Hivemind Control, Monitoring and Alerting Interface

The Hivemind Interface will provide a convenient way to control the sensor system, monitor its status and receive alerts of particularly significant events. Status monitoring will support the needs of both the Hivemind system developer, Experimenters wishing to understand the behavior and impact of the Hivemind system.

DISPLAY

* Status of the monitoring system
	+ Information about the monitored nodes
		- Number nodes monitored
		- # nodes up (vs. down)
		- # nodes not responding (vs. able to be up or down)
		- List of all monitored nodes w/ status
			* Up/down
			* Assigned neighbors
			* …
	+ Information about active Sensors
		- # per class: Ant/Wasp
		- # per state: foraging/following/marking/idle
		- # per type: (partial example list)
			* New service detected
			* New user account detected
			* Vulnerable service
			* Remote accessible service
			* New unknown process
			* Connection out
			* Connection in
			* Unexpected # bytes in
			* Unexpected # bytes out
			* File checksum error
			* Excessive # process restarts
			* Excessive # socket timeouts
			* # sockets in use excessively high (vs. limit)
			* # threads in use excessively high (vs. limit)
			* Custom rule
			* Other rules specific to network devices (vs. hosts) - future
			* …
			* *These will frequently be an estimate of the actual values. This is a decentralized system so complete central status of the entire system is both impractical and likely to degrade performance of the monitored system*
	+ Log of reported events and alerts
	+ Visualizations
		- Sensor activity
		- Heat map of sensor activity
			* Over adjustable time window
		- Neighbor map w/ node status
		- Event activity
	+ Testing/Performance Statistics
		- Time to detect
		- …
* Event Monitoring
	+ Experiment Swap-In Report
		- Baseline + concerns
	+ New node report
		- Single node baseline
	+ Real-time monitoring
* Current value of run-time configurable parameters
	+ Configurable
	+ Non-configurable
		- (those requiring configuration file edit and restart)
* System Control commands
	+ Start node-manager processes
		- On node or by node list
	+ Stop node-manager processes
		- On node or by node list
	+ Get Status from Node-manager
	+ Generate list of nodes being monitored
	+ Assign neighbors to nodes
	+ Start creating sensors
	+ Stop creating sensors
	+ Request termination of a particular sensor
* Sensor Configuration Commands
	+ Add new sensor type
	+ Remove sensor type
	+ Change sensor type parameters
		- Details will vary by sensor type
* Change system/node/sensor parameters
	+ Linger time
	+ Creation rate
	+ Death rate
	+ Crowding factor
	+ Recruitment rate
	+ Focus
	+ Follow chance
	+ Wander Factor
	+ Ant/Wasp ratio
	+ Marker persistence
	+ Trail length
	+ Idle period
	+ …

IMPLEMENTATION

* Access to the Hivemind Control, Monitoring and Alerting interface will be through a web browser. In order to provide both textual and graphical content, the interface content will be HTML5 based. This will allow the system to be monitored and controlled from any browser that supports HTML5. Recent versions of most modern browsers (IE, Safari, Firefox, Chrome, iPhone) support HTML5. All but the animated graphic features should be supported by any browser.
* GUI functions will execute a particular script to perform the required function. These will be a combination of Python scripts (the current command line scripts or similar) an Javascript to support GUI functions.
* For the monitoring functions:
	+ The web GUI will either receive update messages from the node-managers or by periodically polling them.
	+ Some functions will be updated by user command, for example, to query the node-manager about its configuration, e.g. who its neighbors are, its actual configuration parameter values.
* Hivemind enabled Experiments will require a change to their rspec files (or equivalent). This can be automated. Initially by running a script to enable Hivemind monitoring, later by a checkbox on the Federator’s? (e.g. ProtoGENI) command GUI. The IP address of the server to provide monitoring and control access will be returned in the manifest and/or emailed to the experimenter. Alternatively, future implementations may add the capability to enable Hivemind monitoring into GENI infrastructure code.
* Ultimately, the GUI would be served by the same web server used by Experimenters for command and control of their experiments (e.g. ProtoGENI).

ACCESS

* Access to the Hivemind GUI will be available from any browser able to establish a network connection with the html server associated with the experiment or aggregator. Initially this will be provided by an Apache instance on a separate monitoring node associated with each experiment. Later this will be modified to use a web server designated by the Aggregator to allow the Aggregator to monitor the status of the slices under its control.
* Access to an Experiment’s Hivemind will be controlled by settings in the Experiment’s configuration file. Because the interface web server will run on an additional node assigned to the Experiment, access to that node will be restricted in the same manner as access to the other nodes in the Experiment are restricted. Separate usernames and passwords may be used; IP addresses allowed to access the GUI may also be specified by the Experimenter. These may be specified in the configuration of the monitoring system to be used by the web server.
* When monitoring slices on an Aggregator’s site, a server to support monitoring will be provided by the Aggregator. Access to this server would be controlled by the Aggregator and presumably limited to Aggregator staff and associates.