First and foremost, due to limitations on software, hardware and licensing the initial setup is different from the target setup. Once those limitations are lifted, those goals can be met. I will describe both the current setup and the target setup.

The current setup is hosted predominantly on my personal research cluster and leverages the ESXi hypervisor to provide virtual machines to the head node. The head node serves multiple purposes; it hosts the malware DB, samples and metadata collected from VT, and coordinates tasks between multiple appliances. It also hosts the Cuckoo appliance. The Cuckoo appliance performs its malware analysis on other VMs that have been set up specifically for the purpose of malware analysis. Whenever a sample is ingested by Cuckoo, it will pick a template VM based on the type and format of the malware. After which it will spin up the VM and emulate user input, proper system state and network traffic for the sample to interact with.

The data generated by cuckoo is easily accessible through a web portal that displays most of the information gathered. Full data dumps are available in a folder structure containing pcaps, memory dumps, files of interest and metadata gathered. This data was then sorted and relevant data points were marshaled and ingested to a SQL server for easy access and data modeling. Cuckoo's usefulness comes into play by generating more behavioral data than could be obtained through VT's API at the cost of compute power. And given our use case, this becomes a very cost-effective solution to obtaining behaviour data.

Using this one tool, we were able to properly extract system call deltas over time to make a first pass at modeling how malware evolves. This is a very useful first step in defining family similarity. The graph generated was resonant with more traditional genetic models used to display similarity and a very good parallel to start with.

The optimal setup that we would like to see be actionable is one wherein more data could be extracted with less compute cost. To do this, the first step would be to acquire a Hex-Rays license and use that software suite to extract a control flow graph (CFG) of the program. The extracted CFGs could be compared over time to examine not only shifts in system calls, but shifts in program trace execution. This CFG would also contain points of interest worth investigating my more in-depth analysis engines such as MIASM or ANGR.

The second problem encountered was the lack of proper hardware support from Azure. As cuckoo requires nested virtualization to properly display system state to the malware, we were unable to proceed with our given hardware. This, unfortunately, does not have a fix beyond Microsoft enabling this feature on their cloud platform.