FSM Plugin API

FSM plugins are created through the OVSDB Flow_Service_Manager_Config table. A FSM plugin entry contains the following fields:

- **handler**: the fsm plugin name
- **if_name**: tap interface attached to the br-home ovs bridge
- **pkt_capt_filter**: libcpap filter applied to the interface
- **other_config**: a map of [key, values], one item containing the location of the dynamic shared library implementing the plugin

Here is a sample of the table contents:

<table>
<thead>
<tr>
<th>handler</th>
<th>if_name</th>
<th>other_config</th>
<th>pkt_capt_filter</th>
<th>plugin</th>
</tr>
</thead>
<tbody>
<tr>
<td>test</td>
<td>br-home.test</td>
<td>{dso=&quot;/usr/plume/lib/libtest.so&quot;}</td>
<td>&quot;udp dst port 3000&quot;</td>
<td></td>
</tr>
</tbody>
</table>

On reception of this information, fsm will open the dso and lookup a function called `{handler}_init_plugin`, "test_plugin_init" in our example. This symbol is the FSM plugin entry point.

Next fsm will call the plugin entry point and pass the following structure:

```c
struct fsm_session {
    struct schema_Flow_Service_Manager_Config *conf;
    struct fsm_pcaps *pcaps;
    char *topic;
    void (*send_report)(struct fsm_session *, char *);
    void (*handler)(uint8_t *, const struct pcap_pkthdr *, const uint8_t *);
    void (*handler_ctxt);
    void (*update)(struct fsm_session *);
    void (*periodic)(struct fsm_session *);
    void *handle;
    char *session_mqtt_headers[FSM_NUM_HEADER_IDS];
    char mqtt_key[256];
    char mqtt_val[256];
    char dso[256];
    ds_tree_node_t fsm_node;
    int64_t report_count;
    struct ev_loop *loop;
    bool has_topic;
    bool has_awlan_headers;
};
```

The FSM plugin is responsible for instantiating the (*handler), (*update) and (*periodic) function pointers. The (*handler) function pointer is the handler for captured packets. The (*update) function pointer is the callback for ovsdb updates. The (*periodic) function pointer is the callback for periodic calls.