# UniFi - USG Advanced Configuration Using config.gateway.json

This article describes how to perform advanced configurations on the UniFi Security Gateway (USG and USG-PRO-4) using the **config.gateway.json** file.

This article is **not applicable** to the UniFi Dream Machine models, because all configurations are already available in the UniFi Network user interface.

**Notes & Requirements:**

- Ubiquiti Support cannot assist in the creation of the **config.gateway.json** file nor will assistance be provided for command line configuration. If assistance is required, feel free to visit our Community to create a topic and ask for help with your desired configuration.
- This article covers advanced configuration, and should only be used by advanced users.

## Overview

The **config.gateway.json** is a file that sits in the UniFi Network filesystem and allows custom changes to the USG. Some possible customizations include: configuring site-to-site VPNs with hostnames, policy routing of certain traffic out WAN2, or even adding multiple IP addresses on an interface.

When making changes via the **config.gateway.json** file, it is best to enter only the customizations that *cannot* be performed via the Network application. If the formatting is incorrect, a provisioning loop will be triggered on the USG, and a reboot will take place once the USG comes out of the provisioning loop. At this point the **config.gateway.json** file could be corrected or removed to correct this.

***Warning:*** *Some users may find they can get away with using the full config, but this is not recommended as it will most likely cause issues down the road. A provisioning loop might take place when a setting is changed in the Network application that conflicts with a setting in the* ***config.gateway.json*** *file.*

---

### Creating the File

By default, the **config.gateway.json** file doesn't exist. It has to be created in order to use it.

1. Create a new file using a text editor, such as TextEdit or Notepad++.
2. The structure of a json file is just as important as the words themselves. Incorrect placement of brackets, indentations, line breaks or any other structural element will make the json file invalid. It is recommended to run the text through a json validator in order to verify it has the correct syntax. The JSON Formatter website is one example of the many options of json validators you'll find online.

3. Once the contents of the file has been validated, save it by naming it **config.gateway.json** and placing it under the *<unifi_base>/data/sites/site_ID* directory stored on the Network application.
    a. Depending on your operating system, placing the file under this directory might be as simple as drag and drop, or using a FTP server might be necessary. The **config.gateway.json** file must have unifi:unifi as the owner and group permissions. You can check to verify with *ls -l <unifi_base>/data/sites/site_ID*. To change it, once you're in the site directory, use the command:

```
chown unifi:unifi config.gateway.json
```

The location *<unifi_base>* will vary from one operating system to another. See this article for more information. The **site_ID** can be seen in the URL of your browser when on the Network application. The original site is named "default", and every site that is created will be assigned a random string. For example, this is what would be seen in the URL bar when inside the dashboard page of a site:

```
https://127.0.0.1:8443/manage/s/ceb1m27d/dashboard
```

In the above case, the random string **ceb1m27d** is the folder name that shall be used under *<unifi_base>/data/sites/.* Therefore, the **config.gateway.json** file should be placed inside *<unifi_base>/data/sites/ceb1m27d/.*

**User Tips:**

- On Cloud Key, the install path for the .json file is: */srv/unifi/data/sites/[site name/default]/*
- On an Ubuntu, the install path for the .json file is: */usr/lib/unifi/data/sites/[site name/default]/*

---

## Editing the File

Before customizing firewall or NAT rules, take note of the rule numbers used in the UniFi Network Firewall. Default firewall rules start at either 3001 or 6001, and NAT rules will also start at 6001 (which don't overlap with firewall rules). The custom rules created in the **config.gateway.json** file cannot have duplicate rule numbers with the existing rules in the USG, or there will be a provisioning loop. It is recommended to put custom rules before the existing ruleset, as the lower number will win between two matching rules.

***NOTE:*** *When editing this custom json file, it is not necessary to include everything. You must only include the complete "path" to the items you have edited, anything outside of the path can be omitted.*

Think of each node in the json file as a folder that is nested within other folders (except for the level 1 folder which is our main section). The folder path that takes you from level 1 all the way down to the item you will be configuring must be present in the json file. See this example where we want to edit **"close"**, which has the following path: *system > conntrack > timeout > tcp > close.*

```
{
    "system": { Level 1
        "conntrack": {  Level 2
            "modules": {
                "sip": "disable"
            },
            "timeout": { Level 3
                "icmp": 30,
                "other": 600,
                "tcp": { Level 4
                    "close": 10, Level 5
                    "close-wait": 60,
                    "established": 7440,
                    "fin-wait": 120,
                    "last-ack": 30,
                    "syn-recv": 60,
                    "syn-sent": 120,
                    "time-wait": 120
                },
                "udp": {
                    "other": 30,
                    "stream": 180
                }
            }
        }
    }
}
```

Notice that in level 3 "modules" is also present along with "timeout", but we will not include it in the json file because it is not part of the path to "close". Same with the other items in level 5 under "tcp". They do not need to appear in the **config.gateway.json** file because they are not part of the path. A successful change then, in the configuration of "close" from 10 to 20 would look like this:

```
{
    "system": {
        "conntrack": {
            "timeout": {
                "tcp": {
                    "close": 20
                }
            }
        }
    }
}
```

---

**Click here for an example of how a DNAT rule is created for DNS configured using EdgeOS formatting.**

---

## Testing & Verification

It's recommended to validate the code once finished creating the **config.gateway.json**. There are a number of free

options out there, jsonlint.com is a popular one.

After adding the **config.gateway.json** to the UniFi Network site of your choosing, you need to reboot the USG so that the file will provisioned appropriately. The startup process is expected to take 2 to 4 minutes. If it takes longer than that, there may be a formatting error in the config.gateway.json. You can check `server.log` in the application and search for `commit error`. You can usually find what went wrong with the provisioning of the newly customized configuration in the log files. Find information about that here.

**User Tip:** An easy way to test the validity of the json file is: python -m json.tool config.gateway.json

## Deleting Changes or Reverting to Previous State

To remove a certain advanced configuration, just delete the section pertinent to that configuration in the **config.gateway.json**. To completely remove all advanced configurations created in the **config.gateway.json**, delete the file or rename it. This will void all manual changes. The USG will be provisioned with the current config contained within the UniFi Network application.

A best practice when editing an already working **config.gateway.json** file is to create a backup. If you need to add additional changes to the file, rename the current file to **config.gateway.json.old**, and copy all the existing and new changes into a new file named **config.gateway.json**. This essentially creates a backup, and if there happens to be any mistakes resulting in a "commit" error or provisioning loop, you can delete **config.gateway.json**, and try again starting from **config.gateway.json.old.**

---