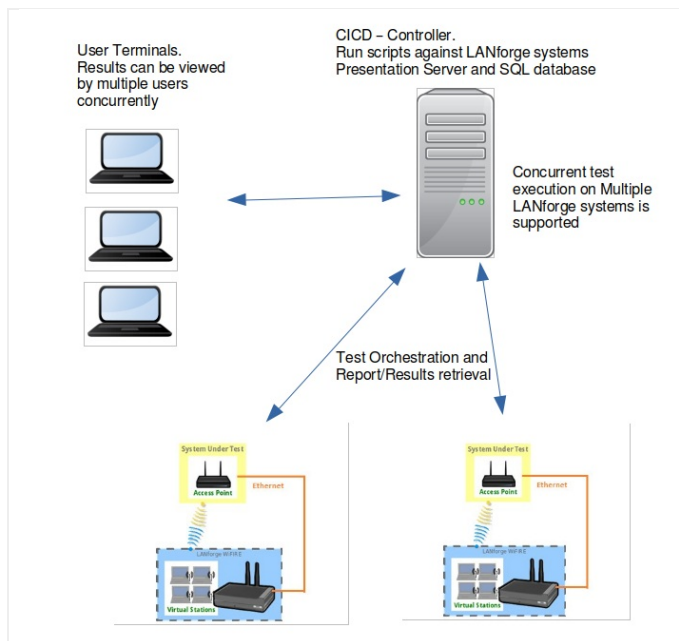


## Basic CICD AP Testing with LANforge

**Goal:** Set up Basic CICD a LANforge system, Regression Automation and Reporting with data from previous runs.

The LANforge CICD framework provides an ability to execute a suite of tests and report results.



1. The following steps are discussed
  - A. Set Up CICD Controller and Environment
  - B. Set Up The JSON Configuration Files
  - C. Test Execution
  - D. Test Results

### 2. Set Up CICD Controller and Environment

- A. clone lanforge-scripts from <https://github.com/greearb/lanforge-scripts>
- B. run `/lanforge-scripts/py-scripts/update_dependencies.py` to install python packages for generating output
- C. Install web server:  
The web server is to allow for viewing of results from User Terminals  
The CICD - Controller is not dependent on a web server, results may be viewed locally on CICD - Controller
  - A. **LANforge** LANforge installation using `kininstall.pl` installs a web server on LANforge  
LANforge installation installs an `httpd` server, LANforge may be used for storing and displaying results.  
For the following example a separate LANforge system (Fedora) was used as the CICD - Controller and `httpd` web server.

```
$ sudo dnf install httpd
```

- B. **Fedora** install `httpd` and configure server

```
$ sudo apt install apache2
```

- C. **Ubuntu** install `apache2` and configure server

- D. Install mail service for email of links to results  
For the example below Linux `mailx` program was used  
Installation of mail services is dependent on the environment in which the CICD - Controller is installed.  
The CICD - Controller is not dependent on email services

- E. Install database `sqlite3`

```
$ sudo dnf install sqlite3
```

- A. **Fedora**

```
$ sudo apt-get update
```

- B. **Ubuntu**

```
$ sudo apt-get install sqlite3
```

- F. Create a `html-reports` directory. On lanforge `/home/lanforge/html-reports`
- G. Determine `sqlite3` database name and location, `sqlite3` db will be created. `./tools/qa_sqlite3.db`

### 3. Set Up The JSON Configuration Files

- A. There are three JSON configuration input files described below. For all the JSON configuration files the CAPITALIZED parameters allow for a value to be entered into one location and used in multiple areas of the CICD framework. For example in `ssid_idx=1` the `SSID_USED` is set to `asus11ax-5`. For the test suite below the `SSID_USED` may be entered instead of `asus11ax-5`, thus if the SSID changes, the SSID will need to be modified in `ct_AX88U_dut`, the `ct_tests.json` will remain untouched. This reduces the need to modify the `ct_test.json` for SSID changes that would affect multiple tests
- A. `--json_rig test_rig.json` this JSON file describes LANforge test rig, [Example ct\\_test\\_rig.json](#)  
The `test_rig.json` describes the LANforge system and test parameters for the CICD - Controller
- B. `--json_dut ct_AX88U_dut.json` this JSON file describes the AP, [Example ct\\_AX88U\\_dut.json](#)  
the `ct_AX88U_dut.json` describes the device under test parameters, `DUT_SET_NAME`: `DUT_NAME ASUSRT-AX88U` for example is used by Chamberview Tests
- C. `--json_test ct_tests.json` this JSON file describes the tests, [Example ct\\_tests.json](#)  
The tests may use the CAPITALIZED variables or may be entered with the command line arguments as they would be entered on the command line.  
The tests are not limited to only python tests

B. **test\_rig.json**

```
{
  "test_rig":{
    "Notes":[
      "This JSON file describes LANforge system and test run configuration"
    ]
  },
  "test_rig_parameters":{
    "TEST_BED": "CT-TEST-001",
    "TEST_RIG": "CT-TEST-001",
    "DATABASE_SQLITE": ". /tools/qa_sqlite3.db",
    "LF_MGR_IP": "192.168.100.116",
    "LF_MGR_PORT": "8080",
    "LF_MGR_USER": "lanforge",
    "LF_MGR_PASS": "lanforge",
    "UPSTREAM_PORT": "1.1.eth2",
    "TEST_TIMEOUT": 600,
    "EMAIL_LIST_PRODUCTION": "support@candelatech.com",
    "EMAIL_LIST_TEST": "support@candelatech.com",
    "EMAIL_TITLE_TXT": "Lanforge QA Testing",
    "EMAIL_TXT": "Lanforge QA Testing"
  }
}
```

C. **ct\_AX88U\_dut.json**

```
{
  "ct_AX88U_dut":{
    "Notes":[
      "The device undertest configuration is contained in this file"
    ]
  },
  "test_dut":{
    "DUT_SET_NAME": "DUT_NAME ASUSRT-AX88U",
    "USE_DUT_NAME": "ASUSRT-AX88U",
    "wireless_network_dict":{
      "ssid_idx=0":{"ssid_idx":"0", "SSID_USED":"asus11ax-2", "SSID_PW_USED":"hello123", "BSSID":"3c:7c:3f:55:4d:60", "SE
      "ssid_idx=1":{"ssid_idx":"1", "SSID_USED":"asus11ax-5", "SSID_PW_USED":"hello123", "BSSID":"3c:7c:3f:55:4d:64", "SE
    }
  }
}
```

D. **ct\_tests.json**

```
{
  "ct_tests_001":{
    "Notes":[
      "This JSON file describes tests to be run by LANforge system"
    ]
  },
  "test_suites":{
    "suite_wc":{
      "create_chamberview_dut_wc":{
        "enabled":"TRUE",
        "load_db":"skip",
        "command":"create_chamberview_dut.py",
        "args": "",
        "args_list":[
          "--lfmgr LF_MGR_IP --port LF_MGR_PORT --dut_name DUT_NAME",
          "--ssid 'ssid_idx=0 ssid=SSID_USED security=SECURITY_USED password=SSID_PW_USED bssid=BSSID'",
          "--ssid 'ssid_idx=1 ssid=SSID_USED security=SECURITY_USED password=SSID_PW_USED bssid=BSSID'",
          "--sw_version DUT_SW --hw_version DUT_HW --serial_num DUT_SERIAL --model_num DUT_NAME"
        ]
      },
      "create_chamberview_wc":{
        "enabled":"TRUE",
        "load_db":"skip",
        "command":"create_chamberview.py",
        "args": "",
        "args_list":[
          "--lfmgr LF_MGR_IP --port LF_MGR_PORT --delete_scenario",
          "--create_scenario scenario_wpa2_wc",
          "--raw_line \"profile_link 1.1 STA-AC 19 'DUT: DUT_NAME Radio-1' NA wiphy7,AUTO -1 NA\" ",
          "--raw_line \"profile_link 1.1 upstream-dhcp 1 NA NA UPSTREAM_PORT,AUTO -1 NA\""
        ]
      },
      "wifi_capacity":{
        "enabled":"TRUE",
        "timeout":"600",
        "iterations":"1",
        "load_db":"skip",
        "command":"lf_wifi_capacity_test.py",
        "args": "",
        "args_list":[
          "--mgr LF_MGR_IP --port LF_MGR_PORT --lf_user LF_MGR_USER --lf_password LF_MGR_PASS --instance_name sc",
          "--upstream UPSTREAM_PORT --batch_size 1,10,19 --loop_iter 1 --protocol UDP-IPv4 --duration 6000",
          "--pull_report --local_lf_report_dir REPORT_PATH --test_tag 'wpa2_wc'",
          "--test_rig TEST_RIG",
          "--set DUT_SET_NAME"
        ]
      },
      "lf_qa":{
        "enabled":"TRUE",
        "timeout":"600",
        "load_db":"skip",
        "command":"./tools/lf_qa.py",
        "args": "",
        "args_list":[
          "--path REPORT_PATH --store --png --database DATABASE_SQLITE"
        ]
      }
    }
  },
  "lf_qa":{
    "enabled":"TRUE",
    "timeout":"600",
    "load_db":"skip",
    "command":"./tools/lf_qa.py",
    "args": "",
    "args_list":[
      "--path REPORT_PATH --store --png --database DATABASE_SQLITE"
    ]
  }
}
```

E. **sample command with above data:**

```
./lf_check.py --json_rig ct_test_rig.json \
--json_dut ct_AX88U_dut.json \
--json_test ct_tests.json \
--suite "suite_wc" \
--path '/home/lanforge/html-reports/ct_results_directory'
```

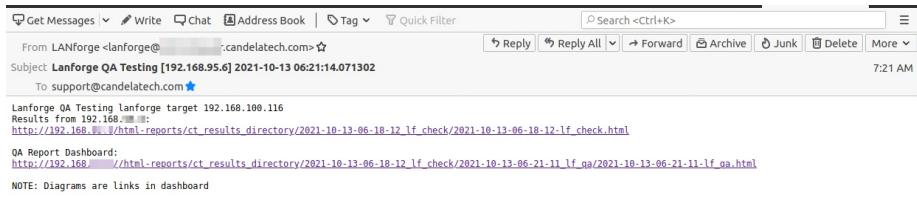
#### 4. Set Up The JSON Configuration Files

- The lf\_check.py is run from the lanforge-scripts/py-scripts/tools directory
- lf\_check.py uses three JSON files as input:  
For Example:  
**ct\_test\_rig.json** - describes the LANforge test rig configuration  
**ct\_AX88U\_dut.json** - describes the device under test  
**ct\_tests.json** - describe the tests to be run.

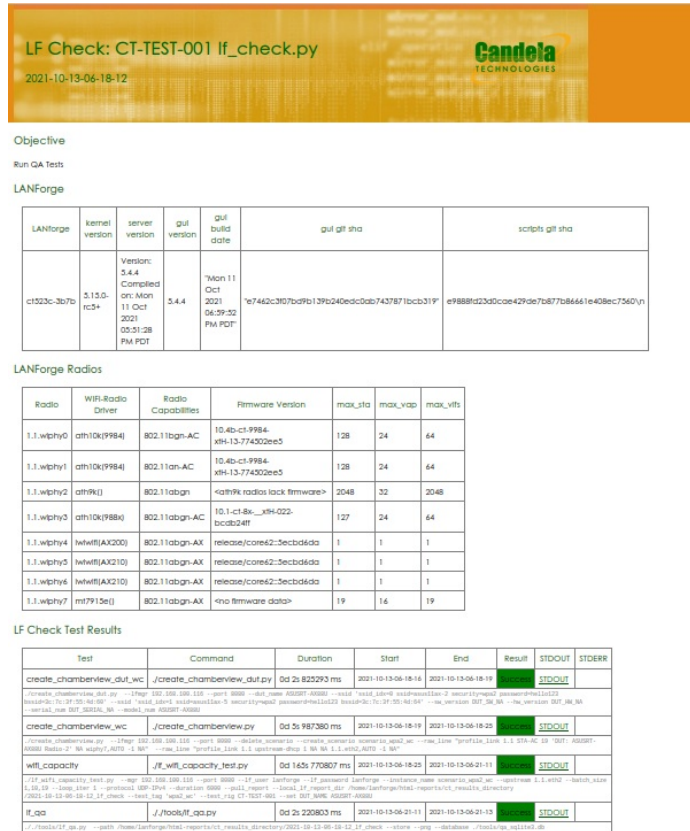
#### 5. lf\_check.py execution, simple command example

A. **./lf\_check.py --json\_rig ct\_test\_rig.json \**  
**--json\_dut ct\_AX88U\_dut.json \**  
**--json\_test ct\_tests.json \**  
**--suite "suite\_wc" \**  
**--path '/home/lanforge/html-reports/ct\_results\_directory'**

#### 6. Sample email sent on run



## 7. lf\_check.py: sample lf\_check.py Report



## 8. lf\_qa.py

lf\_qa.py: process kpi.csv, produces html/pdf results, produces plotly png and interactive graphs from test run kpi

sample command:

```
./lf_qa.py --path /home/lanforge/html-reports/ct_results_directory/(results dir of lf_check.py) \
--store \
--png \
--database ./tools/qa_aqlite3.db
```

## 9. lf\_qa.py: sample lf\_qa.py Report

## Objective

QA Verification

## Device Under Test

| DUT          | SW version | HW version | SN |
|--------------|------------|------------|----|
| ASUSRT-AX88U | DUT_SW_NA  | DUT_HW_NA  | NA |

Test Rig: CT-TEST-001 Links

[PDF Report](#)  
[Current Test Suite Results Directory](#)  
[All Test Rigs Test Suite Results Directory](#)

## Test Suite

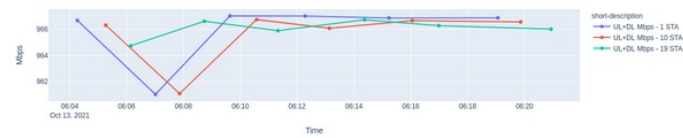
| Test          | Test_Tag | Links                                      |
|---------------|----------|--------------------------------------------|
| WiFi Capacity | wpa2_wc  | <a href="#">html</a> / <a href="#">pdf</a> |

## Suite Summary



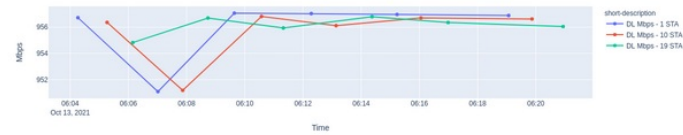
## QA Test Results

WiFi Capacity : Per Stations Rate UL+DL : wpa2\_wc : CT-TEST-001



[WiFi Capacity : Per Stations Rate UL+DL : wpa2\\_wc : CT-TEST-001 Report](#)

WiFi Capacity : Per Stations Rate DL : wpa2\_wc : CT-TEST-001



[WiFi Capacity : Per Stations Rate DL : wpa2\\_wc : CT-TEST-001 Report](#)

## 10. Sample If\_check.py Output example If\_check Report

### 11. Test Control Inputs in Test Suite JSON

- "enabled": "TRUE"**  
Allows for individual test enable and disable of the test.
- "load\_db": "CUSTOM\_DATABASE"**  
Allows for loading a LANforge database prior to the test run.
- "timeout": "300"**  
Allows for test to have individual timeout other than default.
- "iterations": "2"**  
Allows for test to run multiple iterations.