Getting Started with Mycelium

Getting Started with Mycelium

Mycelium is a high-performance, modular Software Defined Radio (SDR) framework designed for automated RF missions, protocol analysis, and rapid prototyping. This guide will help you set up your first mission-ready tool chain.

Core Concepts

Before we begin, it's important to understand the three pillars of the Mycelium architecture:

- **Tool**: A virtual radio instance. Think of a Tool as a software-defined radio receiver or transmitter with a specific **Protocol**, **Modulation**, and **SDR** hardware assigned to it.
- **Directive**: The "brain" of a tool. Directives define a Tool's behavior through a **Condition** (when to act), an **Action** (what to do), and an optional **AfterAction** (the next step).
- **Mission Pack**: A collection of domain-specific protocols and tools (e.g., Aviation, Tactical, IoT) that expand the capabilities of the core framework.

Launching Mycelium

1. Start the Framework

If you've installed the package, simply run:

```
myce
```

If you are running from the build directory:

```
./Mycelium
```

2. Identify Your Hardware

Check which SDR devices are connected to your system:

```
>> sdr show
```

For testing without physical hardware, create a virtual receiver:

```
>> sdr create -n v_rx -t VirtualRx
>> sdr set -n v_rx -g true
```

3. Create and Configure Your First Tool

Let's build a simple FM receiver tool named `my_radio`:

```
>> tool create -p Raw -n my_radio
>> tool set -n my_radio -r v_rx -m FM -f 100300000
```

4. Implement Autonomous Logic (Directives)

Directives allow your radio to act on its own. Let's create a loop that listens for a signal and logs it:

```
# Continuously receive 64k audio samples
>> directive create -t my_radio -n get_audio -c Null -a Receive_Audio -s 65536

# Print a message to the console every time a signal is detected
>> directive create -t my_radio -n notify -c Any -a Print -F "Signal detected at $timestamp"
```

5. Execute the Mission

Start the real-time execution engine:

```
>> execute
```

To stop the session:

```
>> stop
```

Mastering the Workflow

Once you've built a complex setup, you can save it as a mission script for future use:

```
>> save -f mission_alpha.myce
```

You can then relaunch the entire mission with a single command:

```
myce mission_alpha.myce
```

Next Steps

Explore the full power of Mycelium's automation engine in the User Guide & Directive Engine or see real-world applications in Industry Solutions.