In our problem analysis, the first major that stood out was the inability of connecting infrastructure monitoring services such as firewalls and intrusion detection systems into global notification services. The second major issue was the difficulty of administrators remotely administering their services once they had been notified of an issue. This project seeks to make headway in both those areas by producing a scalable application able to interface with popular security solutions and provide easy to use remote administration.

The goals the project are intended to meet are as follows:

* Review literature related to chat bots, natural language processing, cyber security and visualisation methods.
* Address the inability for administrators to be able to access commands from within a network remotely.
* Address connections between notification systems and data generation systems.

The problem with remote infrastructure administration is that generally, you are away from your tools. Having a dashboard with all your servers’ status is useful as long as you can access it. It is very important that administrators know how their infrastructure is operating, as outages may affect a business very quickly. While it is possible that somebody will notice the outage and call the person responsible, it is always better to be informed by trustworthy sources of the scenario occurring. Being away from your tools also limits your ability to work on an issue once notified, and having remote access to tools even as simple as ping can be essential in troubleshooting.

To be more specific with aims, there are several parts to the project that need to work for the project to work as a whole:

* Create a framework for a chat bot and connect it to a popular IM service, and ensuring its style remains modular to allow for easy replacement of services.
* Connect the chat bot to a natural language processing service to allow it to respond certain ways for certain commands, such as LUIS[2]. Give access to commands such as ping, or traceroute.
* Test both the framework and bot to see if it works in an example scenario.
* Critically analyse the framework and bot to see if it lives up to the problem specifications.

Allowing the chat bot to communicate via different IM services is vital. Some companies use Cisco Jabber and some have no IM communication at all. For the sake of having messages that can be formatted, Telegram is a good choice for testing. However, the bot should be able to be modified to work with most IM services with minimal effort. Natural Language Processing services, like Microsoft’s LUIS, Google’s Natural Language API and Facebook’s Wit.ai are all viable to be used for chat bots. Having an easy to train platform that is able to parse requests without complex regular expressions will make it much easier to expand the bot in the future.

While it is not entirely necessary to have the bot connected to a production server, we believe it is important for the bot to encounter live data so it can be taught how to deal with it. Being able to remotely access logs via SSH or FTP and parse them for data is the cornerstone of the security monitoring aspect of the bot.

One of the most important parts of the whole project is the alerts and notifications system. Once an alarm is tripped, a specified user should be informed via their IM service. While we do not expect this aspect to be as powerful as the tools of a well-equipped network administrator, this should at least give them access to part of their arsenal in which to troubleshoot and assess the situation. This could be anywhere from ping, to dig, to a fully-fledged remote shell.

The bot should be scalable to support as many different users, IM services, notification types, API reads and remote servers as possible. The bot should never be the bottleneck, only the hardware on which it runs.

To complete these objectives, we intend to use an array of utilities ranging from online services to open source tools. This will likely include utilising tools on both Windows and Linux for a more realistic approximation of real administration. It is hoped that this tool will be used in day to day administration of a real network, but it should not be specific to that network.

# References

[1] Telegram, “Telegram Messenger,” 2017. [Online]. Available: https://telegram.org/. [Accessed: 05-Mar-2017].

[2] Microsoft, “LUIS: Language Understanding Intelligent Service (beta).” [Online]. Available: https://www.luis.ai/. [Accessed: 15-Dec-2016].

[3] Bro, “New Script-Level Logging API,” 2017. [Online]. Available: https://www.bro.org/development/projects/logging-api.html. [Accessed: 05-Mar-2017].

[4] Aldeid, “Wmic-linux - aldeid,” 2017. [Online]. Available: https://www.aldeid.com/wiki/Wmic-linux. [Accessed: 05-Mar-2017].