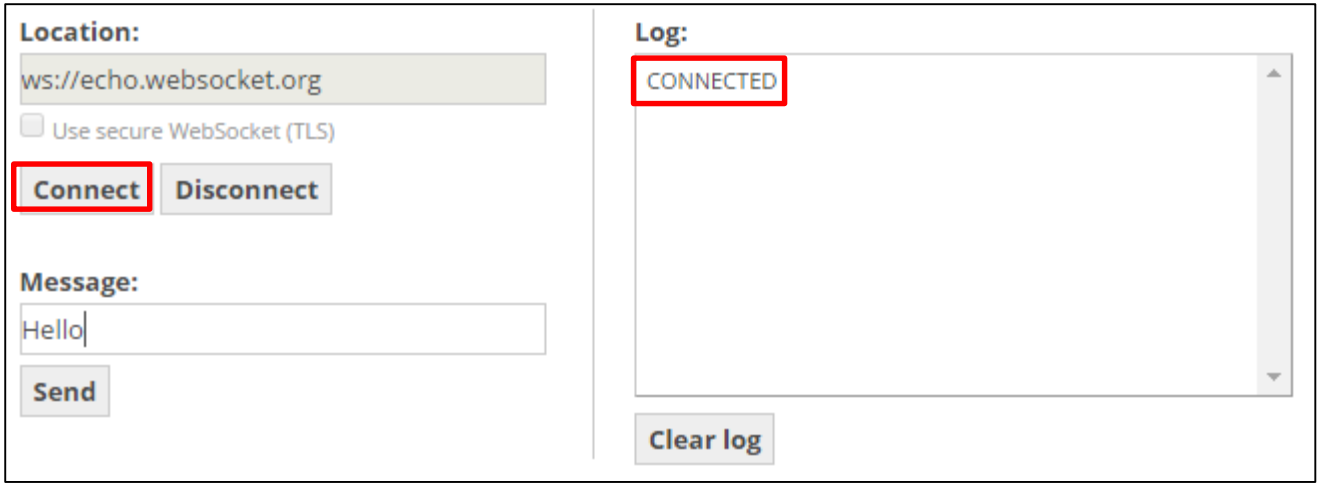
- Objective
  - Create a direct connection between the module and an end device
  - Write data to socket
  - Read data from socket
  - Query socket
  - List socket
  - Close socket

- Prerequisites
  - Work alone
  - Your module have to be connected to an Access point with an Internet Connection

Lab 7 : Websocket

To begin, you need to use a Websocket Server, for example you can use this one : http://www.Websocket.org/echo.html

- Click on Connect



Lab 7 : Open TCP/UDP connection

The WSOCKON command allows to open a TCP/UDP connection to a specified host.

- Syntax
  - AT+S.WSOCKON=<hostname>,[<port>],[<path>],[<TLS>],[<username>],[<passwd>],[<origin>],[<protocols>],[<extensions>]<cr>

- Configuration parameters
  - <hostname> DNS resolvable name or IP address of the Websocket server.
  - <port> Default 80 (if TLS=0) or 443 (if TLS>0).
  - <path> Default:/
  - <TLS> Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS
  - <username> Default: none. Username on the remote server
  - <passwd> Default: none. Passwd on the remote server
  - <origin> Default:none. Header Field Origin
  - <protocols> Default:none. Header Field Protocols
  - <extensions> Default:none. Header Field Extensions

Lab 7 : Websocket

So, with the previous example you can type :

AT+S.WSOCKON=echo.Websocket.org,80,,,,,,,<CR>

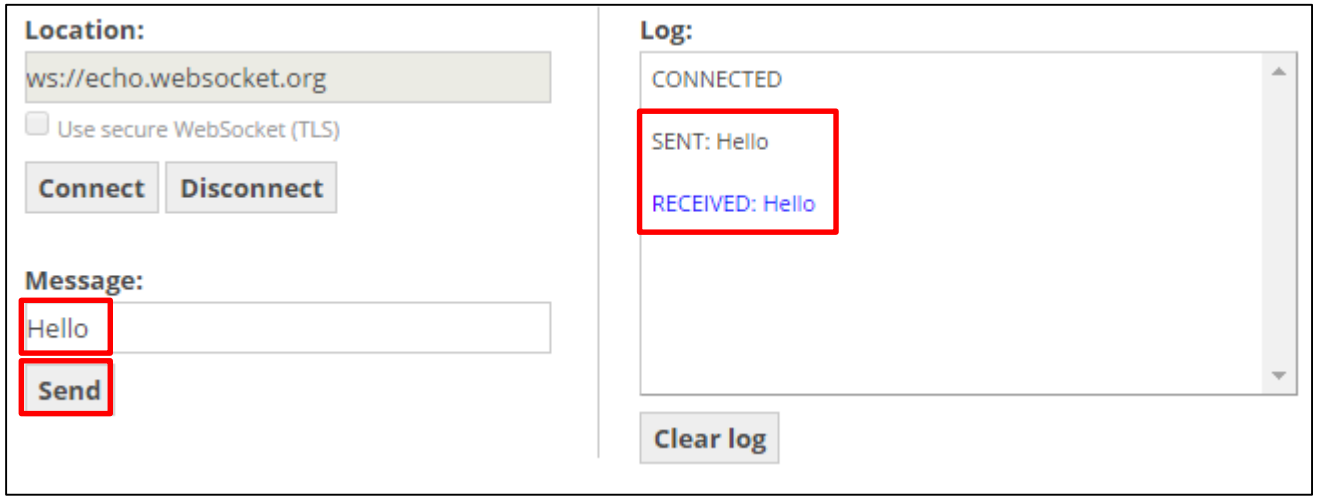
Tera Term output

AT+S.WSOCKON=echo.Websocket.org,80,,,,,,,
AT-S.On:0
AT-S.OK

Lab 7 : Websocket

Now you can send a message, in the example, the server returns the message sent :

- Click the «Send» button



Lab 7 : Write data to Websocket

Now, let’s do that with the SPWF04 module

The WSOCKW command allows to write data to the specified ID socket.

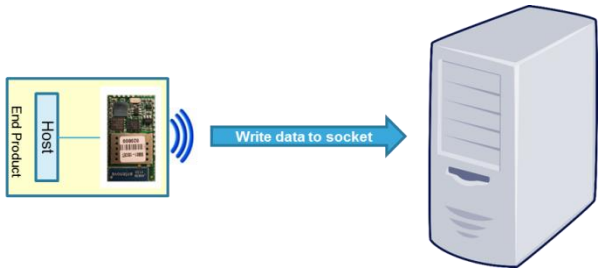
This command accepts data after the <cr> at the end of the command line.

- Syntax
  - AT+S.WSOCKW=<id>,[<lastFrame>],[<lastFrag>],[<binary>],<len><cr> {data}
- Configuration parameters
  - <id> Websocket client identifier
  - <lastFrame> Default:0 . 1-> Last frame flag.
  - <lastFrag> Default:0. 1-> Last frag flag.
  - <binary> Default:0, textual. 1-> Binary Flag

Lab 7 : Write data to Websocket

- Write data
  - Type **AT+S.WSOCKW=**0,1,1,0,6<CR>hello

- SPWF shows that there are <pending data> and their amount



Tera Term output

ID

Pending bytes

AT+S.WSOCKW=0,1,1,0,6
AT-S.OK
+WIND:88:Websocket Data:0:0:1:6:6

Lab 7 : To query a Websocket client for length of pending data

The WSOCKQ command allows to read data from socket.

- Syntax
  - AT+S.WSOCKQ=<ID><CR>
- Configuration parameters
  - <ID>: socket identifier

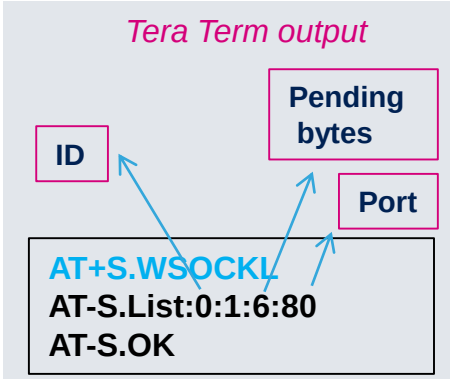
Tera Term output

AT+S.WSOCKQ=<ID>
AT-S.Query:Lengthofdata
AT-S.OK

Lab 7 : List Websocket

The WSOCKL command allows to list opened Websocket clients.

- Syntax
  - AT+S.WSOCKL<cr>



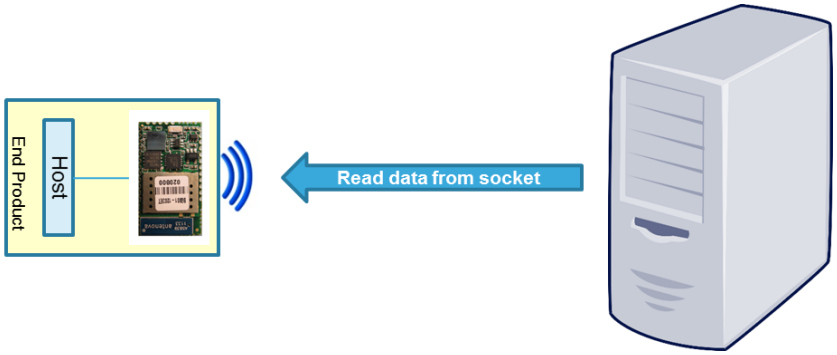
Lab 7 : Read data from Websocket

The WSOCKR command allows to read data from socket.

- Syntax
  - AT+S.WSOCKR=<ID>,[<len>]<cr>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: Defaut:0. Length (in bytes) of the buffer to read. The value 0 indicates to read the full buffer

Lab 7 : Read data from Websocket

- Read data
  - Type **AT+S.WSOCKR=0,**



Tera Term output

AT+S.WSOCKR=0,
AT-S.Reading:5:5
helloAT-S.OK

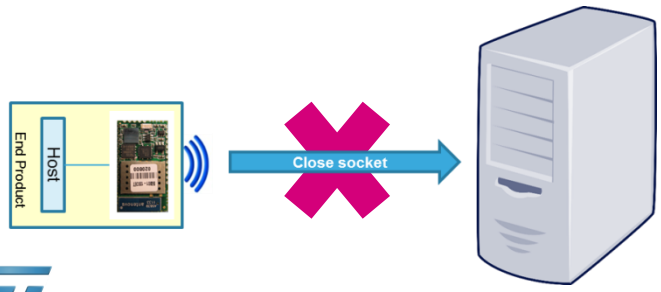
Lab 7 : Close Websocket

The WSOCKC command allows to close Websocket.

- Syntax
  - AT+S.WSOCKC=<id>,[<status>]<cr> Configuration parameters
    - <id> Websocket client identifier
    - <status> Default:0; 0--> Normal Closure; 1-> Going Away; For a complete list of the status values defined for the Websocket refer to the related standard.

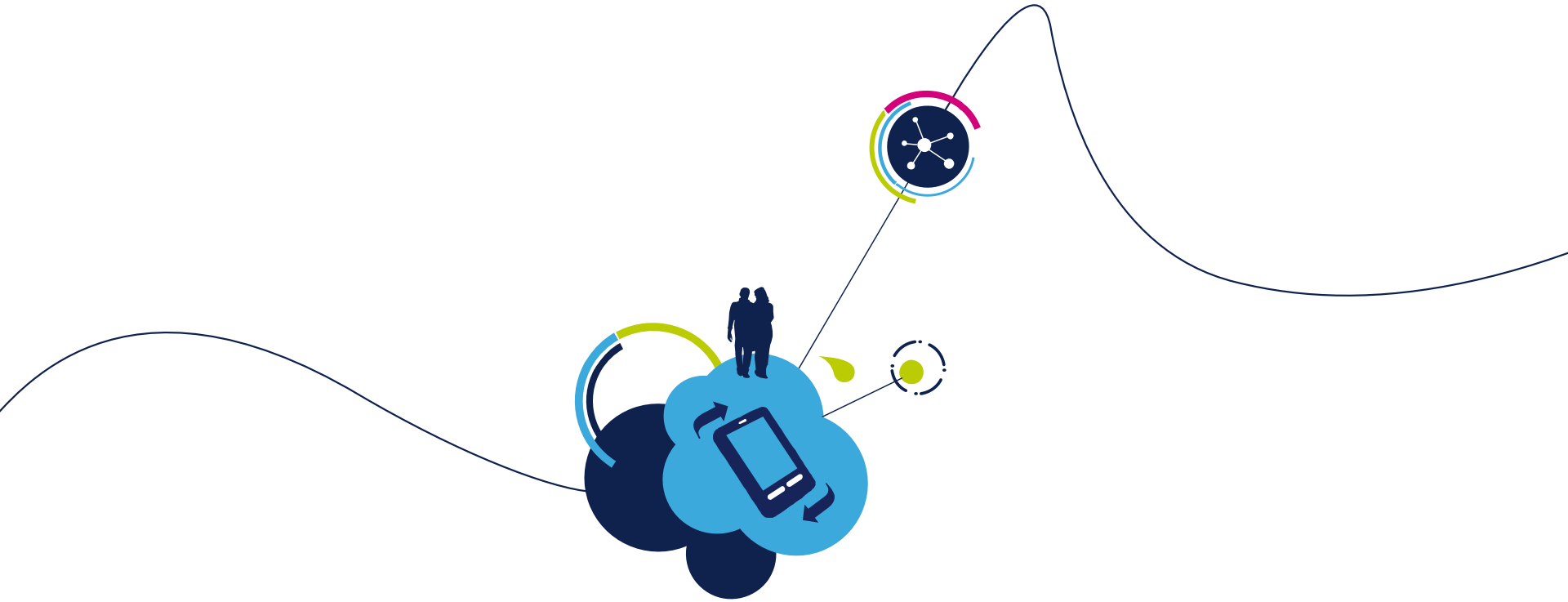
Lab 7 : Close Websocket

- Close Websocket
  - Type **AT+S.WSOCKC=0,**



Tera Term output

AT+S.WSOCKC=0,
AT-S.OK



Proceed to the next LAB!

Lab 8 : HTTP Web Interface

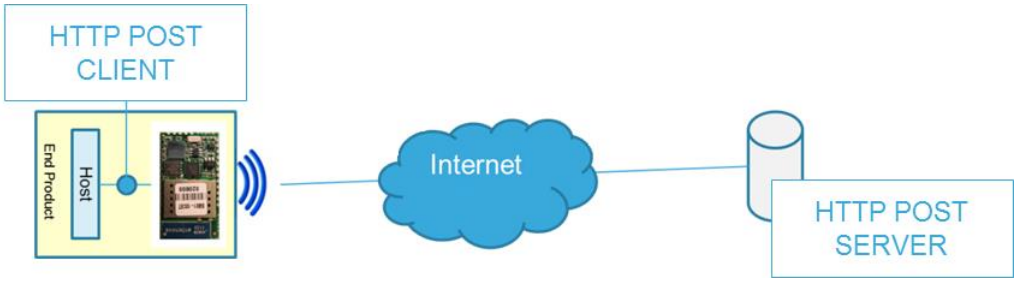
- Objective
  - Discover module capabilities acting as web client or web server
- Prerequisites
  - USB dongle and computer are set up as described in Lab 2
  - Work in couple

Lab 8.1 : Web client mode

- Objective
  - HTTP GET
  - HTTP POST
- Prerequisites
  - USB dongle and computer are set up as described in Lab 2
  - Work in couple

Lab 8.1 : HTTP POST

To perform a post of the specified file to a remote host.



- Syntax
  - AT+S.HTTPPOST=<hostname>,[<Path&queryopts]>,[<port>],[<TLS Kind>],[<username>],[<passwd>],[<In Filename>],[<Out Filename>]<cr>
- Configuration parameters
  - <Hostname> DNS resolvable Name or IP address
  - <Path&queryopts> Default:/index.html. document path & optional query arguments.
  - <port> Default 80 (if TLS=0) or 443 (if TLS>0).
  - <TLS> Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS

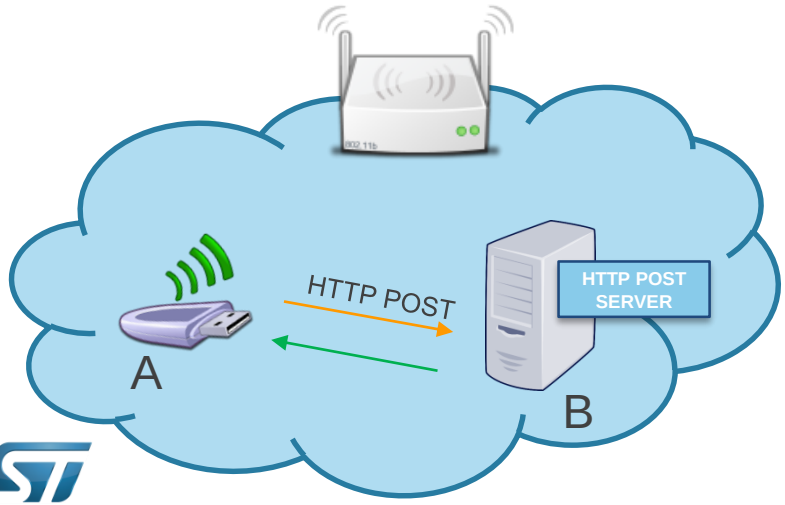
Lab 8.1 : HTTP POST

- Configuration parameters
  - <username> Default: none.
  - <passwd> Default: none.
  - <In Filename> Default: none. Console. When specified the return data are saved in a file.
  - <Out Filename> Default: none. Filename to transfer to the server .

Lab 8.1 : HTTP POST

- The SPWF04S performs an HTTP POST to an HTTP Post Test Server
  - Type: **AT+S.HTTPPOST=posttestserver.com,/post.php,,,,,,**

- The HTTP Post Test Server replies as displayed in the Tera Term output if the HTTP POST successfully



Tera Term output

```
AT+S.HTTPPOST=posttestserver.com,/post.php,,,,,,
Successfully  dumped X post variables.
View it at
http://www.posttestserver.com/data/2017/02/10/07.26
.53614711XX
Post body was X chars long.
AT-S.OK
```

Lab 8.1 : HTTP GET

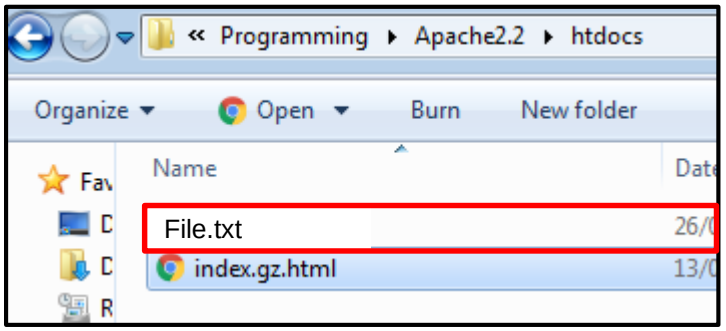
The **Apache Web Server** will be used in this tutorial

(Apache Web Server is available at this link:

http://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.25-win32-x86-openssl-0.9.8y.msi)

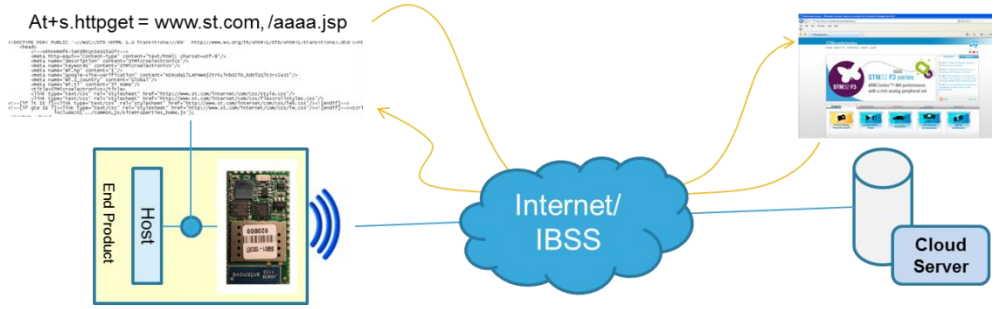
Note: please check that the local firewall is disabled or properly set. It can block the connection from module to Apache.

- Copy the file that you want to get with the http command in the Apache 2.2 *htdocs folder* (this is the default root server directory)



Lab 8.1 : HTTP GET

The HTTP GET feature performs a single HTTP request to the specified host and path. The server response is printed on the UART enabled.



- Syntax

- AT+S.HTTPGET=<hostname>,[<Path&queryopts]>,[<port>],[<TLSKind>],[<username>],[<passwd>],[<In Filename>],[<Out Filename>]<cr>

- Configuration parameters
  - <Hostname> DNS resolvable Name or IP address
  - <Path&queryopts> Default:/index.html. document path & optional query arguments.
  - <port> Default 80 (if TLS=0) or 443 (if TLS>0).

Lab 8.1 : HTTP GET

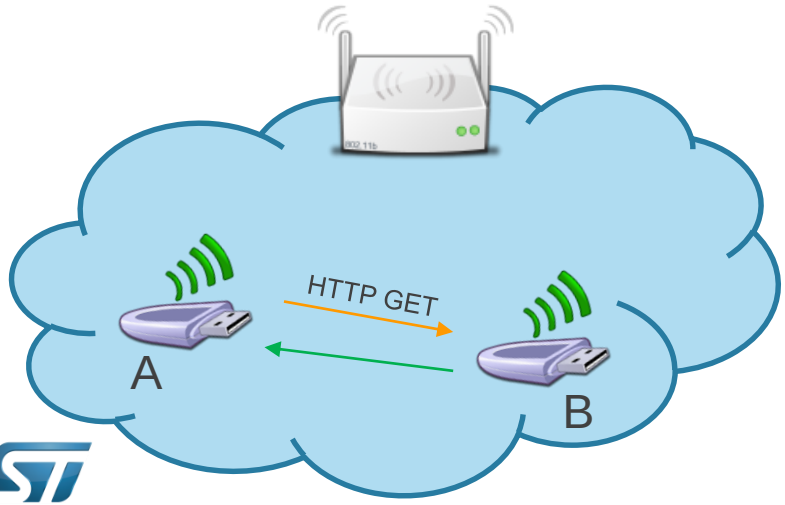
- Configuration parameters
  - <TLS> Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS
  - <username> Default: none.
  - <passwd> Default: none.
  - <In Filename> Default: none. Custom http requests
  - <Out Filename> Default: Console. When specified the return data are saved in a file.

Lab 8.1 : HTTP GET

- Device A performs an HTTP GET to the Device B
  - Device A:

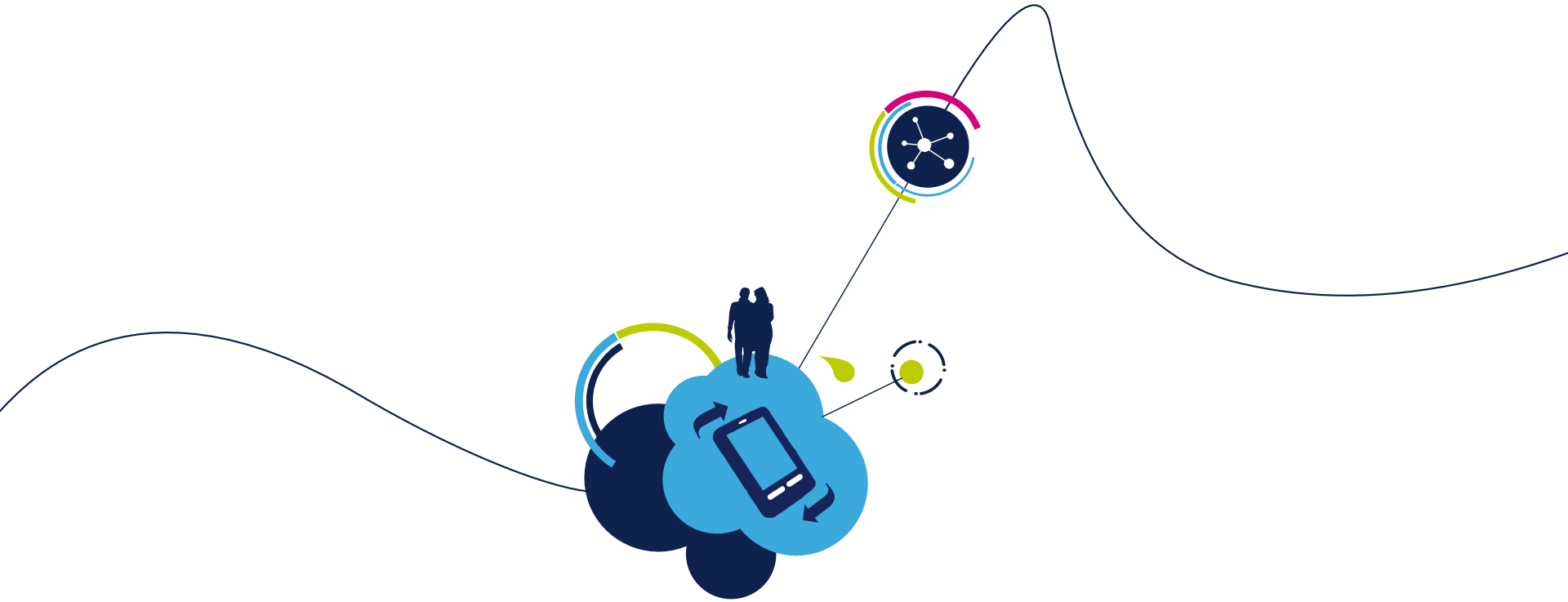
    AT+S.HTTPGET=<Device B IP addr>,/File.txt,,,,,

    Type **AT+S.HTTPGET=**192.168.1.3,/File.txt,,,,,,



Tera Term output

AT+S.HTTPGET=192.168.1.3,/File.txt,,,,,,
ContentofyourfileAT-S.OK



Proceed to the next LAB!

Lab 8.2 : Web Server mode

- Web Server Feature
- Web Server Usage

Lab 8.2.1 : Web Server Feature

- Objective
  - List, Print existing files
  - Create a file, Delete an existing file in RAM
  - Create a file in Flash
  - Create Dynamic Pages
- Prerequisites
  - Work alone

Lab 8.2.1.1 : List, Print existing files

- Objective
  - See the list of files on the server
  - See the contain of a file
- Prerequisites
  - Work in couple

Lab 8.2.1.1 : List existing files

The FSL command lists type, sizes and name of all the existing files.

- Type **AT+S.FSL**

Files stored in the APP Disk

Tera Term output

```
AT+S.FSL
AT-S.Free RAM Disk:14848
AT-S.File:D    484    ScanResult
AT-S.File:I    4241   config.fhtml
AT-S.File:I    676    favicon.gz.ico
AT-S.File:I    697    firstset.gz.html
AT-S.File:I    401    index.gz.html
AT-S.File:I    252    input_demo.fhtml
AT-S.File:I    658    MULTI_CLIENT_SERVER.py
AT-S.File:I    290    output_demo.gz.html
AT-S.File:I    1719   peers.fhtml
AT-S.File:I    882    remote.gz.html
AT-S.File:I    2318   RL_TCP_CL.py
AT-S.File:I    2340   RL_TCP_SE.py
AT-S.File:I    2696   RL_TCP_SE_GC_COLLECT.py
AT-S.File:I    2317   RL_UDP_CL.py
AT-S.File:I    2331   RL_UDP_SE.py
AT-S.File:I    1768   status.fhtml
AT-S.File:I    4134   stlogo.gz.jpg
AT-S.File:I    897    WLAN.py
AT-S.File:I    1246   WLAN.STA.py
AT-S.OK
```

Lab 8.2.1.1 : Print a file

The FSP command prints the content of an existing file.

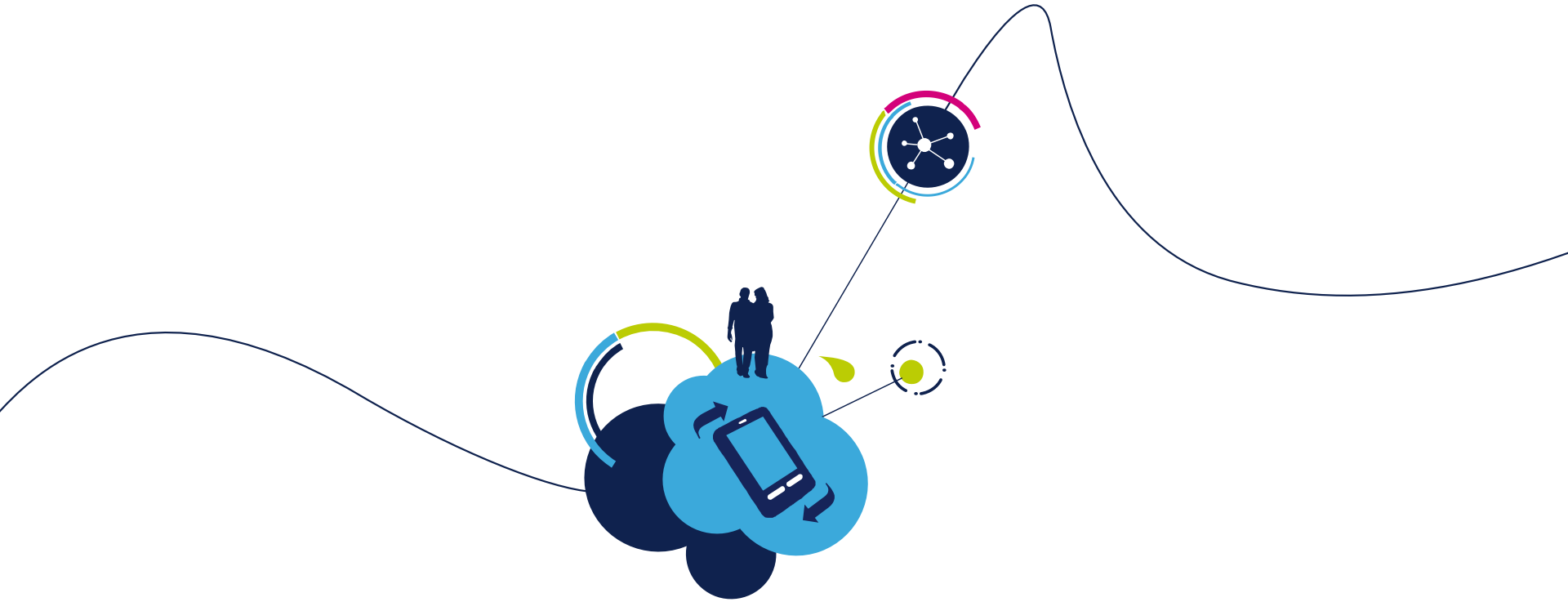
- Type **AT+S.FSP=/input_demo.fhtml,,**

Tip: How to use the offset and length parameters:
AT+S.FSP=/index.html,[offset],[length]

i.e. ***Type***
AT+S.FSP=/input_demo.fhtml,5,20

Tera Term output

```
AT+S.FSP=/input_demo.fhtml,,
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML
4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

<html>

<head><title>Input Demo</title></head>

<body><fieldset>

<legend><h4>SPWF04S Input from
Host</h4></legend>

<!--|06|Input|0|-->

</fieldset>

</body>

</html>

AT-S.OK
```



Proceed to the next LAB!

Lab 8.2.1.2 : Create a file and Delete an existing file in RAM

- Objective
  - Create a file in RAM
  - Delete a file in RAM
- Prerequisites
  - Work in couple

Lab 8.2.1.2 : Create a file in RAM

- The FSC command allows to create a file inside the SPWF04 for delivery by the SPWF04 HTTP server, or appends the data following the command in case the file already exists.

- Syntax
  - **AT+S.FSC=**<filename>,<datalen><cr>{data}

- Configuration parameters
  - <filename> name of the file. Max size is 64 bytes
  - <datalen> amount of space in bytes to allocate for the file.

Type **AT+S.FSC=/wifidemo.html,1965**

Tera Term output

AT+S.FSC=/wifidemo.html,1965
AT-S.OK

Lab 8.2.1.2 : Create a file in RAM

Let's list the files

- Type **AT+S.FSL**

File stored in the RAM Disk

Tera Term output

```
AT+S.FSL
AT-S.Free RAM Disk:14848
AT-S.File:D    484    ScanResult
AT-S.File:I    4241   config.fhtml
AT-S.File:I    676    favicon.gz.ico
AT-S.File:I    697    firstset.gz.html
AT-S.File:I    401    index.gz.html
AT-S.File:I    252    input_demo.fhtml
AT-S.File:I    658    MULTI_CLIENT_SERVER.py
AT-S.File:I    290    output_demo.gz.html
AT-S.File:I    1719   peers.fhtml
AT-S.File:I    882    remote.gz.html
AT-S.File:I    2318   RL_TCP_CL.py
AT-S.File:I    2340   RL_TCP_SE.py
AT-S.File:I    2696   RL_TCP_SE_GC_COLLECT.py
AT-S.File:I    2317   RL_UDP_CL.py
AT-S.File:I    2331   RL_UDP_SE.py
AT-S.File:I    1768   status.fhtml
AT-S.File:I    4134   stlogo.gz.jpg
AT-S.File:I    897    WLAN.py
AT-S.File:I    1246   WLAN.STA.py
AT-S.OK
```

Lab 8.2.1.2

- Open wifidemo.html
  - Open your Web browser (suggested Google Chrome for HTML5 test)
  - In address bar, type <SPWF IP addr>/wifidemo.html

    Type **192.168.0.1xx/wifidemo.html**

Lab 8.2.1.2 : Delete an existing file

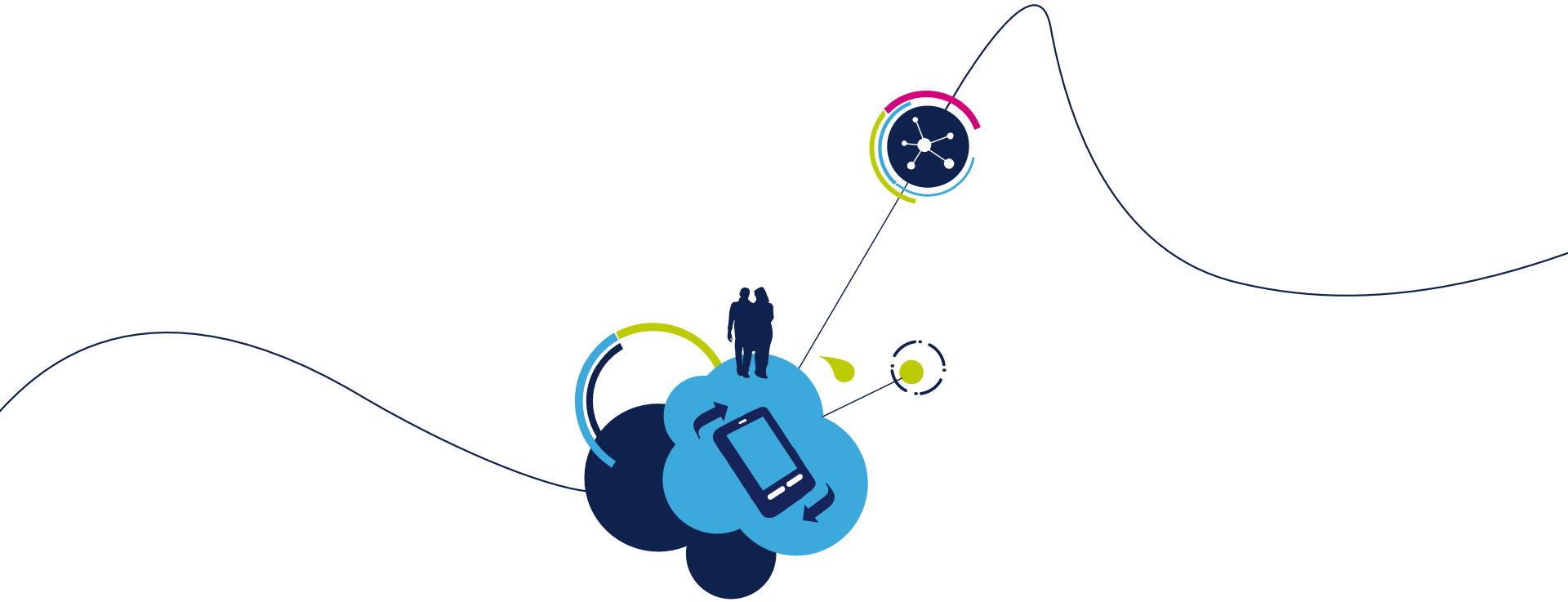
The FSD command allows to delete an existing file by name. Static files may not be deleted.

- Syntax
  - AT+S.FSD=/<filename>

Type **AT+S.FSD=/wifidemo.html**

Tera Term output

AT+S.FSD=/wifidemo.html
AT-S.OK



Proceed to the next LAB!

Lab 8.2.1.3 : Create a file in Flash

- Objective
  - Create a file in Flash
    - FS OTA
    - FS over UART
    - FS over SWD
- Prerequisites
  - Work in couple

Lab 8.2.1.3 : Create an image file

- Objective
  - Create an image file

- Prerequisites
  - Work alone

Lab 8.2.1.3 : Create an image file

- Open the following folder and put your file you want to add to the FS or remove one (960KB max for USER Flash, and 64Kb max for APP Disk) :

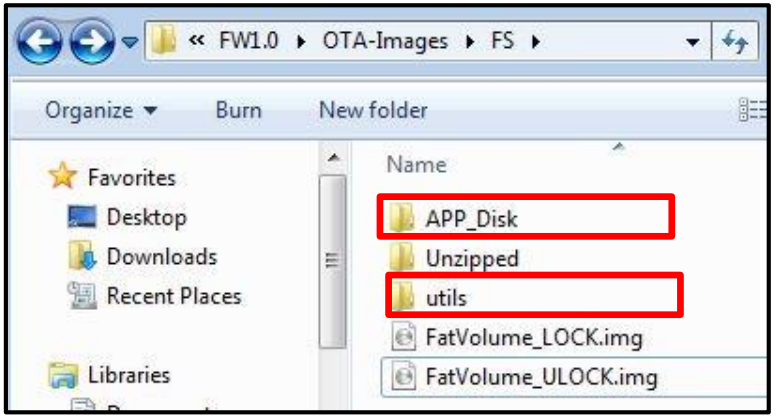
  OTA-Images/FS/APP_Disk

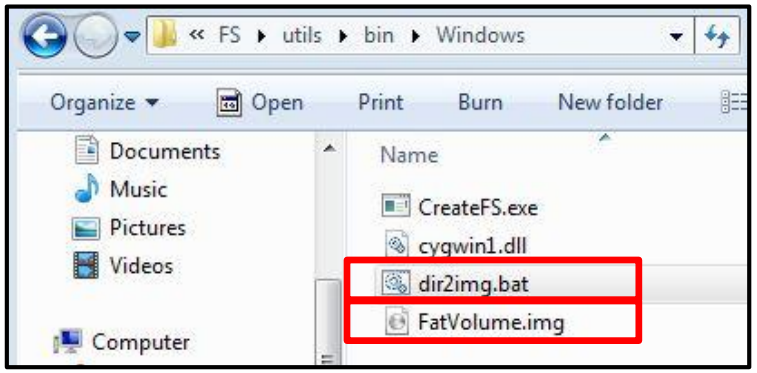
- After having modified as below, run dir2img.bat findable in OTA-Images/FS/utils/bin/Windows

Note bat file can be modified regarding
FS size or location of pages

%CONVERTER% 1024 "..\..\..\APP_Disk"

- You can see the new image





Lab 8.2.1.3.1 : Upgrade FS OTA

- Objective
  - Filesystem update Over-The-Air

- Prerequisites
  - Work alone

Lab 8.2.1.3.1 : Upgrade FS OTA

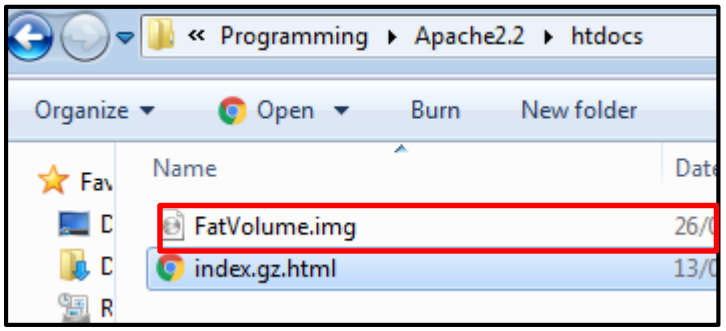
The **Apache Web Server** will be used in this tutorial

(Apache Web Server is available at this link:

http://archive.apache.org/dist/httpd/binaries/win32/httpd-2.2.25-win32-x86-openssl-0.9.8y.msi)

Note: please check that the local firewall is disabled or properly set. It can block the connection from module to Apache.

- Copy FatVolume.img in the Apache 2.2 *htdocs folder* (this is the default root server directory)



Lab 8.2.1.3.1 : Upgrade FS OTA

The FSUPDATE command allows to perform a FileSystem Over-the-air update via a single HTTP (or HTTPS) GET.

- Syntax
  - AT+S.FSUPDATE=<mem>,<hostname>,[<path&queryopts>],[<port>],[<TLS>],[<username>],[<passwd>]<cr>
- Configuration parameters
  - <mem> : specifies the memory where the fs is saved on. The character “e” indicates the user flash, the character ‘i’ indicates the application flash, and the character “x” indicates the external memory volume.
  - <hostname> : Target host. DNS resolvable name or IP address.
  - <path&queryopts> : Default:/fs.img. Document path and optional query arguments.
  - <port> : Default 80 (if TLS=0) or 443 (if TLS>0).
  - <TLS> : Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS
  - <username> : Default: none.
  - <passwd> : Default: none.

Lab 8.2.1.3.1 : Upgrade FS OTA

- In Tera Term: type **AT+S.FSUPDATE=i,192.168.1.129,FatVolume.img,80,0,,**

Warning

The file system will be permanently deleted.
The new IMG will overwrite the existent files in the file system

- Reset the module
  - Type **AT+S.RESET**

For memory mapping
see Lab 0 (File system organization)

Tera Term output

```
AT+S.FSUPDATE=i,192.168.1.129,FatVolume.img,80,0,,
AT-S.Write chunk:2048:8113800
AT-S.Write chunk:2048:8114000
AT-S.Write chunk:2048:8114800
AT-S.Write chunk:2048:8115000
AT-S.Write chunk:2048:8115800
AT-S.Write chunk:2048:8116000
AT-S.Write chunk:2048:8116800
……..
……..
……..
AT-S.Write chunk:2048:8117000
AT-S.Write chunk:2048:8117800
AT-S.Write chunk:2048:8118000
AT-S.Write chunk:2048:8118800
AT-S.Write chunk:2048:8119000
AT-S.Write chunk:2048:8119800
AT-S.Write chunk:2048:811A000
AT-S.Write chunk:2048:811A800
AT-S.Write chunk:2048:811B000
AT-S.OK
AT+S.RESET
+WIND:2:Reset
```

Lab 8.2.1.3.1 : Upgrade FS OTA

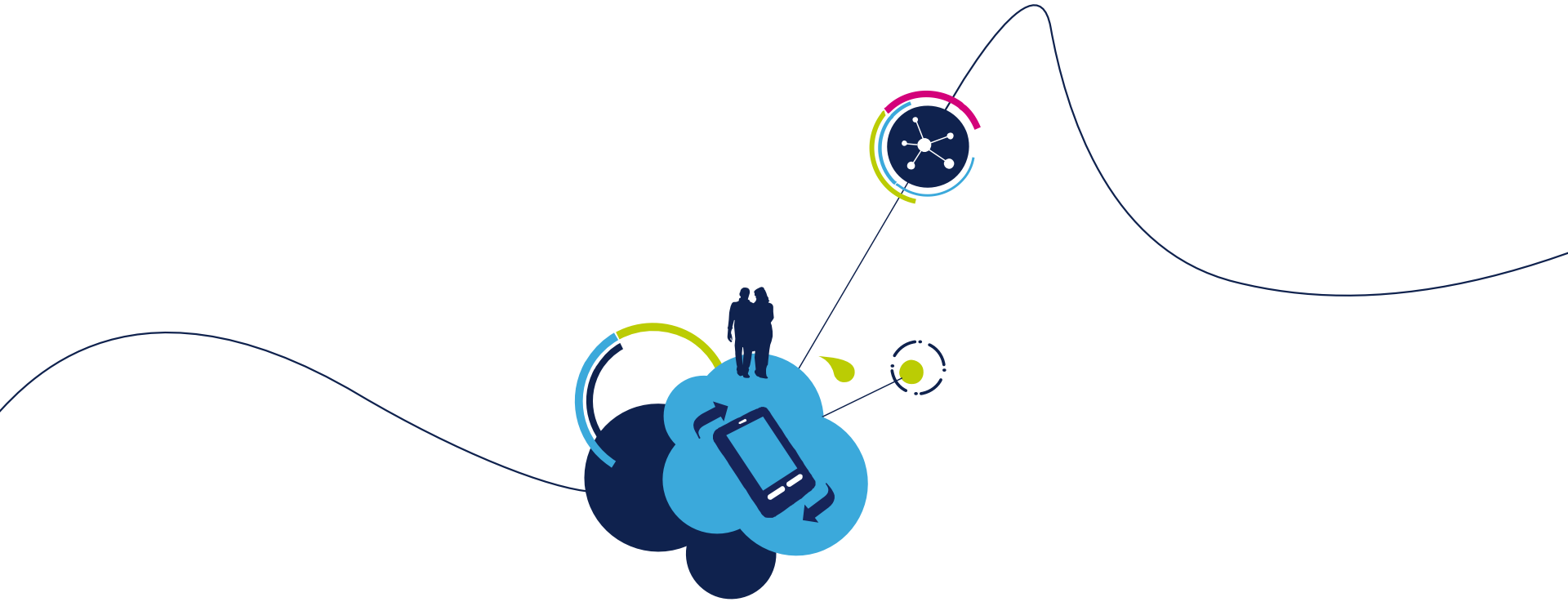
Let's list the files

- Type **AT+S.FSL**

Files stored in the User FLASH

Tera Term output

```
AT+S.FSL
AT-S.Free RAM Disk:15360
AT-S.Free EXT Disk:0
AT-S.File:E    897    WLAN.py
AT-S.File:E    0      __LOCKED__
AT-S.File:E    1719   peers.fhtml
AT-S.File:E    1246   WLAN.STA.py
AT-S.File:E    4241   config.fhtml
AT-S.File:E    1111   LB_TCP_CL.py
AT-S.File:E    1118   LB_TCP_SE.py
AT-S.File:E    1109   LB_UDP_CL.py
AT-S.File:E    1110   LB_UDP_SE.py
AT-S.File:E    2318   RL_TCP_CL.py
AT-S.File:E    2340   RL_TCP_SE.py
AT-S.File:E    2317   RL_UDP_CL.py
AT-S.File:E    2331   RL_UDP_SE.py
AT-S.File:E    1768   status.fhtml
AT-S.File:E    401    index.gz.html
AT-S.File:E    1536   stlogo.gz.jpg
AT-S.OK
```



Proceed to the next LAB!

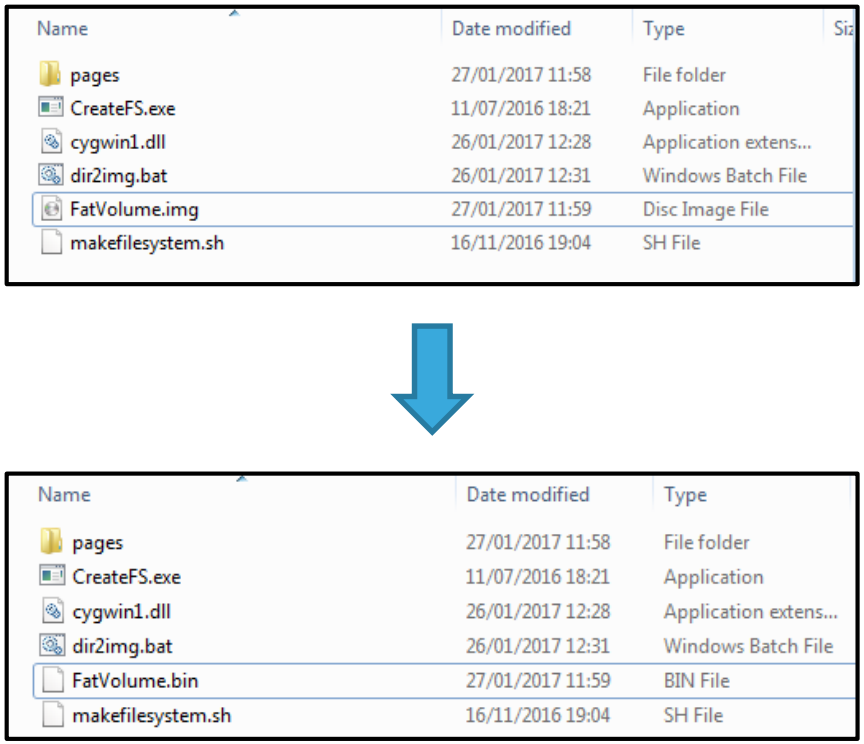
Lab 8.2.1.3.2 : Upgrade FS over UART

- Objective
  - File System update over UART

- Prerequisites
  - Work alone

Lab 8.2.1.3.2 : Upgrade FS over UART

First, you have to change the file extension of the FatVolume file.



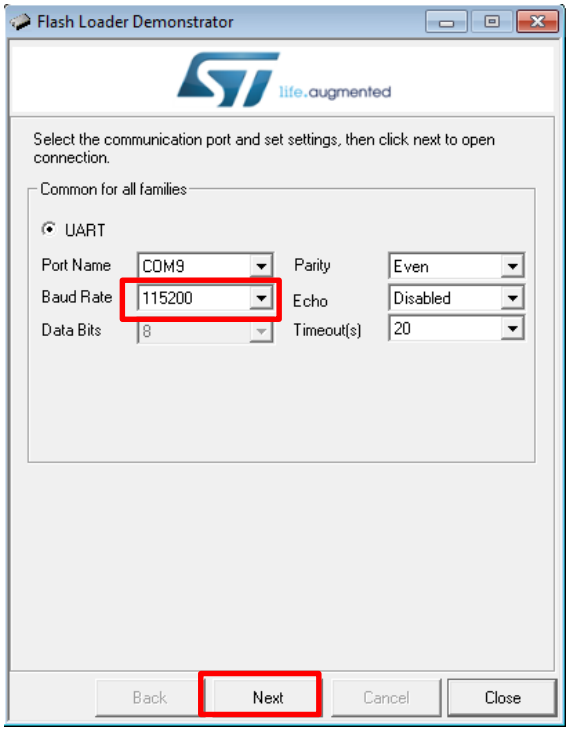
Name	Date modified	Type	Size
pages	27/01/2017 11:58	File folder	
CreateFS.exe	11/07/2016 18:21	Application	
cygwin1.dll	26/01/2017 12:28	Application extens...	
dir2img.bat	26/01/2017 12:31	Windows Batch File	
FatVolume.img	27/01/2017 11:59	Disc Image File	
makefilesystem.sh	16/11/2016 19:04	SH File	

Name	Date modified	Type
pages	27/01/2017 11:58	File folder
CreateFS.exe	11/07/2016 18:21	Application
cygwin1.dll	26/01/2017 12:28	Application extens...
dir2img.bat	26/01/2017 12:31	Windows Batch File
FatVolume.bin	27/01/2017 11:59	BIN File
makefilesystem.sh	16/11/2016 19:04	SH File

Click right on the file and on rename

Lab 8.2.1.3.2 : Upgrade FS over UART

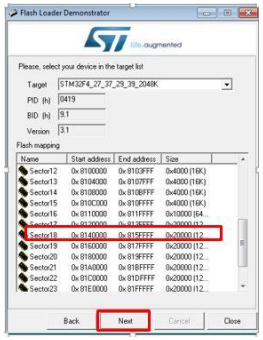
- Run flash loader with X-NUCLEO IDW04A1 board



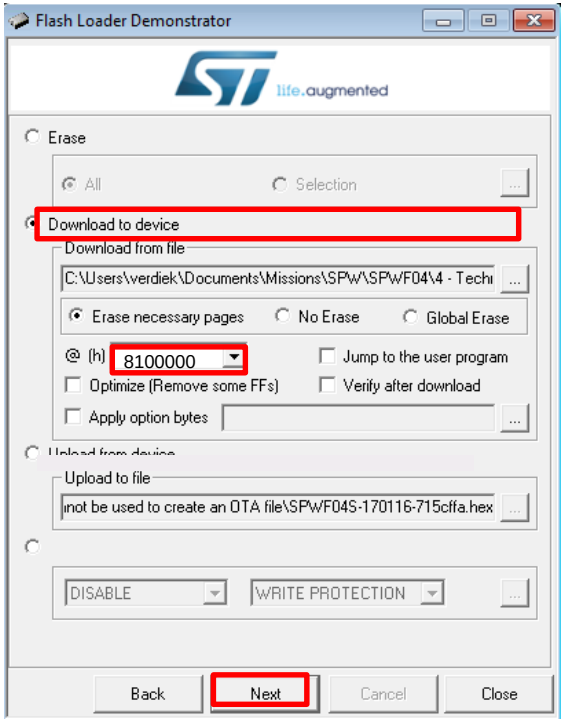
- Select COM port
  - (If not, use the PC's Device Manager to load the device driver. The USB to UART bridge should be in the list of "Ports (COM & LPT)" devices.)
- Set correct settings
  - Baud Rate =115200
  - Parity = Even
  - Echo Disabled
  - Timout 20
- Click the "Next" button.

Lab 8.2.1.3.2 : Upgrade FS over UART

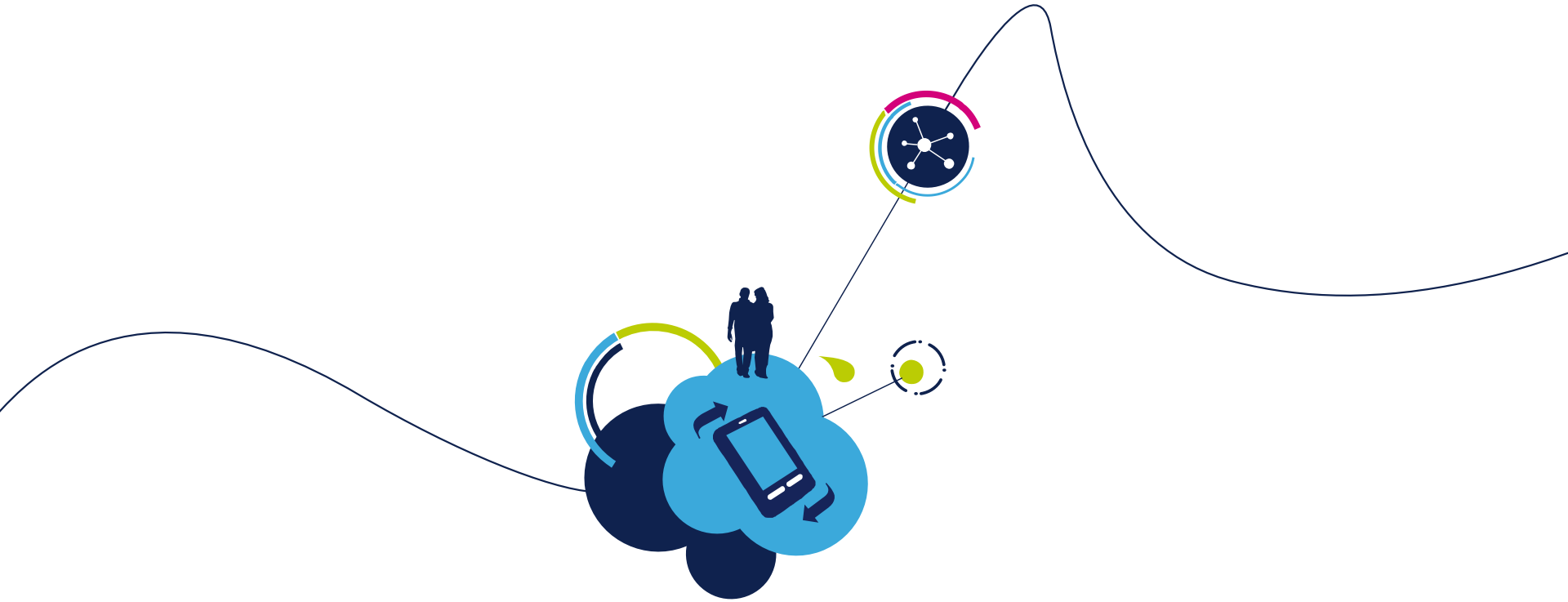
- X-NUCLEO-IDW04A1 board is ready for programming.



- If communication is OK click the "Next" button.



- Select bin file and click on « next ».
- Be careful to select right adress
  For memory mapping
  see Lab 0 (File system organisation)
- At the end of FS upgrade , remove JP2 jumper and press RESET button SW1 on X-NUCLEO-IDW04A1 board.



Proceed to the next LAB!

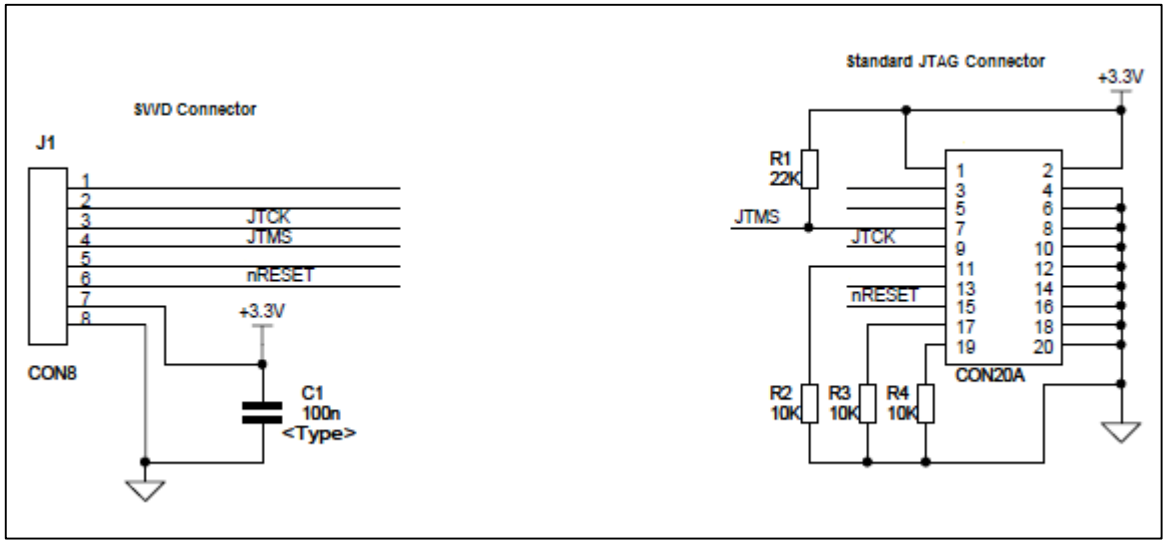
Lab 8.2.1.3.3 : Upgrade FS through SWD

- Objective
  - File System update through SWD

- Prerequisites
  - Work alone

Lab 8.2.1.3.3 : Upgrade FS through SWD

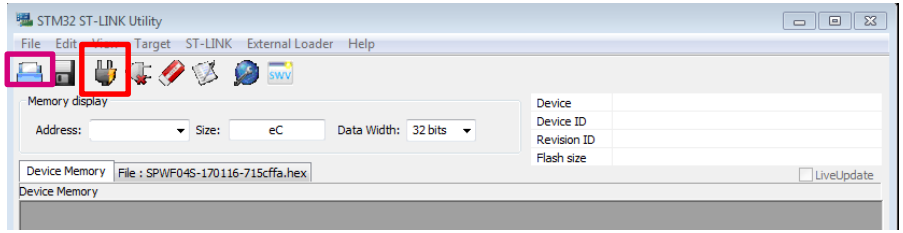
- X-NUCLEO-IDW04A1 J1 Connector details

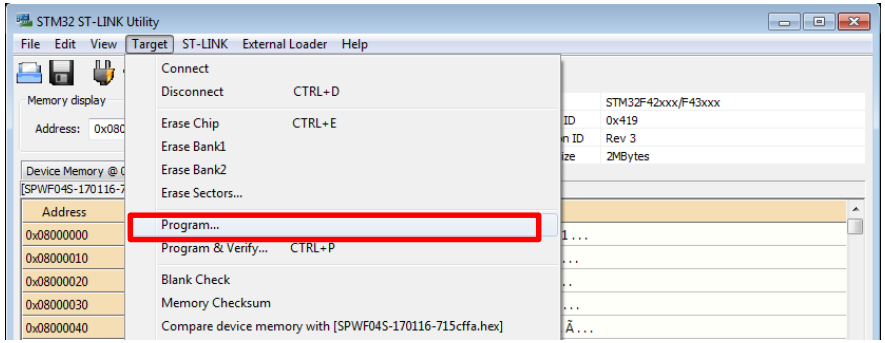


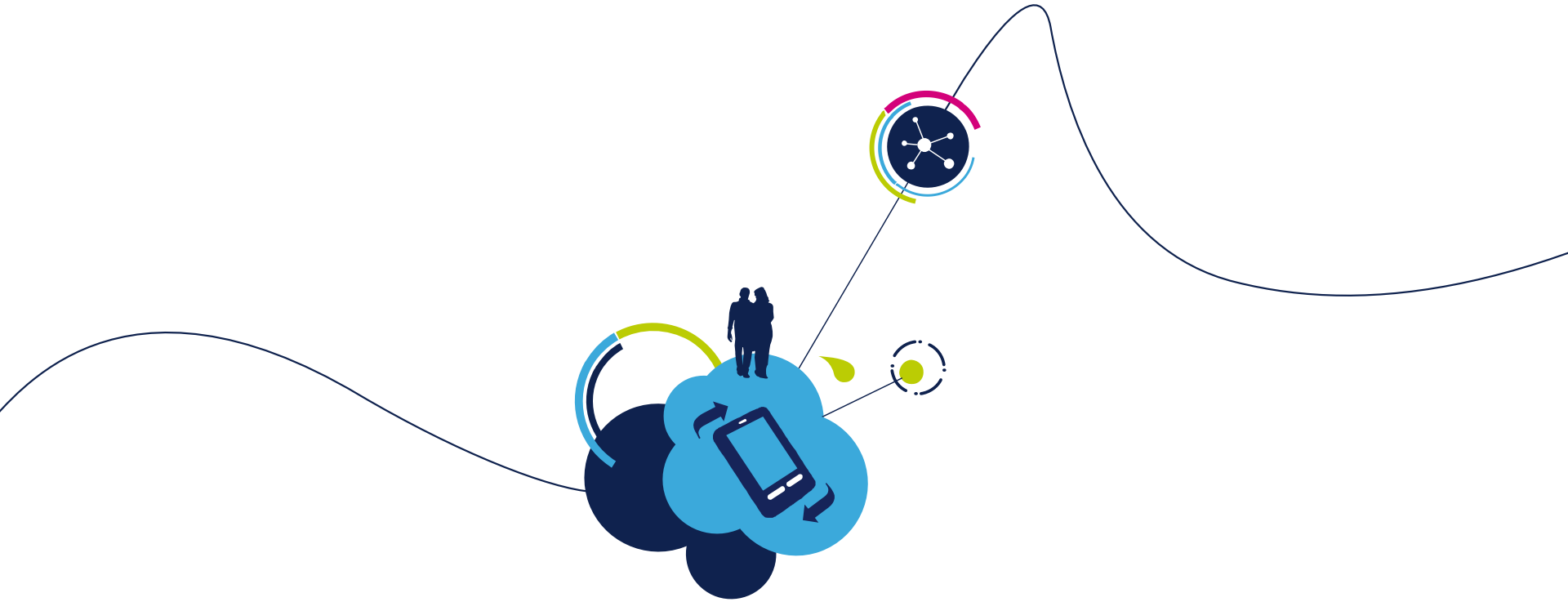
Lab 8.2.1.3.3 : Upgrade FS through SWD

- Download & install ST-Link Utility tool
  - http://www.st.com/content/st_com/en/products/embedded-software/development-tool-software/stsw-link004.html
- Program SPWF04 through SWD
  - Click on Target button
  - Click on open button
  - Select bin file
  - Be careful to select right adress

  (8100000 for example)
  - Program SPWF04







Proceed to the next LAB!

Lab 8.2.1.4 : Create Dynamic Page

- Objective
  - Discover embedded functionalities allowing to have dynamic web pages
  - See the contain of a file
- Prerequisites
  - Work in couple

Lab 8.2.1.4.1 : Input Demo

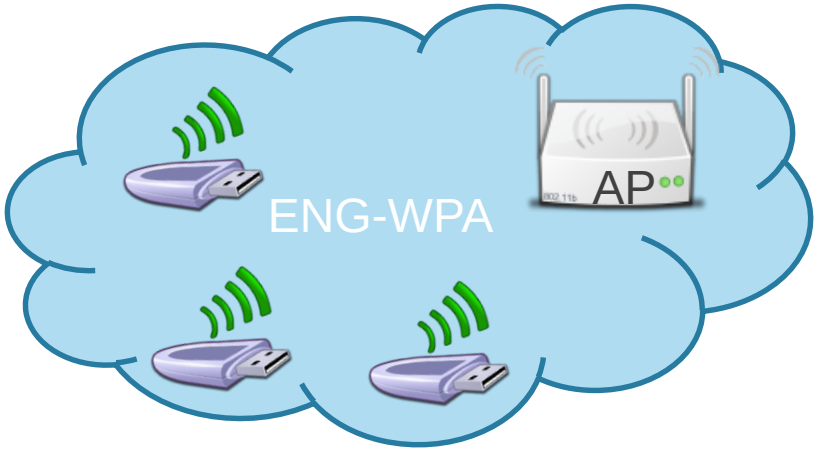
- Objective
  - Send a message from the server (wifi module) to an external client connected to the same network
- Prerequisites
  - module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

Lab 8.2.1.4.1 : Input Demo

The module provides some DEMOs to show the interaction between the module and an external client connected to the same network.

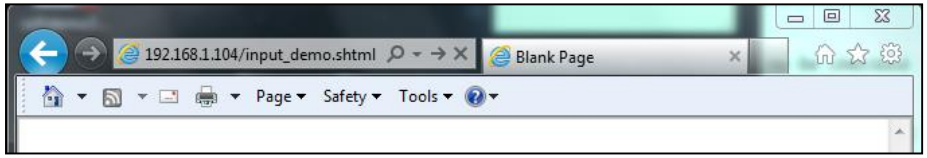
The module should be configured in Mini AP mode (as shown in Lab 3) or should be connected to the AP (as shown in Lab 4).

In order to run this demo, the client have to open the **input_demo.shtml** page stored in the module.



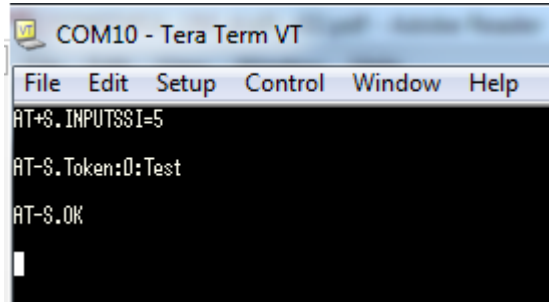
Lab 8.2.1.4.1 : Input Demo

- Open your web browser
- In the address bar, type 192.168.x.1xx/input_demo.shtml



- Type AT+S.INPUTSSI=<length><cr>{data}

On Tera term

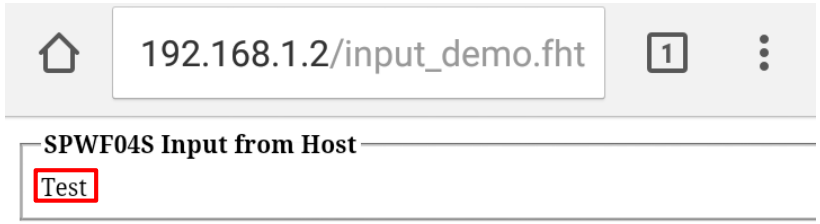


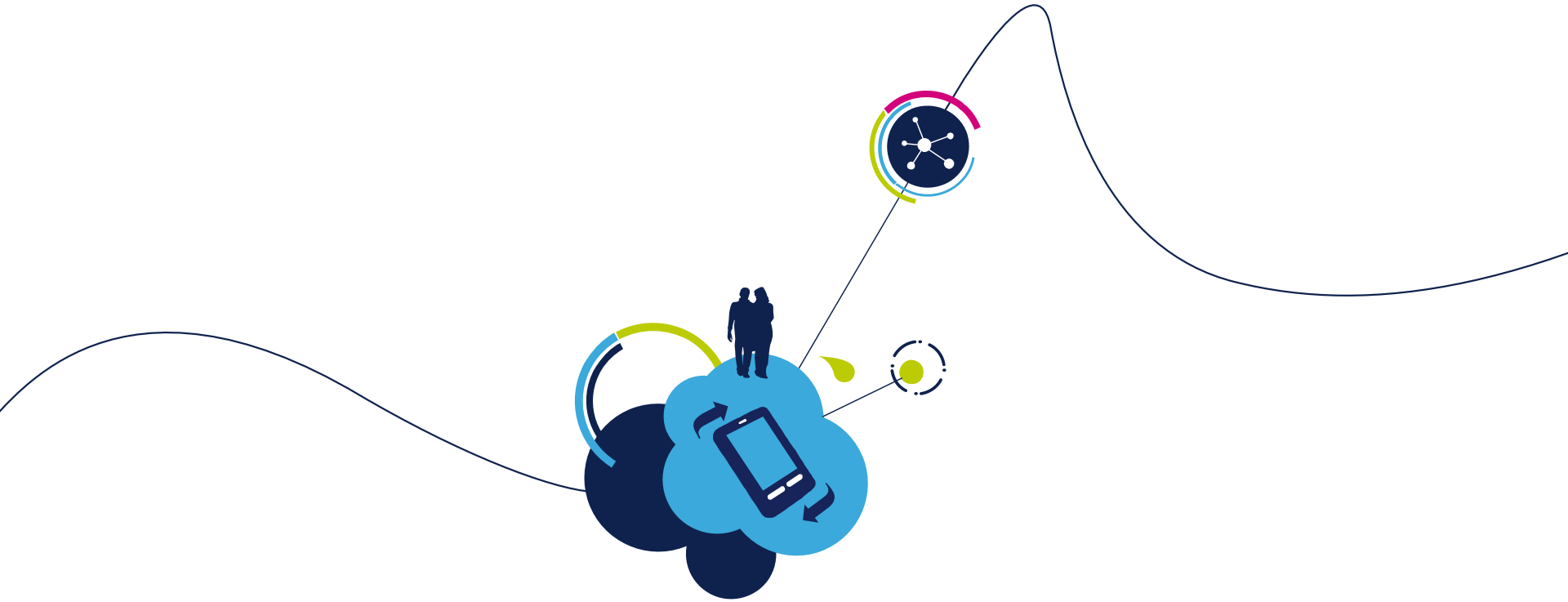
! ***Note: If the buffer is already full, WIND:56 is not shown, so take care of data length***

Note2: data[0] is used as separator. Please refer to UM2114 for a complete description of such command. Here, {data} is {\|Test}

Lab 8.2.1.4.1 : Input Demo

- The module receives the HTML page containing the string inserted server-side.





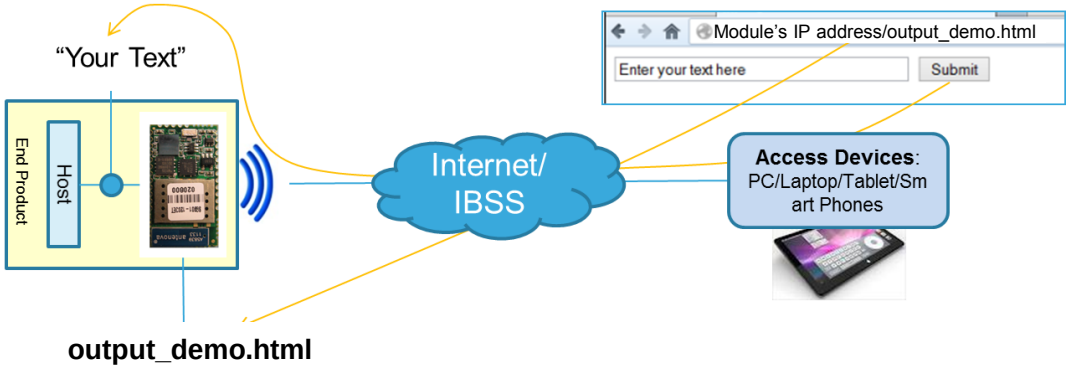
Proceed to the next LAB!

Lab 8.2.1.4.2 : Output Demo

- Objective
  - Send a message from an external client to the wifi module connected to the same network
- Prerequisites
  - module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

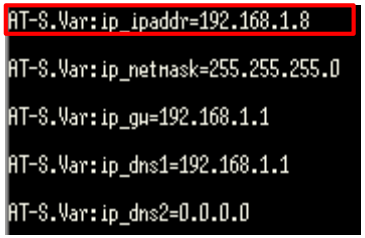
Lab 8.2.1.4.2 : Output Demo

A built-in html page “output_demo.html” allows to remotely push characters on the serial port from a remote browser.

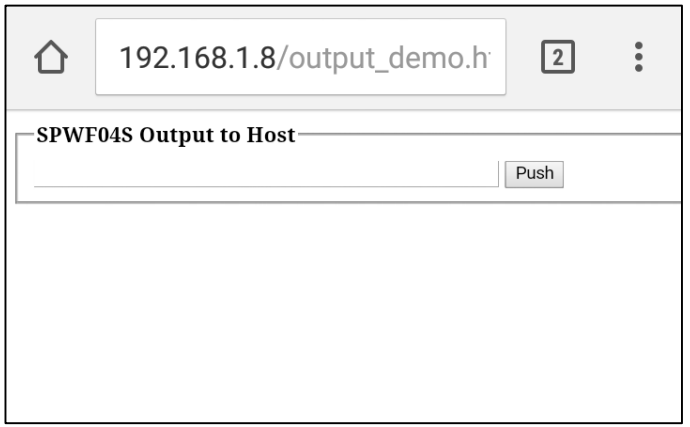


Lab 8.2.1.4.2 : Output Demo

- Find your IP address
  - Type **AT+S.STS**

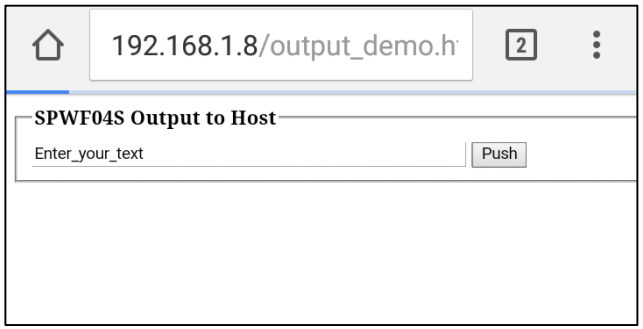


- Associate your computer with the AP
- Open your web browser
- In the address bar, type <module's IP Address>/output_demo.html
  - Type 192.168.1.8/output_demo.html



Lab 8.2.1.4.2 : Output Demo

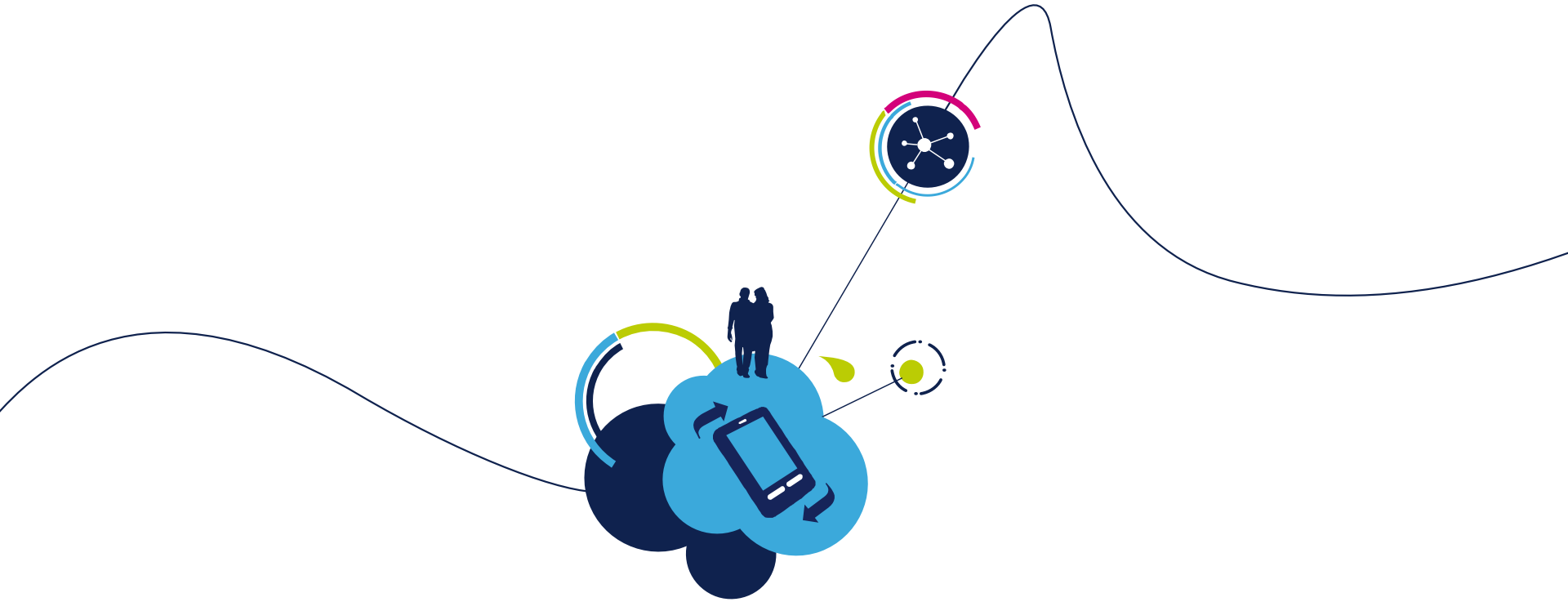
- Enter the text (Test max lenght is 64)
- Click on «Push»



- The text will be sent to the serial port of the module

```
+WIND:57:Output from remote:15:Enter_your_text
```

! ***Note: Max allowed length of sent string is 128. Refer to Lab8.2.2 or AN4965 "WebServer on SPWF04S module" if you need to Post files***



Proceed to the next LAB!

Lab 8.2.1.4.3 : Remotely control - GPIOs

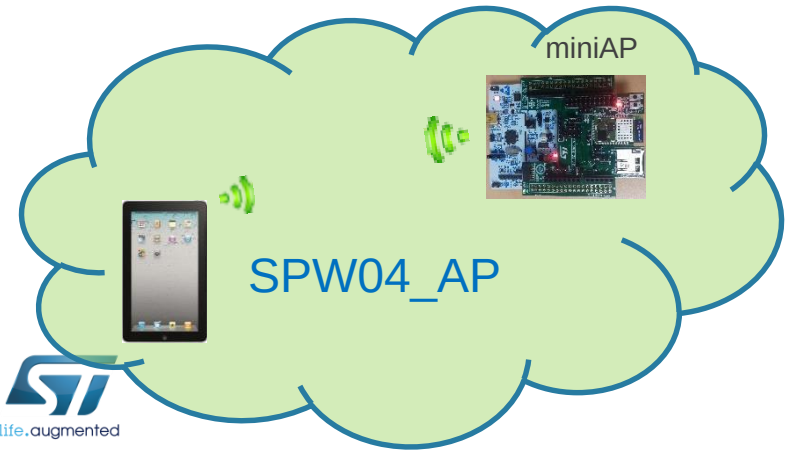
- Objective
  - write remotely a GPIO
  - configure remotely a GPIO
  - read remotely a GPIO
- Prerequisites
  - module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

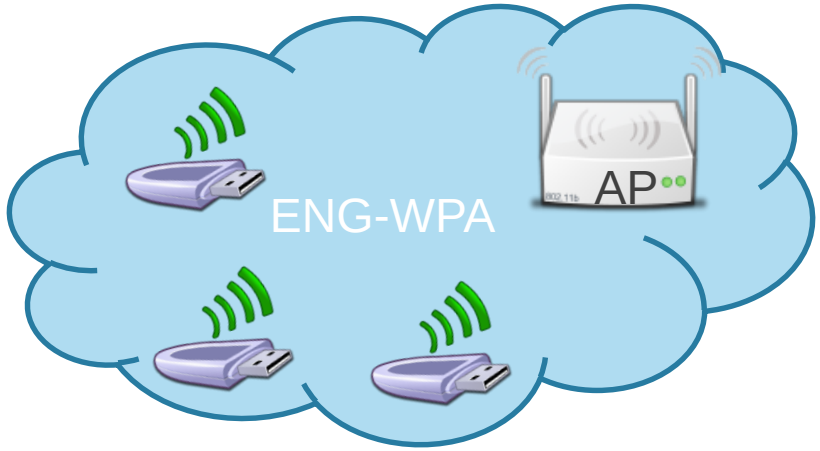
Lab 8.2.1.4.3 : Remotely control - GPIOs

This feature allows to remotely write, configure and read a GPIO.

Here the module will be configured in Mini AP mode.

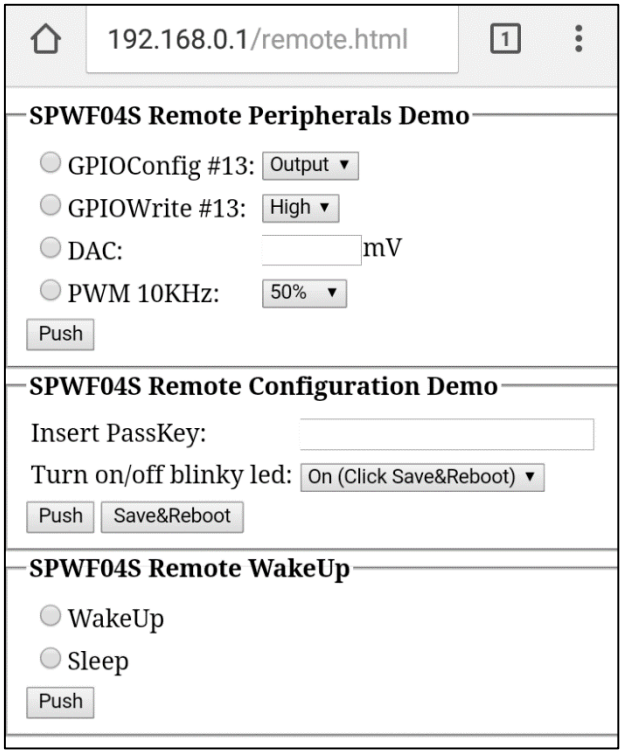
The external client have to open the **remote.html** page stored in the module.

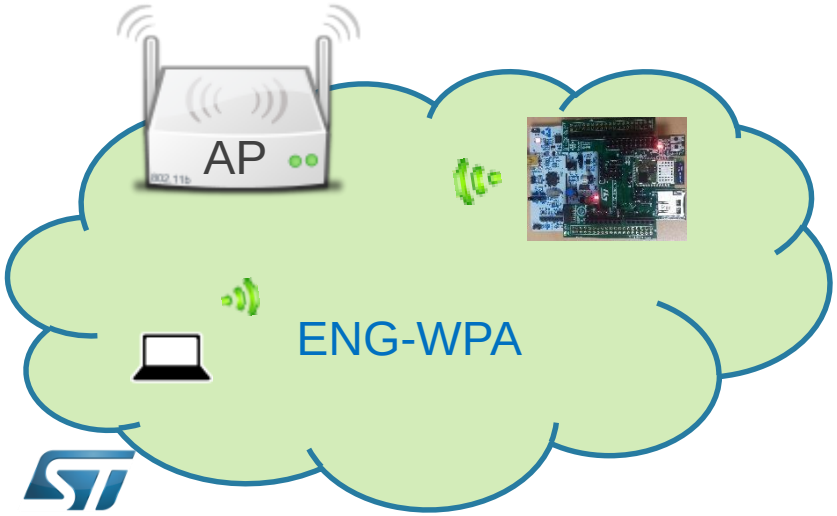




Lab 8.2.1.4.3 : Remotely control - GPIOs

- Open your web browser
- In the address bar, type 192.168.x.1xx/remote.html

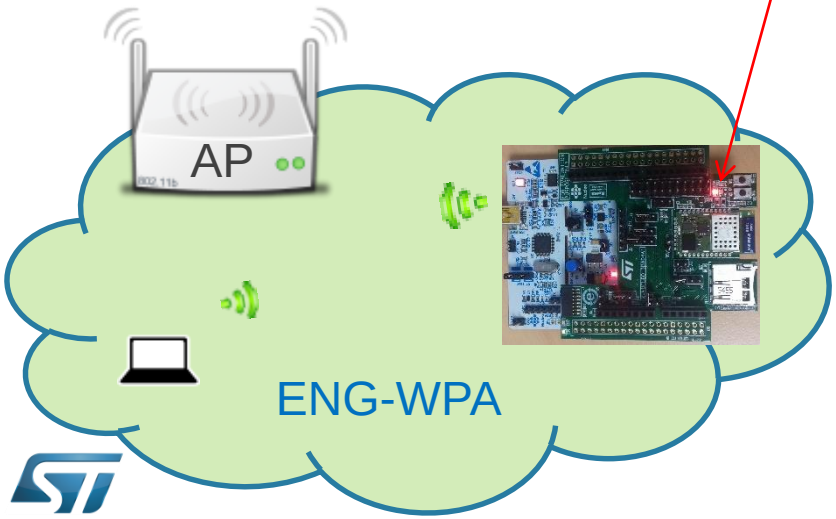


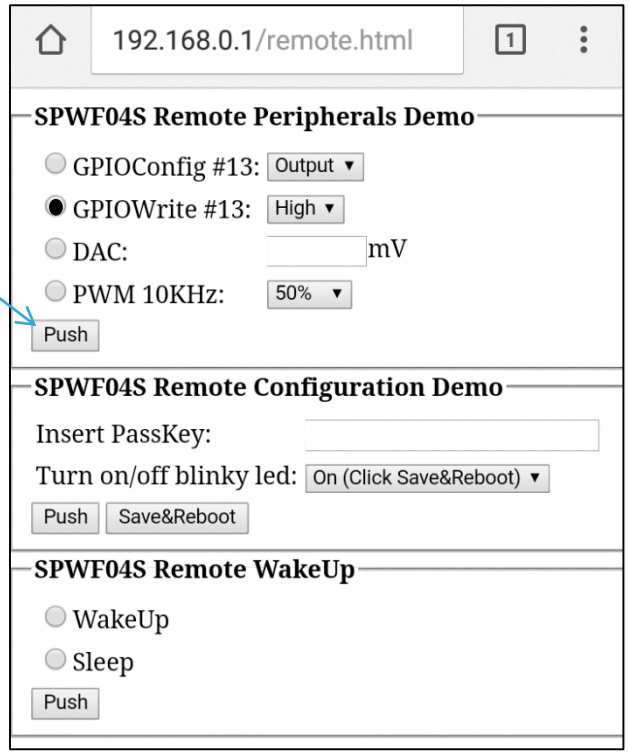


life.augmented

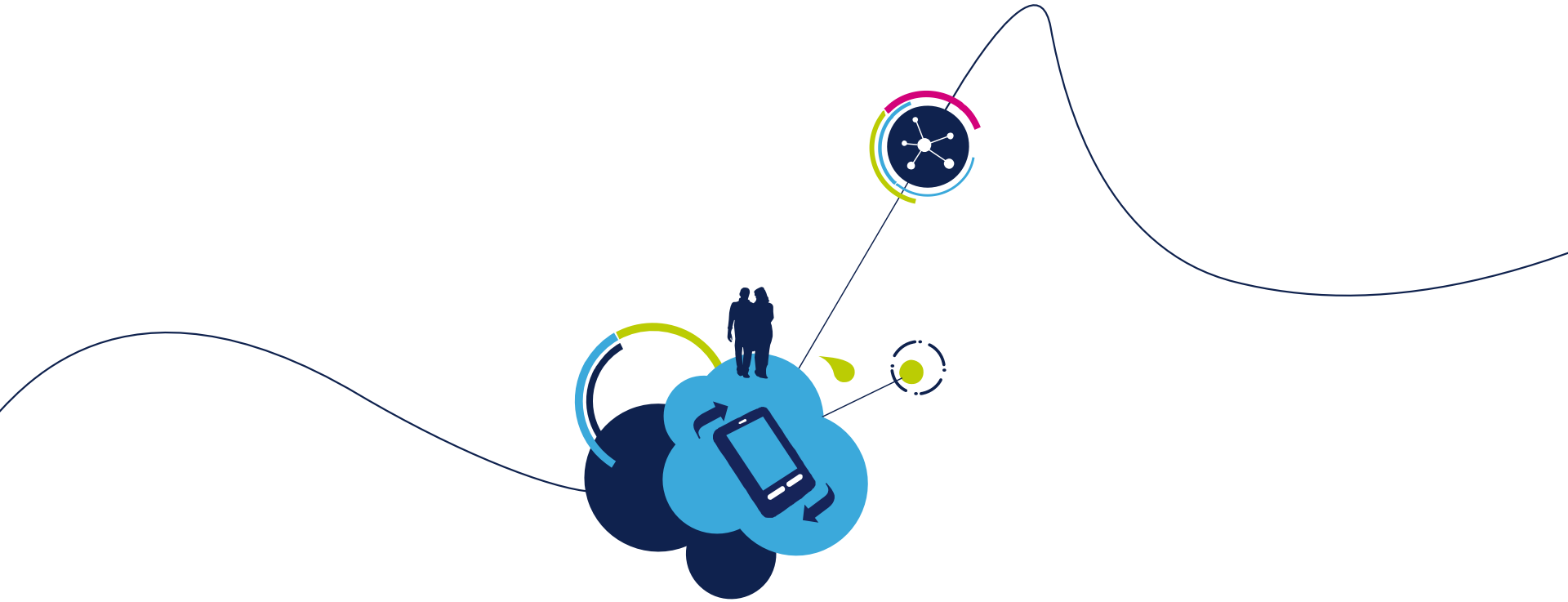
Lab 8.2.1.4.3 : Remotely control - GPIOs

- Try to write the GPIO13connected to the LED3
- Click on the «Push» button and LED3 will switch on





life.augmented



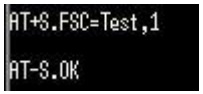
Proceed to the next LAB!

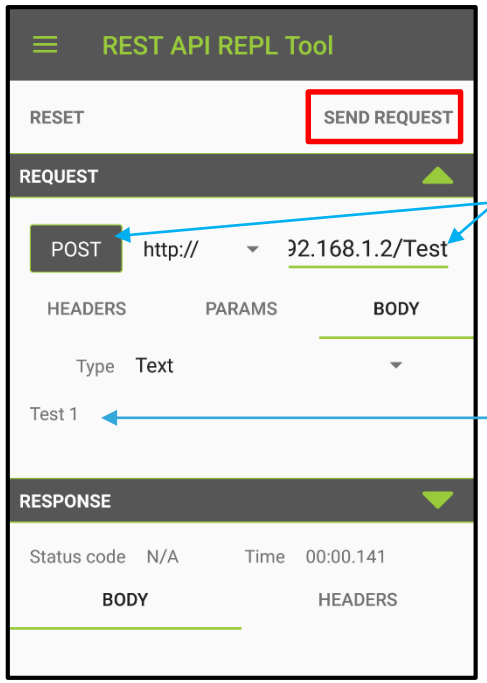
Lab 8.2.2 : Web Server Usage

- Objective
  - Get a file from the SPWF04 through a device
  - Send a file to the SPWF04 through a device
- Prerequisites
  - Module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

Lab 8.2.2 : Web Server Usage
Move data in a file on internal FS

For this part, we will use a smartphone with REST API REPL Tool App.
Smartphone and SPWF04 need to be connected to the same network.





- **Create a file of any character in your module**
- **Choose « Post » and enter the IP adress of the SPWF04 on your smartphone and the name of the file in the module**
- **Enter the text you want to put in the file**
- **Click the «Send Request» Button**

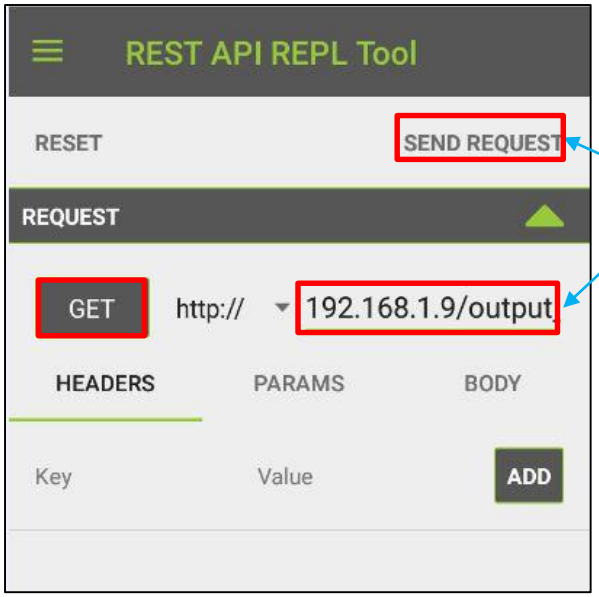
Tera Term output

AT+S.FSP=Test,,

Test 1AT-S.OK

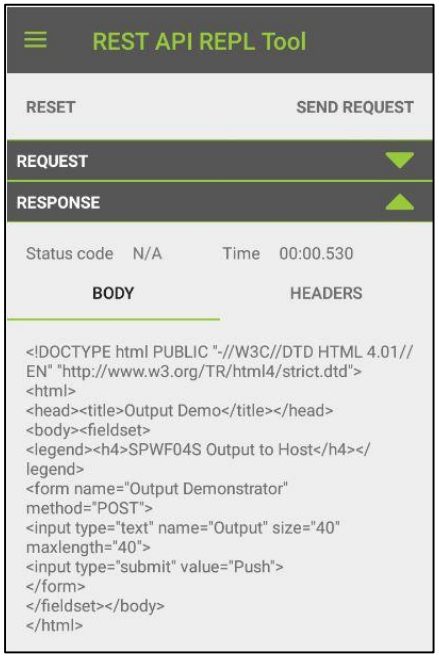
life.augmented

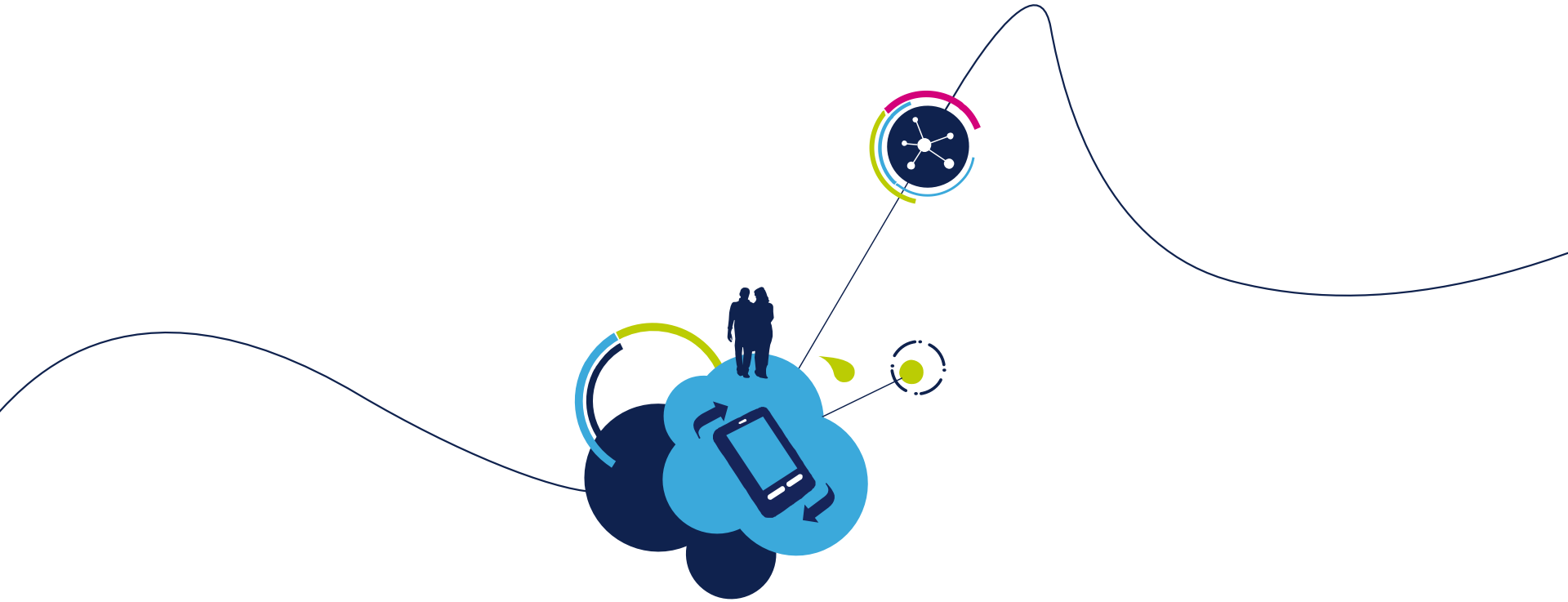
Lab 8.2.2 : Web Server Usage Download file from internal FS



- Enter the IP adress of the SPWF04 on your smartphone and the document you want do download (in this exemple it's output_demo.html)
- Choose « Get » and click the « SEND REQUEST » Button

- You can see the content of the file in the response





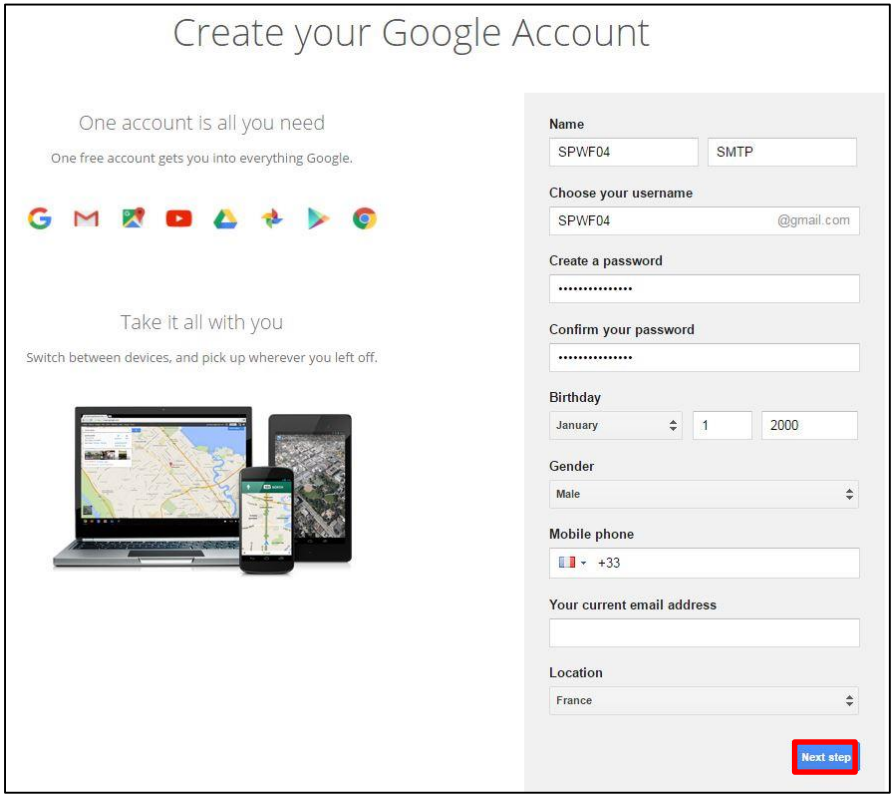
Proceed to the next LAB!

Lab 9 : SMTP

- Objective
  - Send an email to a SMTP Server (For pratically reasons, here, the SMTP through secure server will not be aborded)
- Prerequisites
  - Module connected to an AP with internet connection (Lab 4.2)

Lab 9 : SMTP

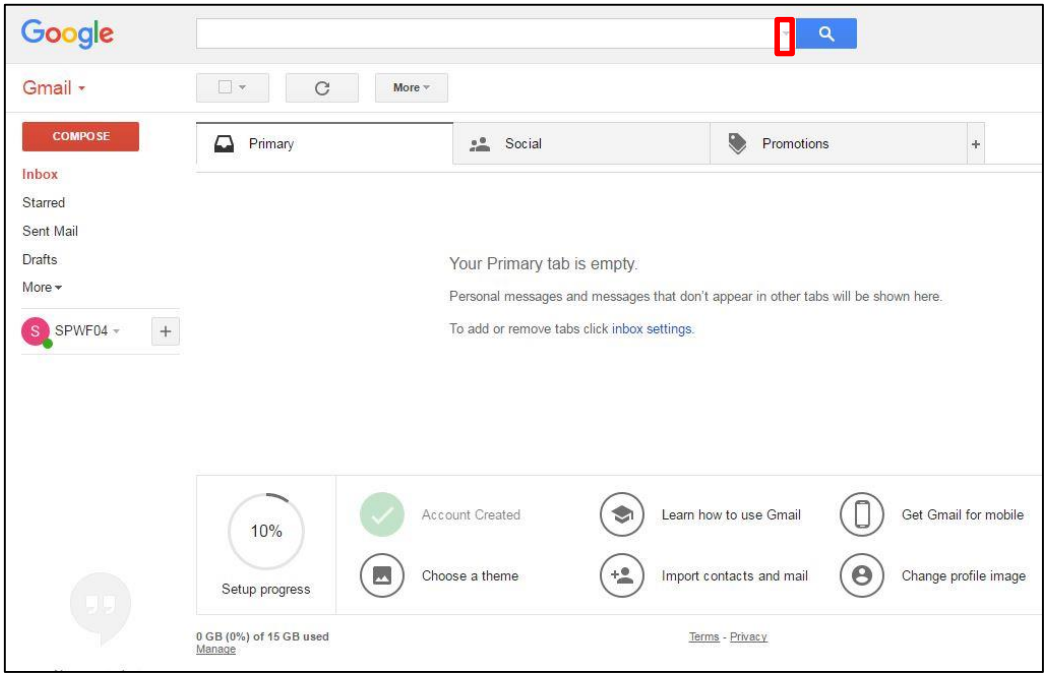
Following example is requesting to create a gmail account (aspmx.l.google.com allows the utilisation of the 25 port).



- Open the following link : https://accounts.google.com/SignUp?service=mail&continue=https%3A%2F%2Fmail.google.com%2Fmail%2F&hl=en
- Complete the form
- Click the “Next Step” button

Lab 9 : SMTP

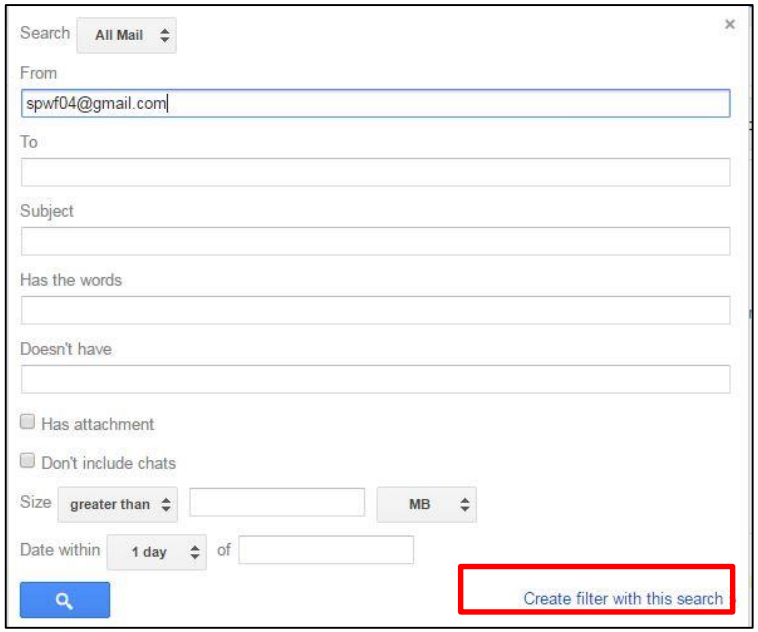
On the inbox page, you have to change the settings



- Click this button

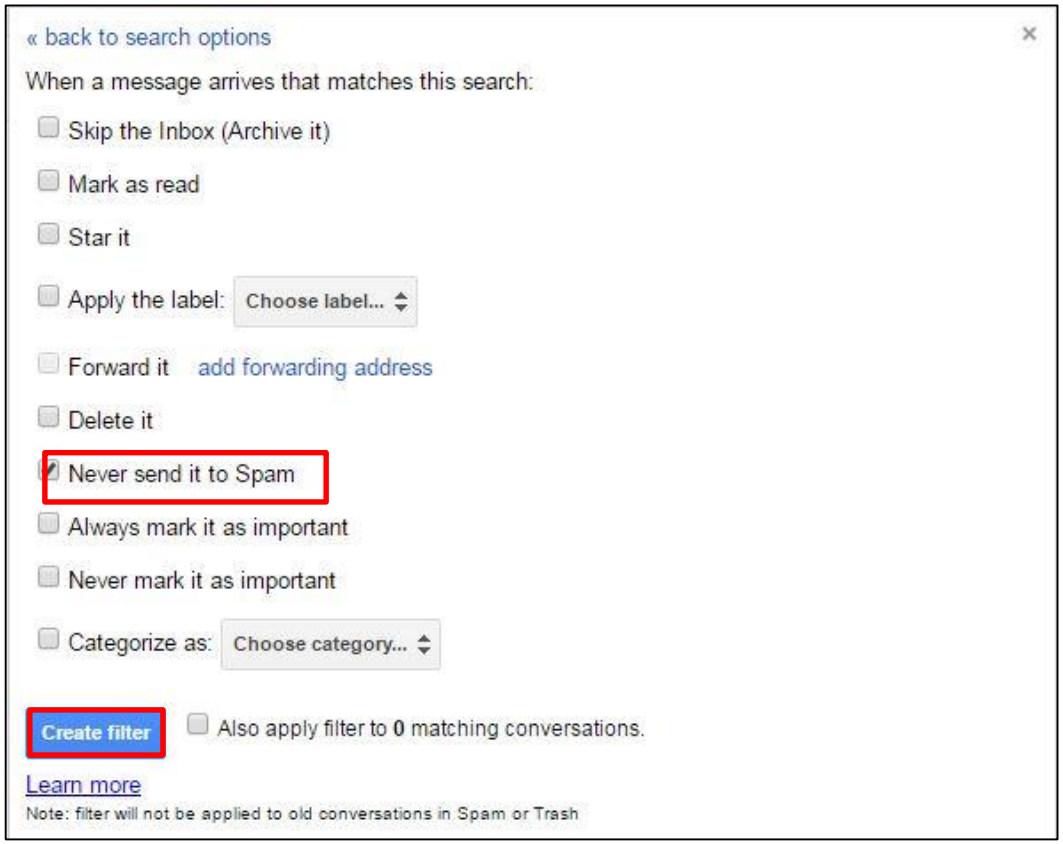
Lab 9 : SMTP

- Enter your email address
- Click on “Create filter with this search”



Lab 9 : SMTP

- Check the box next to “Never send it to Spam”
- Click on “Create filter”



Lab 9 : SMTP

The stack implements the protocol SMTP to send a secure email.

This command accepts data after the <cr> at the end of the command line.

- Syntax
  - **AT+S.SMTP=**<hostname>,[<port>],[<TLS Kind>],[<username>],[<passwd>],[<ID>],<Address>,<TO>,,,<Subject>,,<Len><cr>{data}

- Configuration parameters
  - <Hostname> DNS resolvable Name or IP address of the remote host
  - <(Port)> Default is 25 (if TLS=0) or 465 (if TLS>0). Server Port.
  - <(TLS Kind)> Default:unsecured TLS Security option. 0->unsecured, 5-> SMTPS on port 465 if available, otherwise SMTP + STARTTLS if available, otherwise no security, 8-> SMTP + STARTTLS if available, otherwise the mail is not sent, 9->SMTPS on port 465 if available, otherwise SMTP + STARTTLS if available, otherwise the mail is not sent.

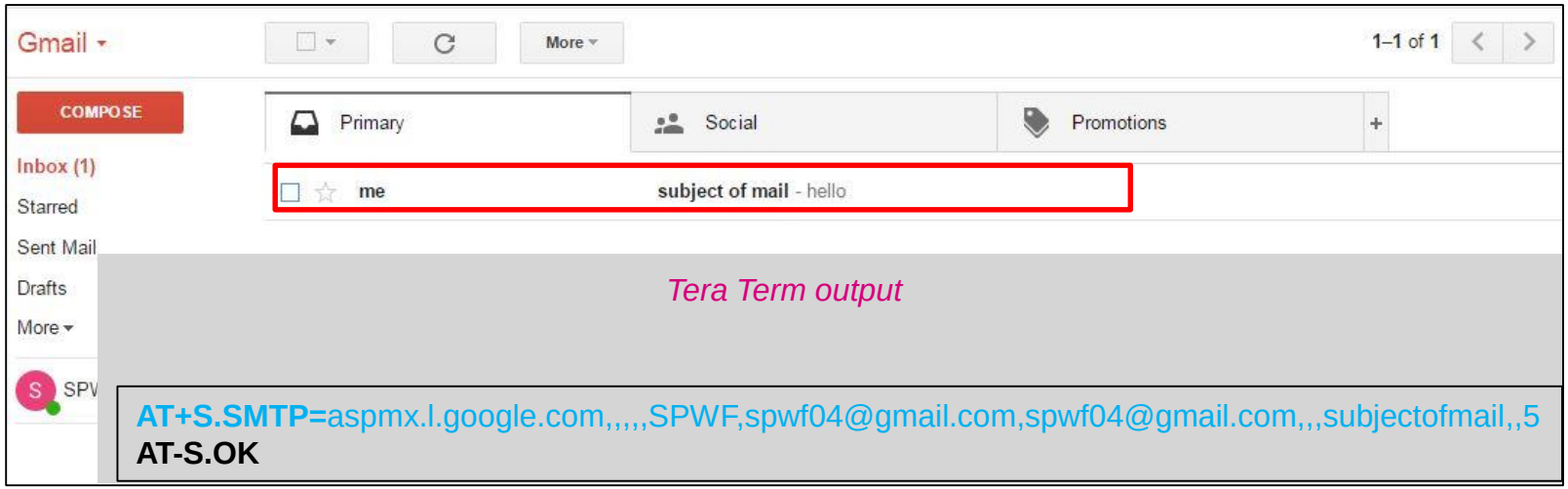
Lab 9 : SMTP

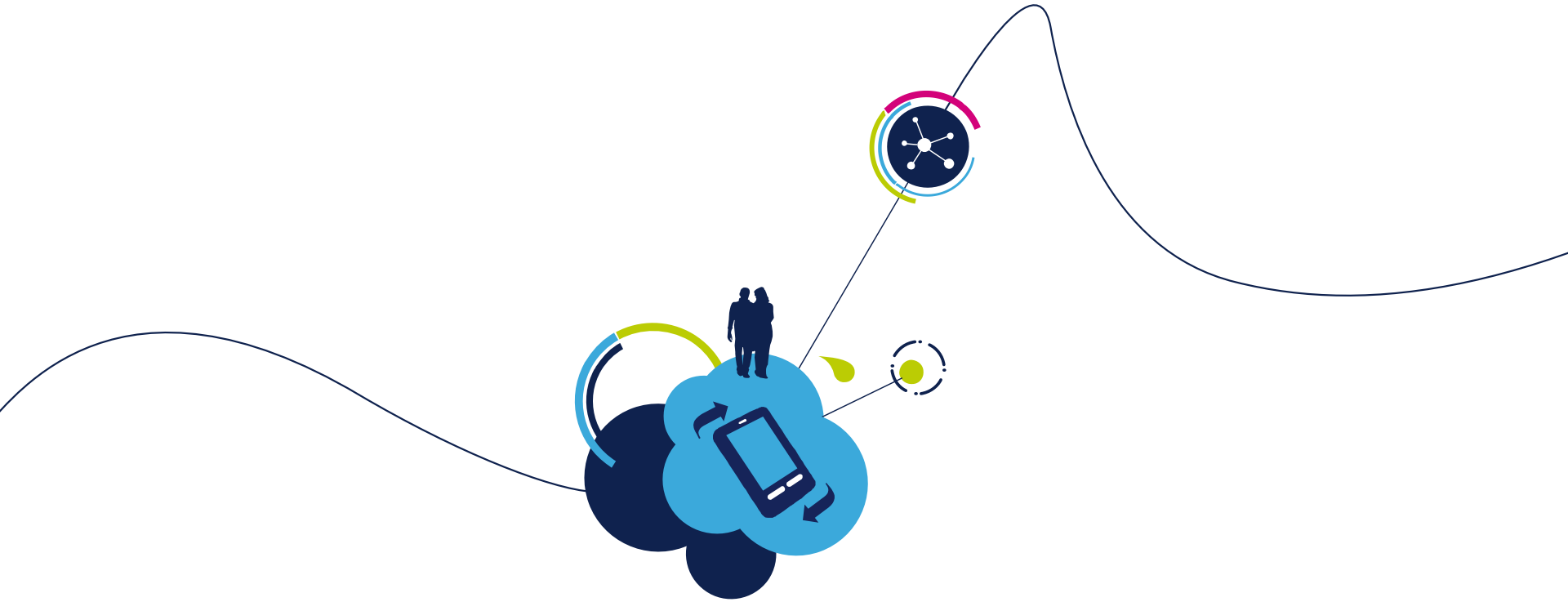
- Configuration parameters
  - <username> User of the SMTP server
  - <passwd> Passwd of the SMTP server
  - <ID> Default: nv_model used during Helo
  - <Adress> Email address on the SMTP server
  - <TO> Destinator emails. Multiple emails are separated by a semicolon
  - <Subject> Email Subject. String Message.
  - <Len> Length of the Body Message.

Lab 9 : SMTP

- In Tera Term: Type

AT+S.SMTP=aspmx.l.google.com,,,,,SPWF,spwf04@gmail.com,spwf04@gmail.com,,,subjectofmail,,5<CR>hello





Proceed to the next LAB!

Lab 10 : MQTT

- Objective
  - Suscribe/Publish to a topic

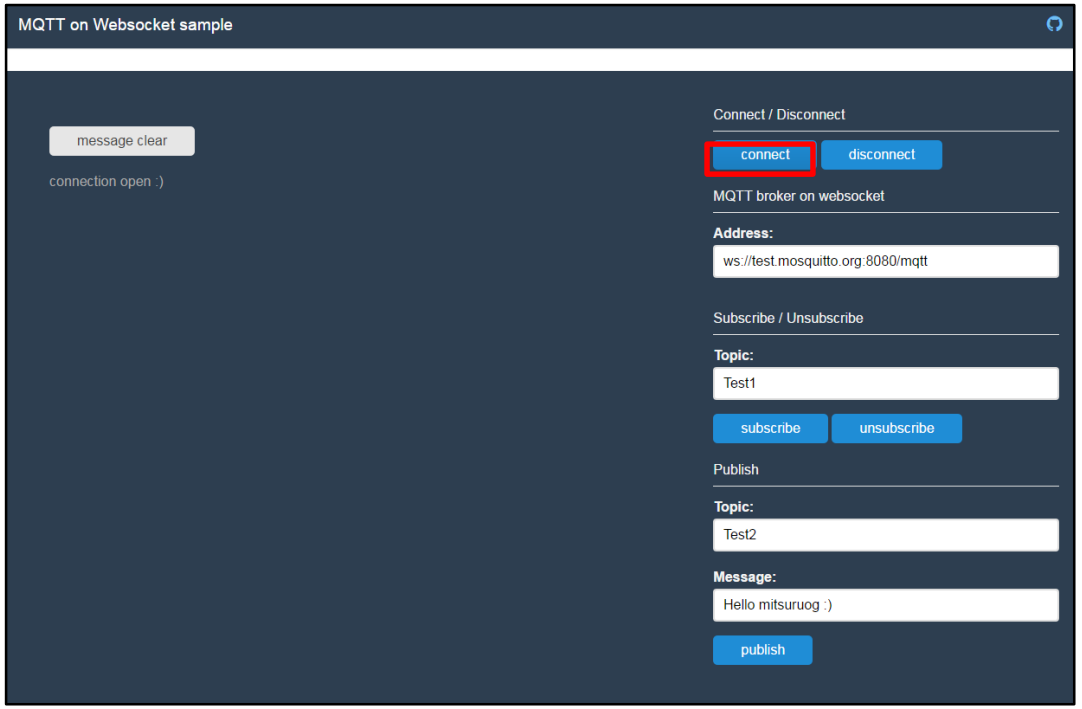
- Prerequisites
  - Your module have to be connected to an Access point with an Internet Connection (Lab 4.2)

Lab 10 : MQTT

You need to use a MQTT Broker, for example you can use this one : http://mitsuruog.github.io/what-mqtt/ (As you can see on the top of the picture below, the MQTT stack is, here, base on Websocket)

The server at the address “test.mosquitto.org” being in free access, it could be down. If it’s the case, don’t hesitate to try with another on like “broker.hivemq.com”

- Click on Connect



Lab 10 : MQTT

To open a connection with an MQTT Broker. The command **AT+S.MQTTCONN** returns a local ID=0, used in the correspondent commands. The device manages one MQTT connection at a time.

- Syntax
  - **AT+S.MQTTCONN=**<hostname>,[<port)>],[<path>],[<use TLS>],[<username>],[<passwd>],[<userID>],[<KeepAlive>],[<Retry>],[<LastWill QoS>],[<LastWill Topic>],[<lastWill Message>]<cr>
- Configuration parameters
  - <Hostname> DNS resolvable name or IP address of the MQTT Broker
  - <(port)> Default:1883. TCP socket port.
  - <(path)> Default:/.
  - <(use TLS)> Default: 0. Values range: 0->unsecured; 1->autodetect; 2-> TLS
  - <username> Default:none. User Name
  - <passwd> Default: none. Passwd

Lab 10 : MQTT

- Configuration parameters
  - <(userID)> Default: nv_model used during MQTT communications
  - <(KeepAlive)> Default:60 seconds.
  - <(Retry)> Default:15 seconds
  - <(LastWill QoS)> Default: 0. Last action to be executed by the broker when the node disappears without a disconnect procedure.
  - <(LastWill Topic)> Default: None. Last Will Topic
  - <(LastWill Message)> Default: None. Published on the Last Will Topic

Lab 10 : MQTT

So, with the previous example you can type :

AT+S.MQTTCONN=test.mosquitto.org,8080,/mqtt,,,,SPWF04S,,,,,

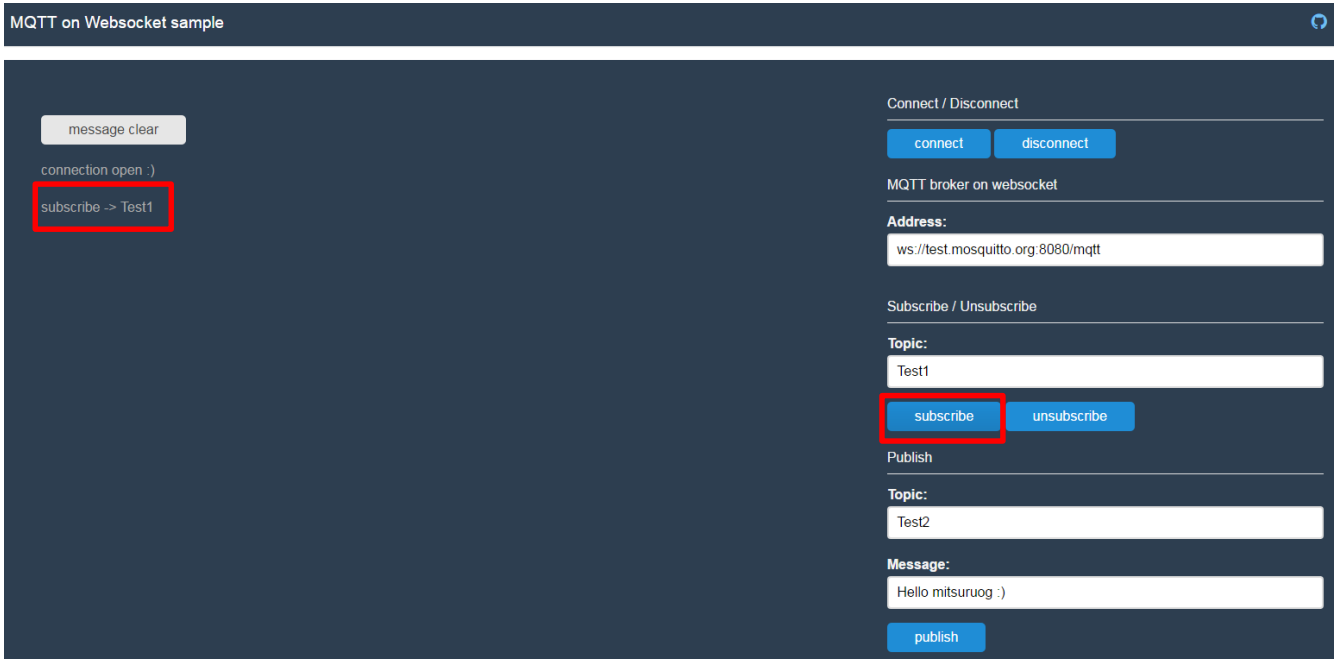
Tera Term output

AT+S.MQTTCONN=test.mosquitto.org,8080,/mqtt,,,,SPWF04S,,,,,
AT-S.On:0:0
AT-S.OK

Lab 10 : MQTT

Now you can subscribe to a topic, for example, Test1

- Click on subscribe



Lab 10 : MQTT

The **AT+S.MQTTPUB** is used to publish a message to an MQTT Broker.
This command accepts data after the <cr> at the end of the command line.

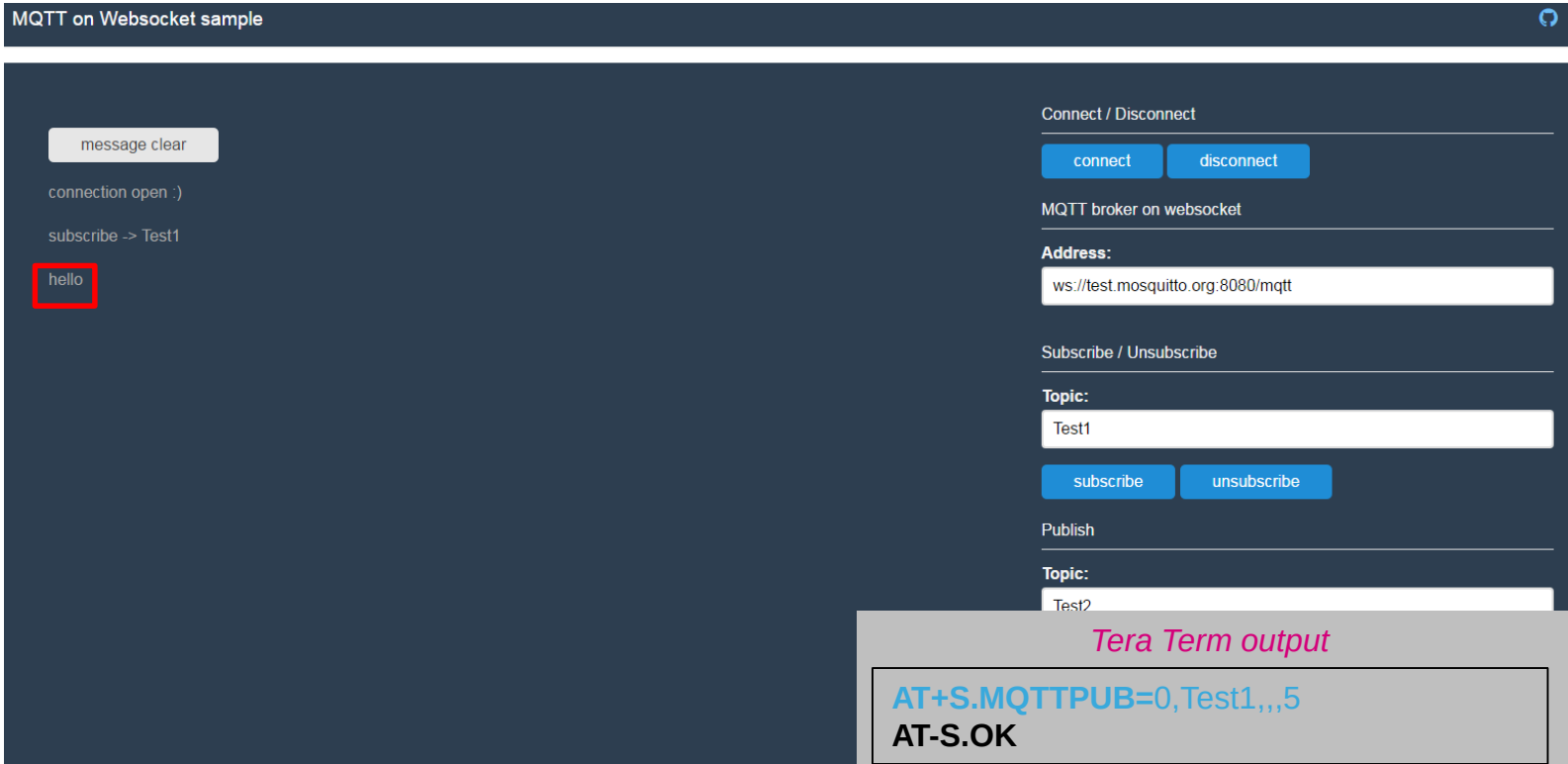
- Syntax
  - **AT+S.MQTTPUB=**0,<Topic>,[<QoS>],[<Retained Flag>],<Len><cr>{data}

- Configuration parameters
  - <Topic> Topic where the message is published
  - <QoS> Default: 0. Values Range: 0->at most once delivery; 1-> at least one delivery; 2-> exactly one delivery
  - <Retained flag> Default: 0. Possible values: 0->do not retain, 1 -> retain
  - <Len> MQTT message length

Lab 10 : MQTT

You can now plubished something on Test1 topic :

AT+S.MQTTPUB=0,Test1,,,5<CR>hello



Tera Term output

AT+S.MQTTPUB=0,Test1,,,5
AT-S.OK

Lab 10 : MQTT

The **AT+S.MQTTSUB** is used to subscribe topic to an MQTT Broker.

- Syntax
  - **AT+S.MQTTSUB=**0,<topic>,[<QoS>]<cr>
- Configuration parameters
  - <topic> Topic where the node subscribe to
  - <(QoS)> Default:0. Values Range: 0->at most once delivery; 1-> at least one delivery; 2-> exactly one delivery.

Lab 10 : MQTT

You can now subscribed to a topic :

AT+S.MQTTSUB=0,Test2,<CR>

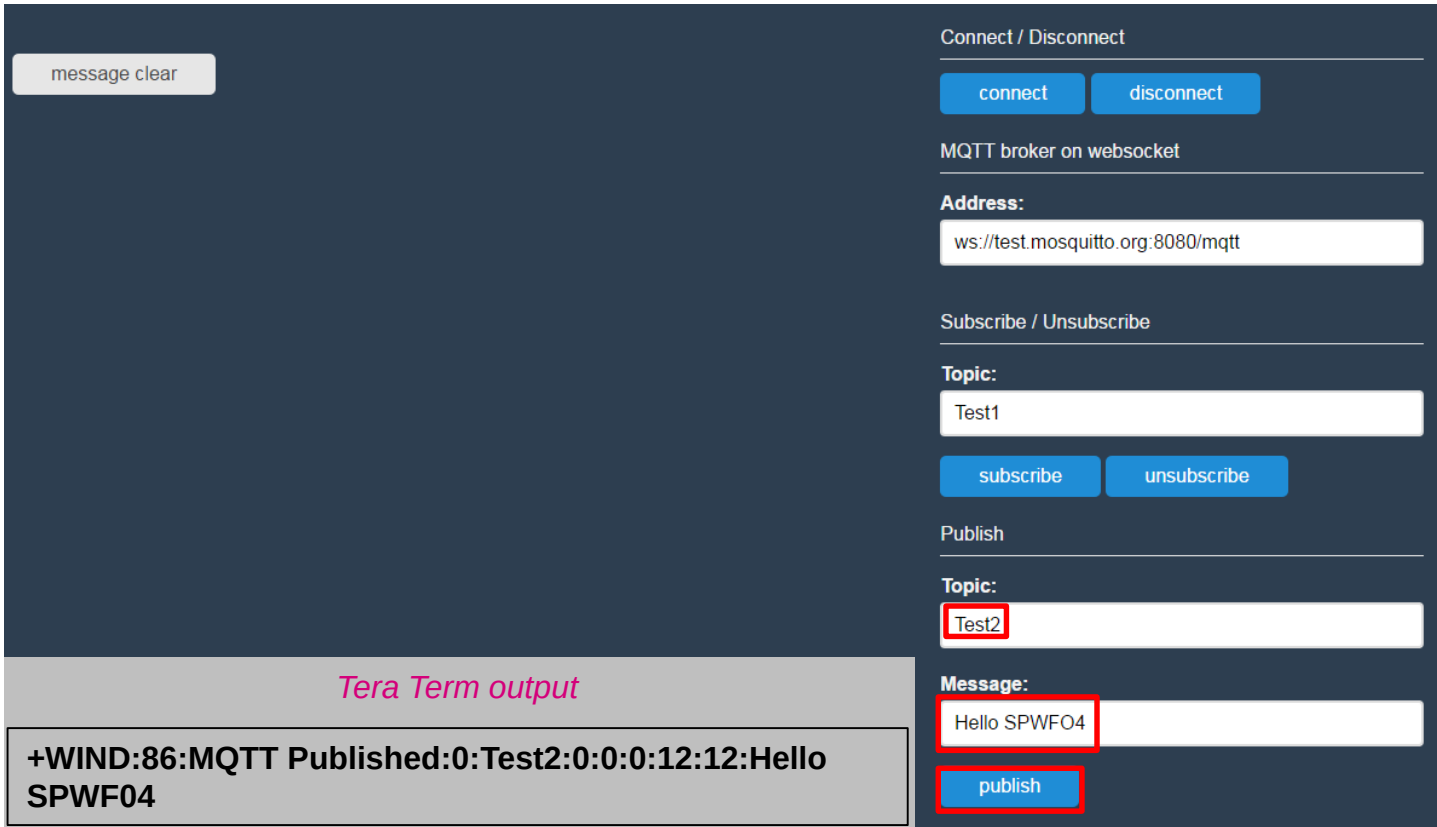
Tera Term output

AT+S.MQTTSUB=0,Test2,
AT-S.OK

Lab 10 : MQTT

Now if you publish something on the topic Test2 :

- Enter «Test2» in Topic
- Write your message
- Click the «publish» button



Lab 10 : MQTT

The **AT+S.MQTTUNSUB** is used to unsubscribe topic from an MQTT Broker.

- Syntax
  - **AT+S.MQTTSUB=**0,<topic><cr>
- Configuration parameters
  - <topic> Topic where the node unsubscribe from

Lab 10 : MQTT

If you unsubscribed to the Test2 topic :

AT+S.MQTTUNSUB=,Test2<CR>

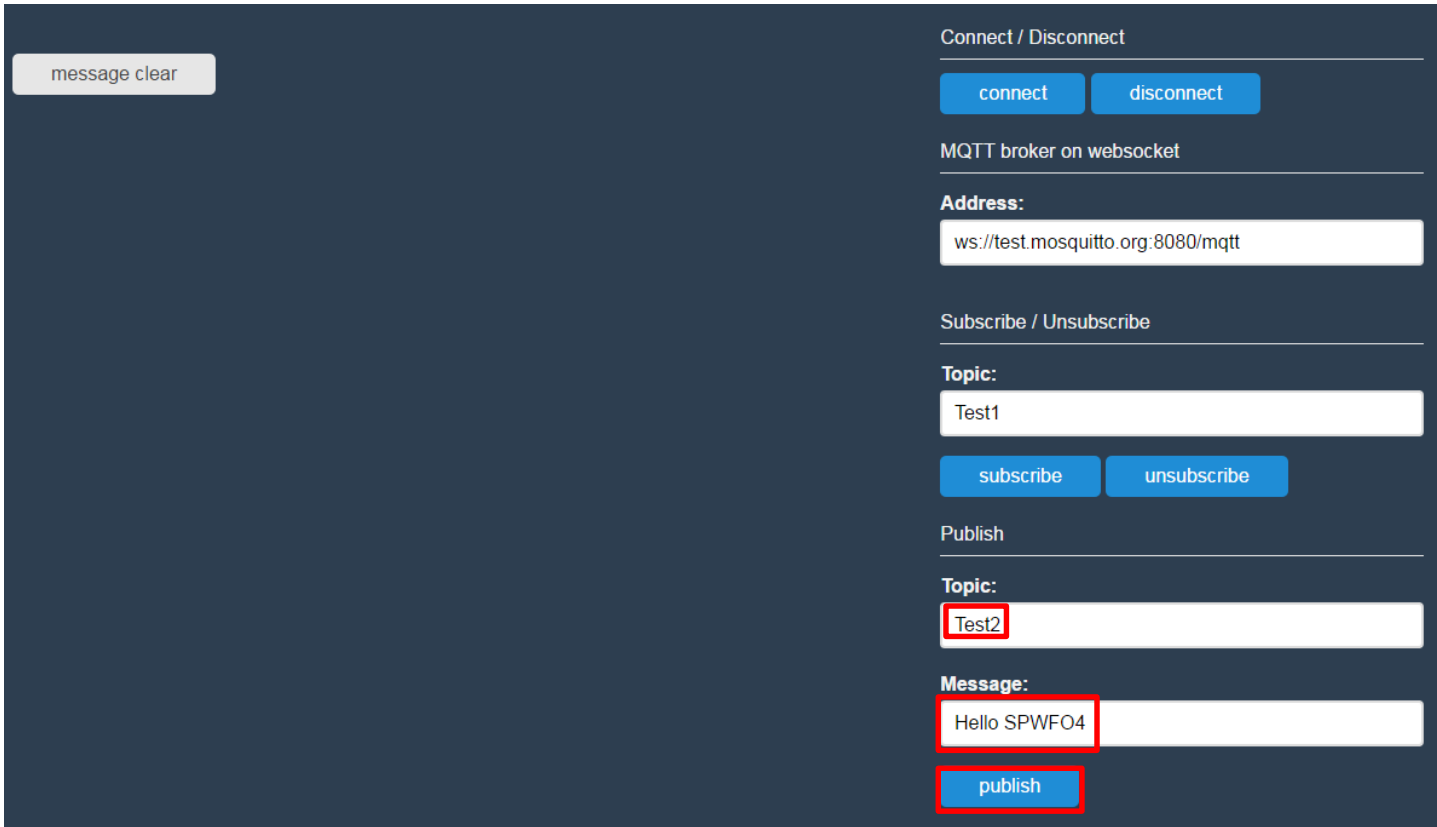
Tera Term output

AT+S.MQTTUNSUB=,Test2
AT-S.OK

Lab 10 : MQTT

Now if you publish something on the topic Test2 :

- Nothing happen on Tera Term



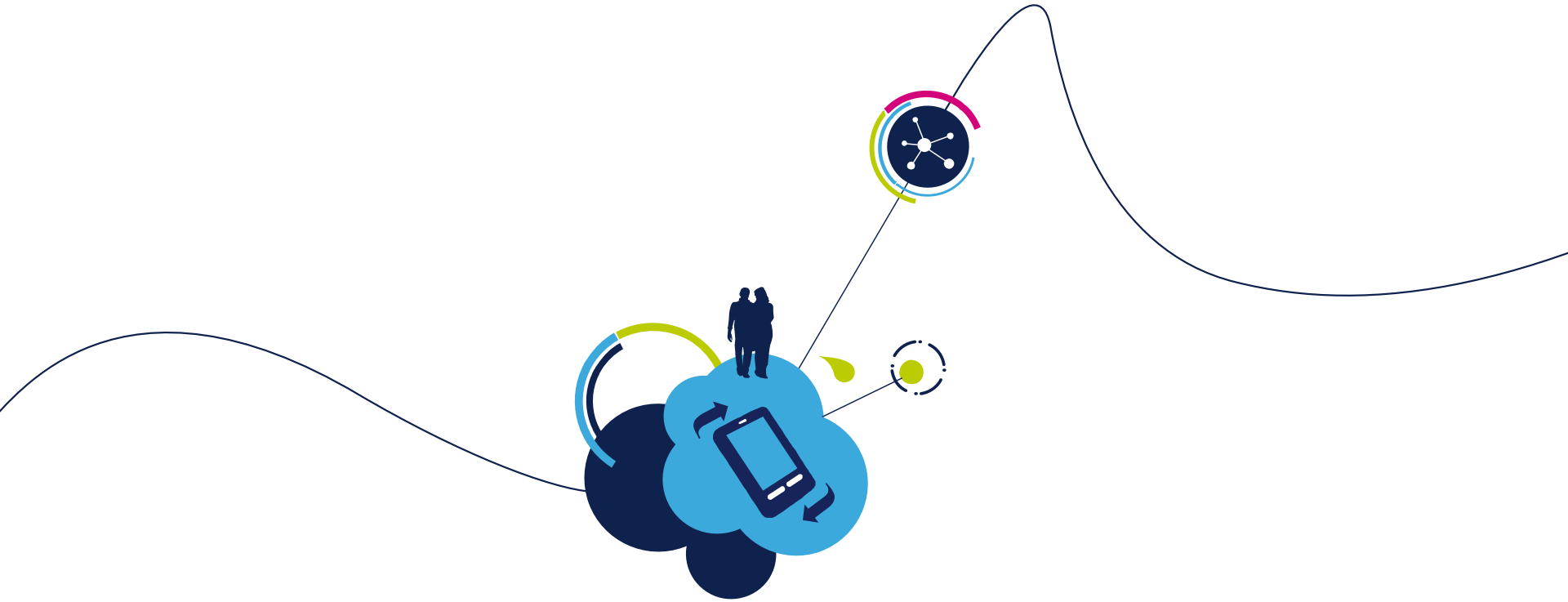
Lab 10 : MQTT

The **AT+S.MQTTDISC** is used to disconnect from an MQTT Broker .

- Syntax
  - **AT+S.MQTTDISC=**0<cr>

Tera Term output

```
AT+S.MQTTDISC=0
AT-S.OK
+WIND:87:MQTT Closed:1:0
```



Proceed to the next LAB!

Lab 11 : TFTP

- Objective
  - Get a file from a server or send a file to it through the SPWF04
  - Get a file from the SPWF04 or send a file to it through a device
- Prerequisites
  - Module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

Lab 11.1 : TFTP Client

- Objective
  - Get a file from a server through the SPWF04
  - Send a file to a server through the SPWF04
- Prerequisites
  - Module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

Lab 11.1 : TFTP Client

The stack implements the TFTP client protocol to transfer files on a UDP port. The command Put a request to a specified TFTP Server

- Syntax
  - **AT+S.TFTPPUT=**<hostname>,[<port>],<local_filename><cr>
- Configuration parameters
  - <Hostname> DNS resolvable name or IP address of the TFTP remote server
  - <port> Default: 69. Socket UDP port.
  - <local_Filename> filename to send to the remote host.

Tera Term output

AT+S.TFTPPUT=<hostname>,[<port>],<local_filename>
AT-S.OK

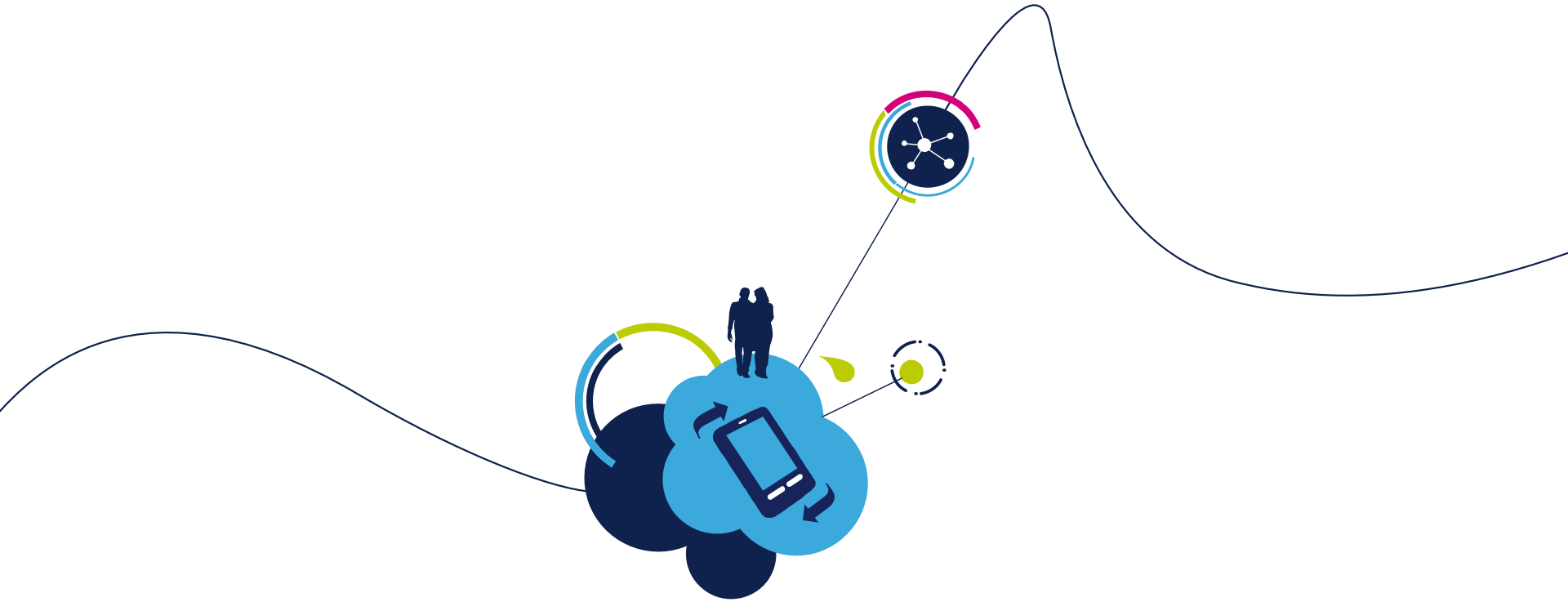
Lab 11.1 : TFTP Client

The stack implements the TFTP client protocol to transfer files on a UDP port. The command Get a request to a specified TFTP Server.

- Syntax
  - **AT+S.TFTPGET=**<hostname>,[<port>],<filename>,<local_filename><cr>
- Configuration parameters
  - <Hostname> DNS resolvable name or IP address of the TFTP remote server
  - <port> Default:69. Socket UDP port.
  - <Filename> filename to get from the remote host. It contains the complete path.
  - <local_filename> Default:2:<Filename> . Filename used locally.

Tera Term output

AT+S.TFTPGET=<hostname>,[<port>],<filename>,<local_filename>
AT-S.OK



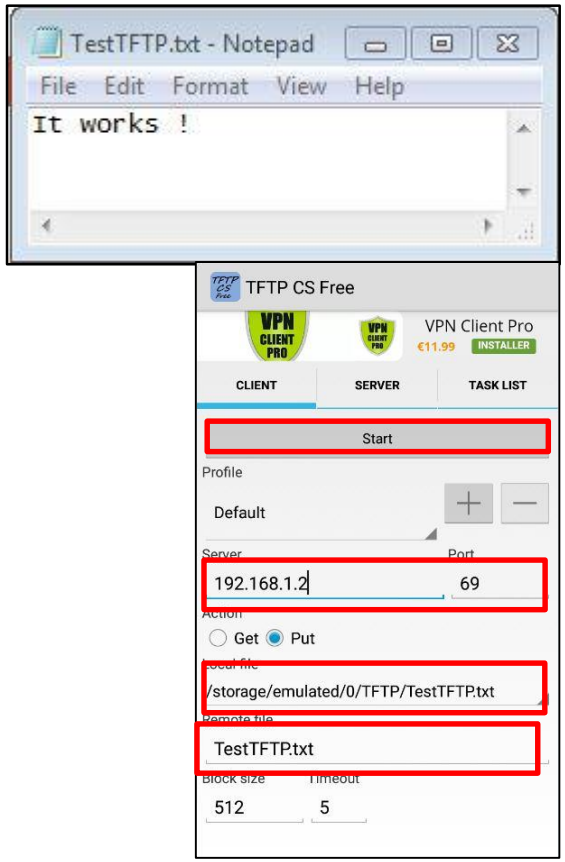
Proceed to the next LAB!

Lab 11.2 : TFTP Server

- Objective
  - Get a file from the SPWF04 through a device
  - Send a file to the SPWF04 through a device
- Prerequisites
  - Module connected to the AP (Lab 4.2) or module configured in MiniAP (Lab 4.1) mode

Lab 11.2 : TFTP Server Move file to internal FS

For this part, we will use a smartphone with TFTP CS Free App.
Smartphone and SPWF04 need to be connected to the same network.

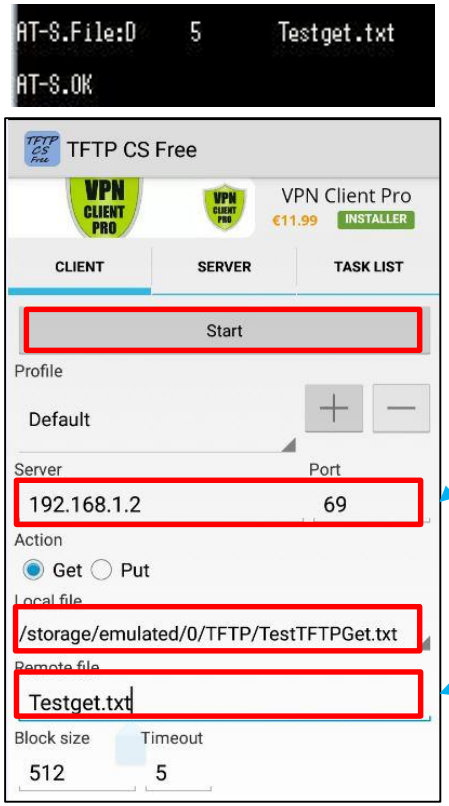


- **Create a file in your smartphone**
- **Enter the IP adress of the SPWF04 on your smartphone and the port on the TFTP App**
- **If you want to put a file to the SPWF04, choose « Put » and select the file you want to send**
- **Enter the name taken by the file once it will be inside the module**
- **Click the «Start» Button**

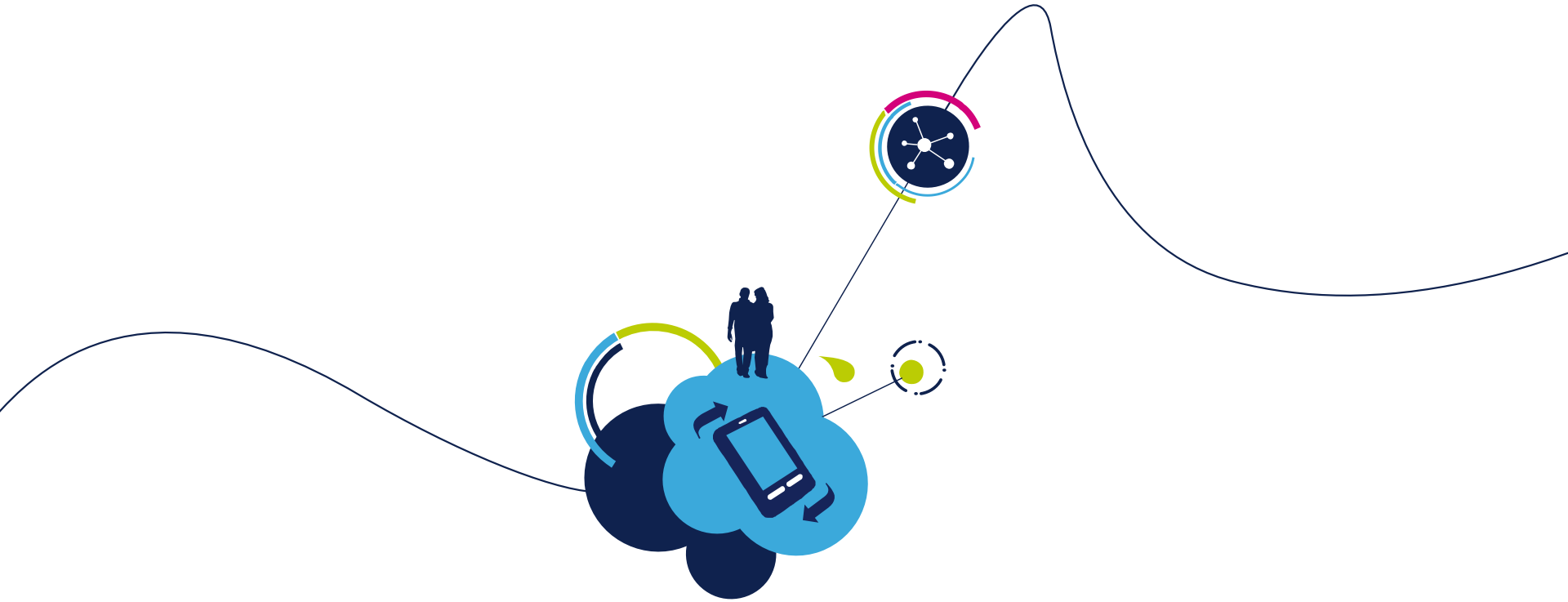
Tera Term output

+WIND:90:TFTP File Received:192.168.1.4:TestTFTP.txt

Lab 11.2 : TFTP Server Download file from internal FS



- Create a file in the SPWF04 File System
- Enter the IP adress of the SPWF04 on your smartphone and the port on the TFTP App
- If you want to get a file from the SPWF04 File System, choose « Get » and select the file you want to receive
- Enter the name taken by the file once it will be inside your smartphone
- Click the «Start» Button



Proceed to the next LAB!

Lab 12: ADC

- Objective
  - Know the voltage value of a GPIO
- Prerequisites
  - Work alone

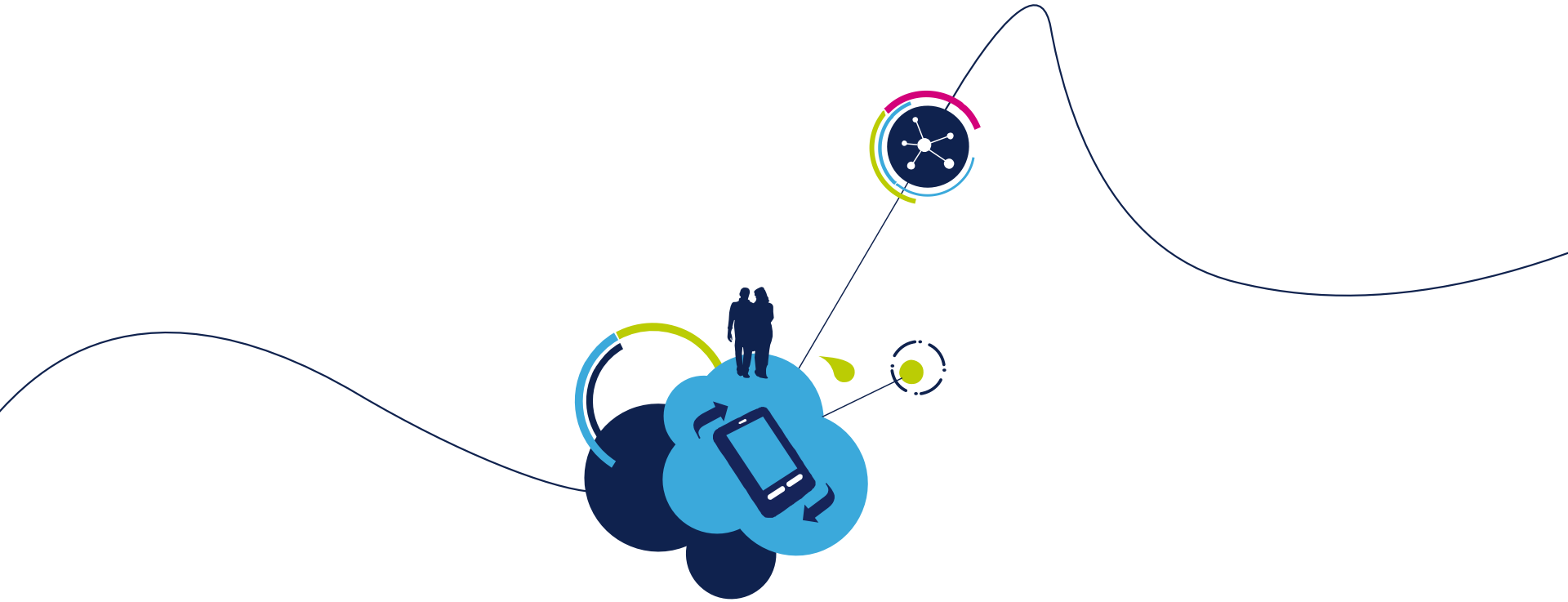
Lab 12 : ADC

The ADC command returns voltage value on selected GPIO. The value range is between 0 and 3300 mV, with a measurement accuracy of 10mV.

- Syntax
  - **AT+S.ADC=**<num><cr>
- Configuration parameters
- <num> : specifies the GPIO to be used for conversion. Available GPIOs are 0, 1 and 16

Tera Term output

AT+S.ADC=<num>
AT-S.Value:1636
AT-S.OK



Proceed to the next LAB!

Lab 13 : Low power modes

- Objective
  - Practice with Radio Power Save
  - Sleep Mode
  - Standby Mode
- Prerequisites
  - module connected to the AP

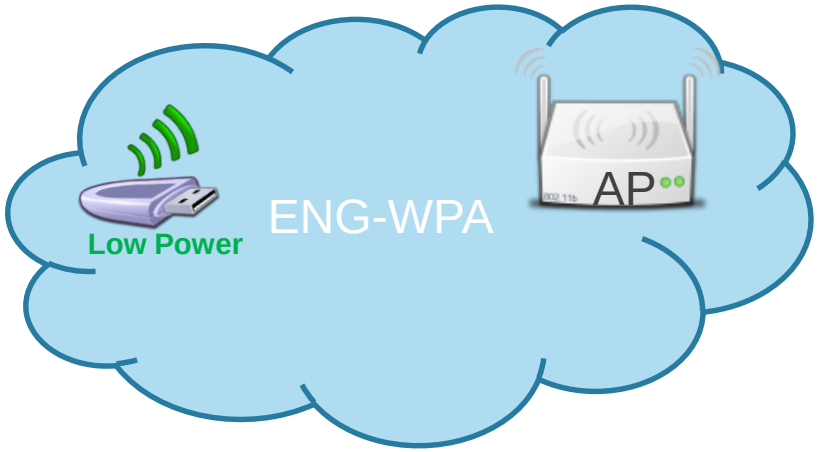
Lab 13 : Low power modes

This feature allows to enable the low power states.

The module supports the “Radio Power Save” mode, the “Sleep” mode and the “Standby” mode.

The module should be connected to the AP (as shown in Lab 3) in order to use the “Radio Power Save” mode.

Please refer to “SPWF04S Power Management Application Notes” for more details.



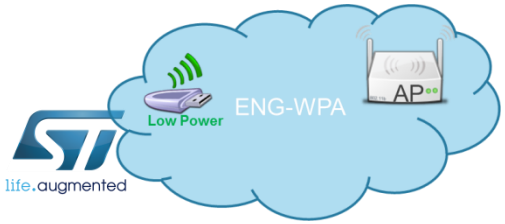
Lab 13 : Low power modes – Power Save

Module Power State	STM32	WLAN
Standby	Standby	Off
Sleep	Stop	PS or Fast-PS
Power Save	Run	PS or Fast-PS
Active Rx	Run	Rx Idle / Rx Active
Active Tx	Run	Tx Active

Lab 13 : Low power modes – Power Save

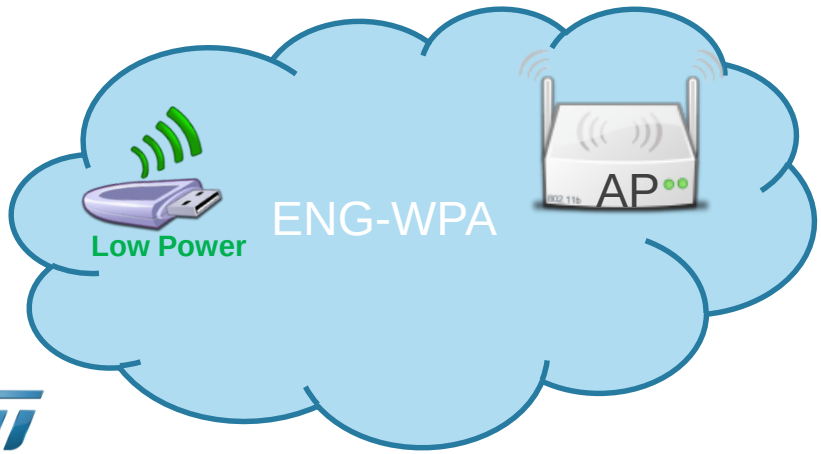
- By default, the module starts in ACTIVE mode.
- Enable the Power Save Mode:
  - Type **AT+S.WIFI=0**
  - Type **AT+S.SCFG=wifi_powersave,1** (“wifi_powersave,2” enables the Fast-PS mode)
- Enable the doze operational mode:
  - Type **AT+S.SCFG=wifi_operational_mode,11** (“wifi_operational_mode,12” enables the quiescent mode)
- Choose the wake up mode:
  1. Wake up every n. beacon (specified in the wifi_beacon_wakeup variable)
     - Type **AT+S.SCFG=wifi_listen_interval,0**
     - Type **AT+S.SCFG=wifi_beacon_wakeup,1**

     OR
  2. Wake up every n. beacon adaptively (specified in the wifi_beacon_wakeup variable)
     - Type **AT+S.SCFG=wifi_listen_interval,1**
     - Type **AT+S.SCFG=wifi_beacon_wakeup,1**



Lab 13 : Low power modes – Power Save

- Save the settings on the flash memory and reset the module
  - Type **AT+S.WIFI=1**
- The WIND:66 message related to Low Power Mode will be displayed



Tera Term output

```
AT+S.WIFI=1
AT-S.OK
+WIND:1:Poweron:xxxxxxx-yyyyyyy-SPWF04S
+WIND:13:Copyright (c) 2012-2017 STMicroelectronics,Inc.
All rights Reserved: SPWF04Sx
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join:02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'IoT' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up:192.168.1.104
+WIND:66:Low Power mode:1
```

Lab 13 : Low power modes – Sleep Mode

Module Power State	STM32	WLAN
Standby	Standby	Off
Sleep	Stop	PS or Fast-PS
Power Save	Run	PS or Fast-PS
Active Rx	Run	Rx Idle / Rx Active
Active Tx	Run	Tx Active

Lab 13 : Low power modes – Sleep Mode

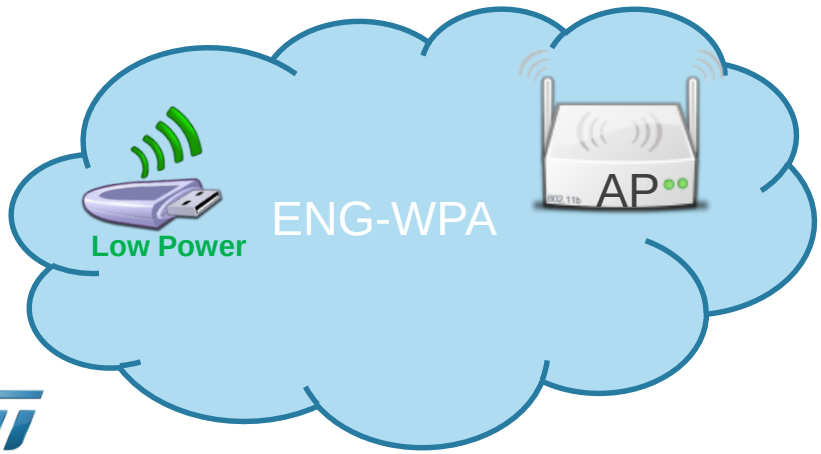
- In the Sleep Mode, the core STM32 is stopped
- Enable the Sleep Mode:
  - Type **AT+S.WIFI=0**
  - Type **AT+S.SCFG=sleep_enabled,1**
- Enable the Power Save Mode:
  - Type **AT+S.SCFG=wifi_powersave,1** (“wifi_powersave,2” enables the Fast-PS mode)
- Enable the doze operational mode:
  - Type **AT+S.SCFG=wifi_operational_mode,11** (“wifi_operational_mode,12” enables the quiescent mode)
- Choose the wake up mode:
  1. Wake up every n. beacon (specified in the wifi_beacon_wakeup variable)
     - Type **AT+S.SCFG=wifi_listen_interval,0**
     - Type **AT+S.SCFG=wifi_beacon_wakeup,1**

     OR
  2. Wake up every n. beacon adaptively (specified in the wifi_beacon_wakeup variable)
     - Type **at+s.scfg=wifi_listen_interval,1**
     - Type **at+s.scfg=wifi_beacon_wakeup,1**

Lab 13 : Low power modes – Sleep Mode

- Save the settings on the flash memory and reset the module
  - Type **AT+S.WIFI=1**

- The WIND:69 message related to Sleep Mode will be displayed

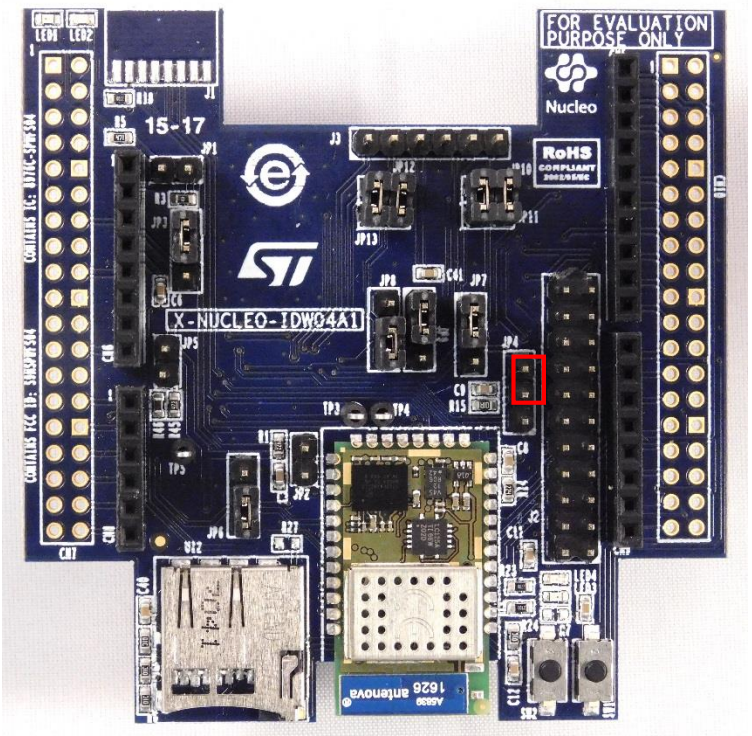


Tera Term output

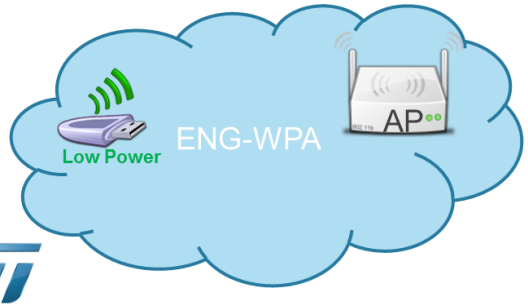
```
AT+S.WIFI=1
AT-S.OK
+WIND:1:Poweron:xxxxxxx-yyyyyyy-SPWF04S
+WIND:13: Copyright (c) 2012-2017 STMicroelectronics,
Inc.  All rights Reserved.
+WIND:3:Watchdog Running
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:19:WiFi Join:02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'IoT' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up:192.168.1.104
+WIND:66:Low Power mode:1
+WIND:69:Going into DeepSleep
+WIND:69:Resuming from DeepSleep
```

Lab 13 : Low power modes – Sleep Mode

- Wake STM32 up using the GPIO6
  - Put the GPIO6 to 3.3V (jumper on JP4 as in the picture)



- The WIND messages will be displayed



Tera Term output

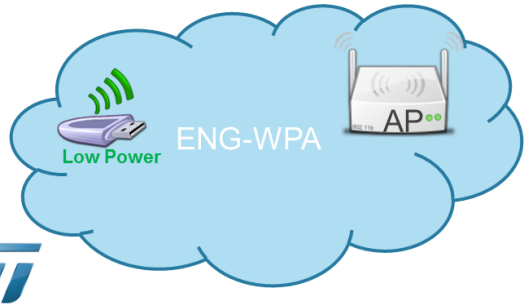
+WIND:53:Wakeup
+WIND:70:Resuming from DeepSleep

life.augmented

Lab 13 : Low power modes – Sleep Mode

- Put STM32 in sleep mode using the GPIO6
  - Go back the GPIO6 floating

The WIND:69 message will be displayed

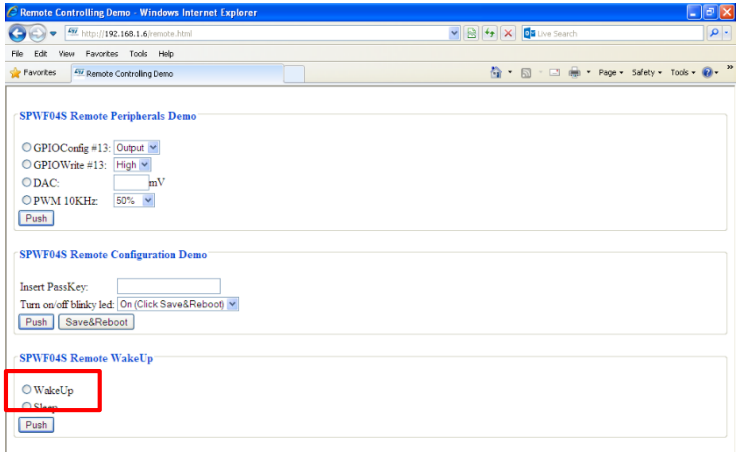


Tera Term output

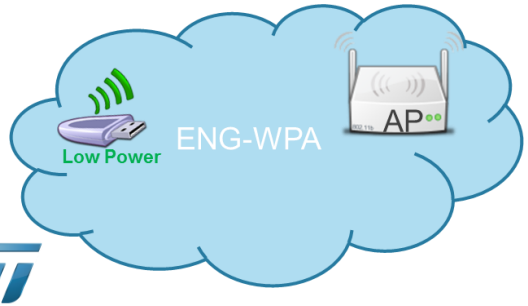
```
+WIND:69:Going into DeepSleep
+WIND:70:Resuming from DeepSleep
```

Lab 13 : Low power modes – Sleep Mode

- Wake STM32 up using the remote page
  - **Connect a device to the same module's network**
  - **Open the following link: http://[module_IP_address]/remote.html**
  - **Select "WakeUp" and click on "Push" button to wake up the module**



- The WIND message will be displayed



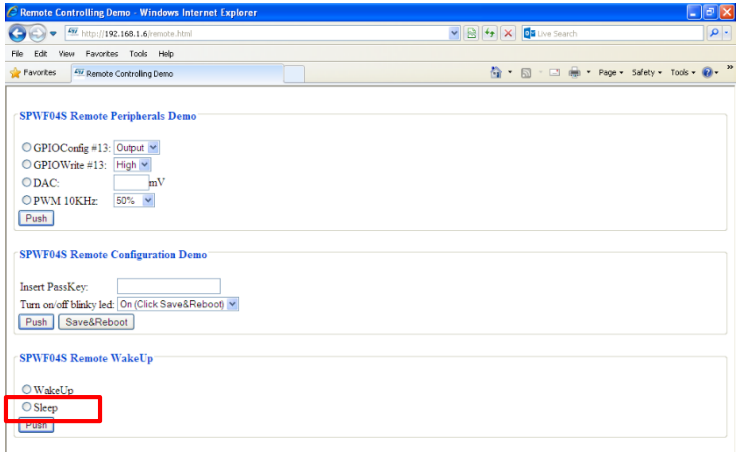
Tera Term output

+WIND:70:Resuming from DeepSleep

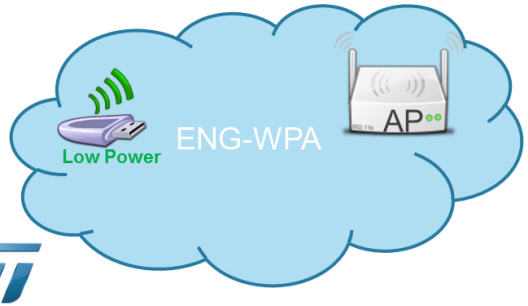
life.augmented

Lab 13 : Low power modes – Sleep Mode

- Put STM32 in sleep mode using the remote page
  - **Connect a device to the same module's network**
  - **Open the following link: http://[module_IP_address]/remote.html**
  - **Select "Sleep" and click on "Push" button to put the module in sleep mode**



- The WIND message will be displayed



Tera Term output

+WIND:69:Going into DeepSleep

life.augmented

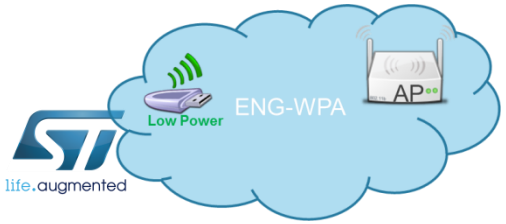
Lab 13 : Low power modes – Standby Mode

Module Power State	STM32	WLAN
Standby	Standby	Off
Sleep	Stop	PS or Fast-PS
Power Save	Run	PS or Fast-PS
Active Rx	Run	Rx Idle / Rx Active
Active Tx	Run	Tx Active

Lab 13 : Low power modes – Standby Mode

- During the Standby Mode, both the STM32 and the Radio will be put in standby mode. The standby mode allows to achieve the lowest power consumption.

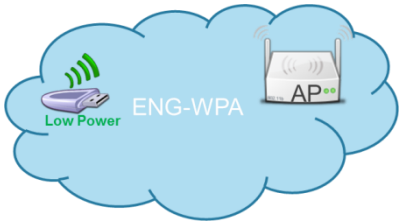
- Enable the Standby Mode:
  - Type **AT+S.WIFI=0**
  - Type **AT+S.SCFG=sleep_enabled,0 (Sleep mode must be disabled)**
  - Type **AT+S.SCFG=standby_enabled,1**
  - Please be sure that GPIO6 isn't forced high
- Set the standby time to wake up via RTC alarm:
  - Type **AT+S.SCFG=standby_time,15**



Tip: The Wifi Standby Mode can also be quickly enabled using the command: AT+S.PMS=3

Lab 13 : Low power modes – Standby Mode

- Save the settings on the flash memory and reset the module
  - Type **AT+S.WIFI=1**
  - Type **AT+S.WCFG**
  - Type **AT+S.RESET**
- After 15 seconds, the module will be rebooted and will return in the ACTIVE state.

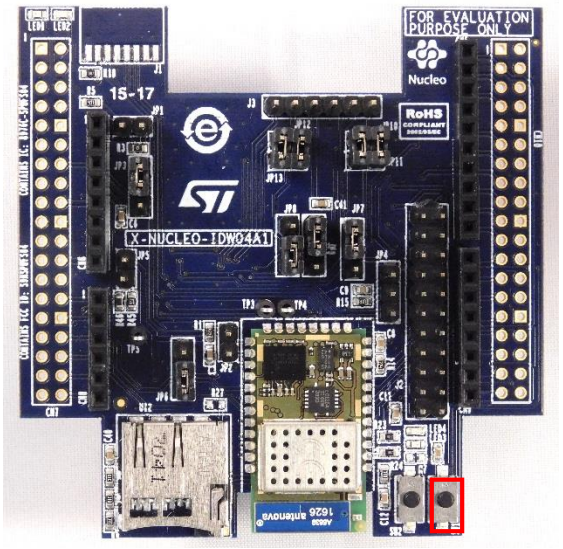


Tera Term output

```
AT+S.WIFI=1
AT-S.OK
AT+S.WCFG
AT-S.OK
AT+S.RESET
+WIND:2:Reset
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:38:WiFi: Powered Down
+WIND:67:Going into Standby:15
+WIND:68:Resuming from Standby
+WIND:1:Poweron:xxxxxxx-yyyyyyy-SPWF04S
+WIND:13:Copyright (c) 2012-2017
STMicroelectronics, Inc.  All rights
Reserved:SPWF04Sx
+WIND:0:Console active
+WIND:3:Watchdog Running:20
+WIND:32:WiFi Hardware Started
+WIND:19:WiFi Join:02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'IoT' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up:192.168.0.1
```

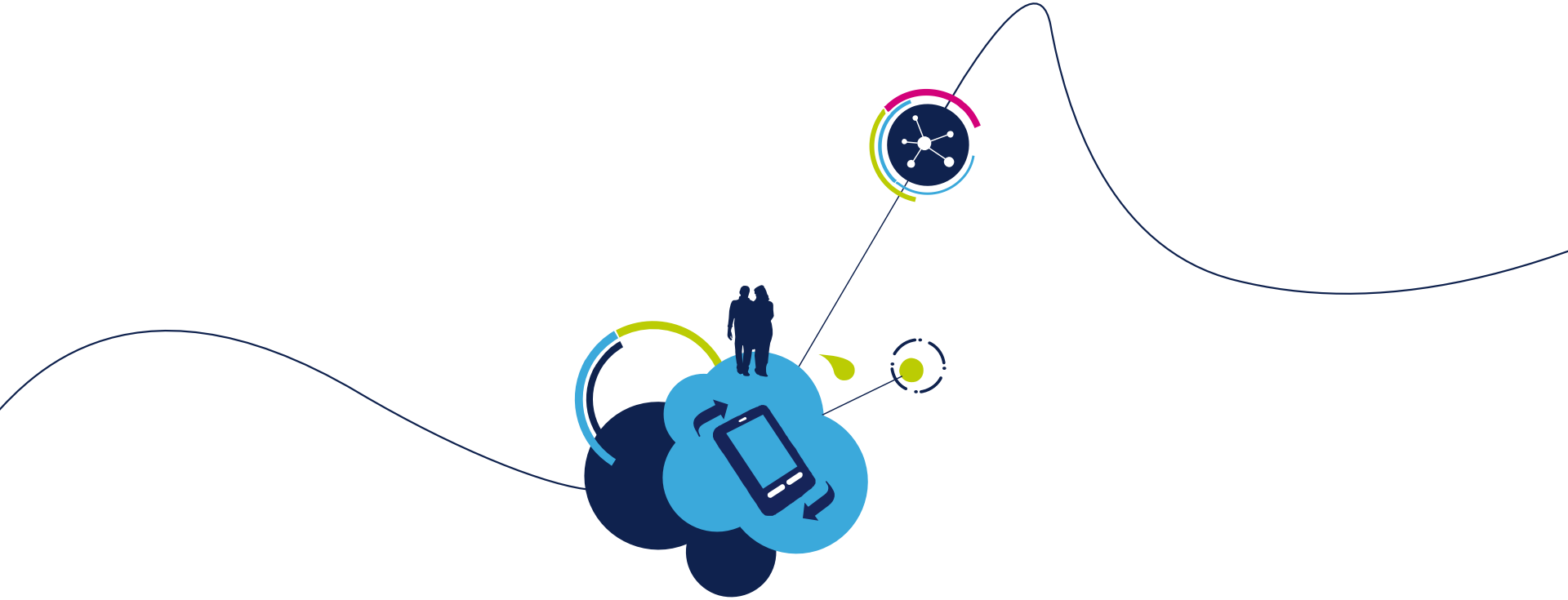
Lab 13 : Low power modes – Standby Mode

- Wake up the module using the GPIO6
- Enable the Standby Mode:
  - Type **AT+S.WIFI=0**
  - Type **AT+S.SCFG=sleep_enabled,0 (make sure that Sleep mode is disabled)**
  - Type **AT+S.SCFG=standby_enabled,1**
  - Type **AT+S.WIFI=1**
  - Type **AT+S.WCFG**
  - Type **AT+S.RESET**
- Put the GPIO6 to 3.3V to wake up the module
- The WIND message will be displayed and the module will return in ACTIVE state



Tera Term output

```
AT+S.WIFI=0
AT-S.OK
AT+S.SCFG=sleep_enabled,0
AT-S.OK
AT+S.SCFG=standby_enabled,1
AT-S.OK
AT+S.WIFI=1
AT-S.OK
AT+S.WCFG
AT-S.OK
AT+S.RESET
+WIND:2:Reset
+WIND:68:Resuming from Standby
+WIND:1:Poweron:xxxxxxx-yyyyyyy-SPWF04S
……
```



THANK YOU!