NaCl

Not Another Configuration Language

1 Datatypes

Datatypes that exist in NaCl behind the scenes are:

- integer (a number, f.ex. 10 or (-10))
- IPv4 address (f.ex. 10.0.0.45)
- IPv4 cidr (f.ex. 10.0.0.0/24)
- bool (f.ex. false)
- string (f.ex. “Hi”)
- range (f.ex. 10-20 or 10.0.0.40-10.0.0.50)
- list (f.ex. [10, 20, 30])
- object (f.ex. { key1: 10, key2: 20 })

2 Typed objects

A typed object initialization has the following structure: <type> <name> <value>, where the type can be Iface, Gateway or Vlan.

2.1 Iface

An Iface is a type that has certain requirements. The following attribute must be specified for each Iface created:
- index (integer)

The following attributes must be given if you don't set the config attribute to dhcp, or don't set the config attribute at all:
- address (IPv4 address)
- netmask (IPv4 address)

Other attributes that can be specified are:
- gateway (IPv4 address)
- dns (IPv4 address)
- config (dhcp, dhcp-with-fallback or static)
- masquerade (can be set to true or false, where false is default)
- prerouting (names of functions)
- input (names of functions)
- output (names of functions)
- postrouting (names of functions)
- vlan (names of Vlans or anonymous Vlan objects)

The value of an Iface can be an object. The object consists of key value pairs, separated by comma, and the pairs are enclosed by curly brackets:

```
Iface eth0 {
        address:        10.0.0.45,
        netmask:        255.255.255.0,
        gateway:        10.0.0.1,
        dns:            8.8.8.8,
        index:          0
}
```

The value can also simply be the configuration type (config) you want the Iface to have: dhcp, dhcp-with-fallback or static. Different requirements are connected to each of these.

For example, if you only want to set an Iface configuration to dhcp, you can use this syntax:

```
Iface eth0 dhcp
```

But since the index attribute always has to be set, you also need to set this:

```
Iface eth0 dhcp
eth0.index: 0
```

The dhcp-with-fallback configuration requires you to specify a fallback address and netmask:

```
Iface eth0 {
        config: dhcp-with-fallback,
        index: 0,
        address: 10.0.0.45,
        netmask: 255.255.255.0
}
```

The static configuration also requires an address and a netmask to be specified. This is default, and doesn't need to be specified:

```
Iface eth0 {
        index: 0,
        address: 10.0.0.45,
        netmask: 255.255.255.0
}
```

An Iface's attributes can be set outside an object specification as well. F.ex.:

```
Iface eth0 dhcp-with-fallback
eth0.index: 0
eth0.address: 10.0.0.45
```

```
eth0.netmask: 255.255.255.0
eth0.gateway: 10.0.0.1
```

These attributes can be set anywhere in the NaCl file.

An Iface has 4 chain attributes that functions can be pushed onto (we'll come back to functions later, but the name of a function can be set as an Iface's chain's value). These chains are prerouting, input, output and postrouting.

```
Iface eth0 dhcp
eth0.index: 0
eth0.prerouting: my_function
```

More than one function can be added to a chain, but only one function of the type Filter should be added to each. This is because a Filter always returns either drop or accept, and the Filter that follows a Filter will never get called. If you want to add more than one function to a chain, you have to specify a list:

```
Iface eth0 {
        config: dhcp,
        index: 0,
        prerouting: [ my_filter, my_first_nat, my_second_nat ]
}
```

2.2 Gateway

A Gateway object consists of routes. The value of a Gateway object can either be a list of route objects, or an object consisting of key value pairs, where each pair's value is a route object:

```
Gateway myGateway [
        {
                net: 10.0.0.0,
                netmask: 255.255.255.0,
                iface: eth0
        },
        {
                net: 10.10.10.0,
                netmask: 255.255.255.0,
                iface: eth1
        },
        {
                net: 0.0.0.0,
                netmask: 255.255.255.0,
                nexthop: 10.0.0.1,
                iface: eth0
```

```
        }
]
```

or

```
Gateway myGateway {
        route1: {
                net: 10.0.0.0,
                netmask: 255.255.255.0
        },
        route2: {
                net: 10.10.10.0,
                netmask: 255.255.255.0,
                iface: eth1
        },
        defaultRoute: {
                net: 0.0.0.0,
                netmask: 255.255.255.0,
                nexthop: 10.0.0.1,
                iface: eth0
        }
}
```

If you create a Gateway with named routes, you can refer to these routes elsewhere in the NaCl file to set values that you haven't already set inside the route:

```
myGateway.route1.iface: eth0
```

The possible attributes of a Gateway route is:

- net (IPv4 address)
- netmask (IPv4 address)
- gateway (IPv4 address)
- iface (name of an Iface)
- nexthop (IPv4 address)
- cost (integer)

2.3 Vlan

The Vlan type is similar to the Iface object, but is meant to be added to an Iface's vlan attribute.

The following attributes can be specified for a Vlan object:

- address (IPv4 address)
- netmask (IPv4 address)
- gateway (IPv4 address)
- index (integer)

Index, address and netmask are mandatory to specify.

```
Vlan myFirstVlan {
        index: 13,
        address: 10.50.0.10,
        netmask: 255.255.255.0
}
```

The Vlan can then be added to an Iface's vlan:

```
Iface eth0 dhcp
eth0.vlan: myFirstVlan
```

More than one Vlan can be added to an Iface's vlan:

```
Iface eth0 dhcp
eth0.vlan: [ myFirstVlan, mySecondVlan ]

Vlan mySecondVlan {
        index: 22,
        address: 10.60.0.10,
        netmask: 255.255.255.0
}
```

A Vlan object doesn't need to be created, however, to set an Iface's vlan attribute:

```
Iface eth0 {
        index: 0,
        address: 10.0.0.45,
        netmask: 255.255.255.0,
        gateway: 10.0.0.1,
        vlan: [
                {
                        index: 13,
                        address: 10.50.0.10,
                        netmask: 255.255.255.0
                },
                {
                        index: 22,
                        address: 10.60.0.20,
                        netmask: 255.255.255.0
                }
        ]
}
```

3 Untyped objects

You can create objects with values of any of the datatypes listed in section 1. The initialization of an untyped object consists of: <name>: <value>

myPort: 4040

myPorts: [30, 40, 50, 60]

myAddress: 10.0.0.45

myAddresses: [10.0.0.40, 10.0.0.50, 10.0.0.80-10.0.0.90, 30.20.10.0/24]

myCidr: 10.0.0.0/24

myCidrs: [10.0.0.0/24, 30.20.10.0/20, 100.20.32.50/32]

```
myObject: {
        key1: 10,
        key2: {
                key2-1: 50,
                key2-2: 60
        }
}
```

These objects can be used in your functions or as values to your Iface, Vlan or Gateway routes' attributes.

4 Functions

4.1 Filter

Firewall

4.2. Nat

dnat
snat