1 Scenarios

Scenario 1
Gratuitous ARP responses are sent when a host joins a network to let others know of its presence. It may also be sent in case of a fail-over when a backup host takes over. This takes place when sender and target IPs are the same (http://networkengineering.stackexchange.com/questions/10645/identify-gratuitous-arp)
Flag: notice

Scenario 2
ARP replies should not try to bind an IP address to the broadcasting MAC (ff:ff:ff:ff:ff:ff).
Flag: error

Scenario 3
An ARP reply containing a new IP address for an existing binding is sent out.
Flag: notice

Scenario 4
ARP replies attempting to overwrite entries in the reference table, which should not appear.
Flag: error

Scenario 5
There should not be any contradictions to official IP-MAC bindings in ARP replies.
Flag: error

2 Documentation

The configuration file is an XML. The root is `<conf>` which currently has one type of child `<modules>`. Here the user can define the module with the tag `<module>` which has the children `<name>`, `<description>` (which displays the error message), `<subscriptions>` and attributes that are specific to the module, `reference-table` and `rules` for `DAI`. `<subscriptions>` has the child `<subscription>` which takes two kinds, `<packet>` and `<bus>`. So each module can tell the PCAP module which packets it is interested in, and the bus subscription enables the inter-communication of modules.

```
<conf>
   <modules>
      <module>
         <name>TEST</name>
         <description>Testing module which makes it easy to check if it received messages from
             subjects</description>
         <subscriptions>
            <packets/>
            <bus>DAI,error</bus>
         </subscriptions>
      </module>
      <module>
         <name>DAI</name>
         <description>Dynamic ARP Inspection</description>
         <subscriptions>
            <packets>ARP</packets>
            <bus/>
         </subscriptions>
         <reference-table>
            <entry>
               <type>switch</type>
               <ip>145.94.212.11</ip>
               <mac>a1:b2:c3:d4:e5:f6</mac>
            </entry>
            <entry>
```

```
                <type>official</type>
                <ip>145.94.212.99</ip>
                <mac>aa:bb:cc:dd:ee:ff</mac>
            </entry>
        </reference-table>
        <rules>
            <rule>
                <name>static-binding</name>
                <description>Binding belongs to a static ARP table</description>
                <flag>error</flag>
            </rule>
            <rule>
                <name>gratuitous</name>
                <description>Response was triggered without a request</description>
                <flag>notice</flag>
            </rule>
            <rule>
                <name>response-from-broadcasting</name>
                <description>Response was from the broadcasting address which will cause host to
                    broadcast all messages sent to IP address</description>
                <flag>error</flag>
            </rule>
            <rule>
                <name>overriding-existing-binding</name>
                <description>Response triggers an update of an entry in the ARP
                    table</description>
                <flag>notice</flag>
            </rule>
        </rules>
    </module>
  </modules>
</conf>
```

In the example XML above, the module for `DAI` subscribes to `ARP` packages from PCAP but is not subscribed to any of the other modules via the bus. The `TEST` module, on the other hand, subscribes to the output from `DAI` and `error`, but not to any packages.

For details on how to execute the program, please refer to the README file.

Error Handling

The `error` module is in-built and uses the logic in place to notify the bus that there has been an error which forced the operation of the module to stop. This way the bus can log the error and display the message that caused the module to fail, preventing the entire program from crashing. The `TEST` module subscribes to the error message so that the forwarding mechanism can be verified in a unit test.

This takes place, for example, when conditions for a rule is met, but it was not in the .config file. The test `testErrorHandlingWhenRuleIsMissing` in `tests.py` demonstrates this.

Design Pattern

This assignment was designed using the `Observer pattern`. When the main.py file is executed, the `pcapy` library either captures live traffic or reads from a `.pcap` file. First, however, it reads the configuration file which sets up the modules, pcap and the bus and subscribes the modules as per the `<subscriptions>` element.

Then the packets are parsed and forwarded to all module has subscribed to them through `Pcap` defined in `pcapp.py`. All modules, for now, are located in `module.py` and in the case of `DaiModule`, calls `ArpAnalyzer` and forwards the results to the `bus` in `bus.py`. The bus logs the message and forwards it to the `DAI` subscriptions.

Testing

The following ARP files are used in `tests.py` but can also be used for normal execution.

- arpdump.pcap

- arpsample.pcap
- gratuitous.pcap (Scenario 1)
- brcast-error.pcap (Scenario 2)
- ip-update.pcap (Scenario 3)
- mac-update.pcap (Scenario 3)
- static-violation.pcap (Scenario 4 and 5)

To test the aforementioned in-built `error` module, the following can be executed.
python main.py conf-error.xml gratuitous.xml
The `conf-error.xml` file is the same as `conf.xml` only it has the rule for gratuitous replies missing.