CS 477
Final Project
Architectural Report
Fall 2019

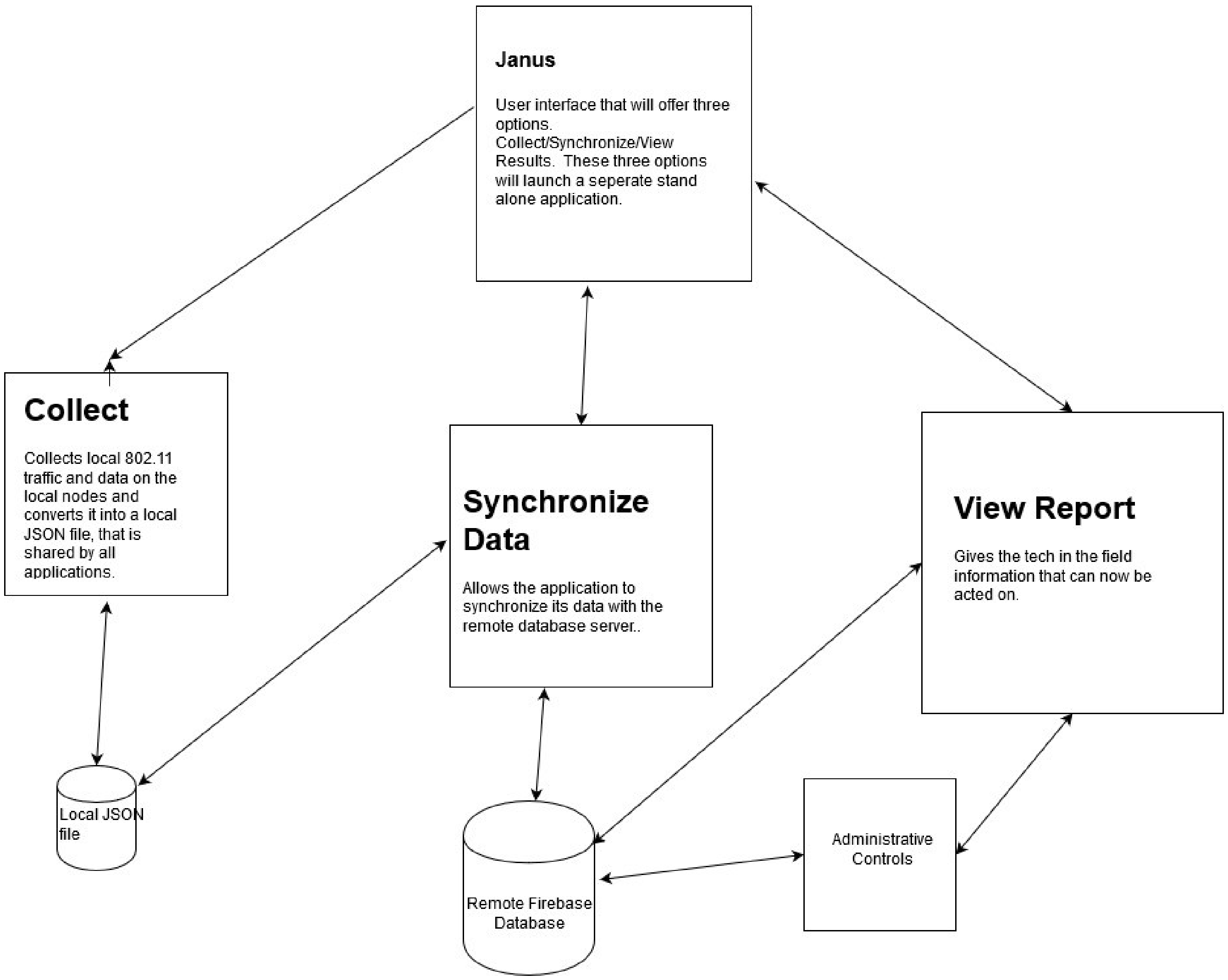
Christopher Wells
G-00260513

Matt Mullaly
G-00806288

Overview:

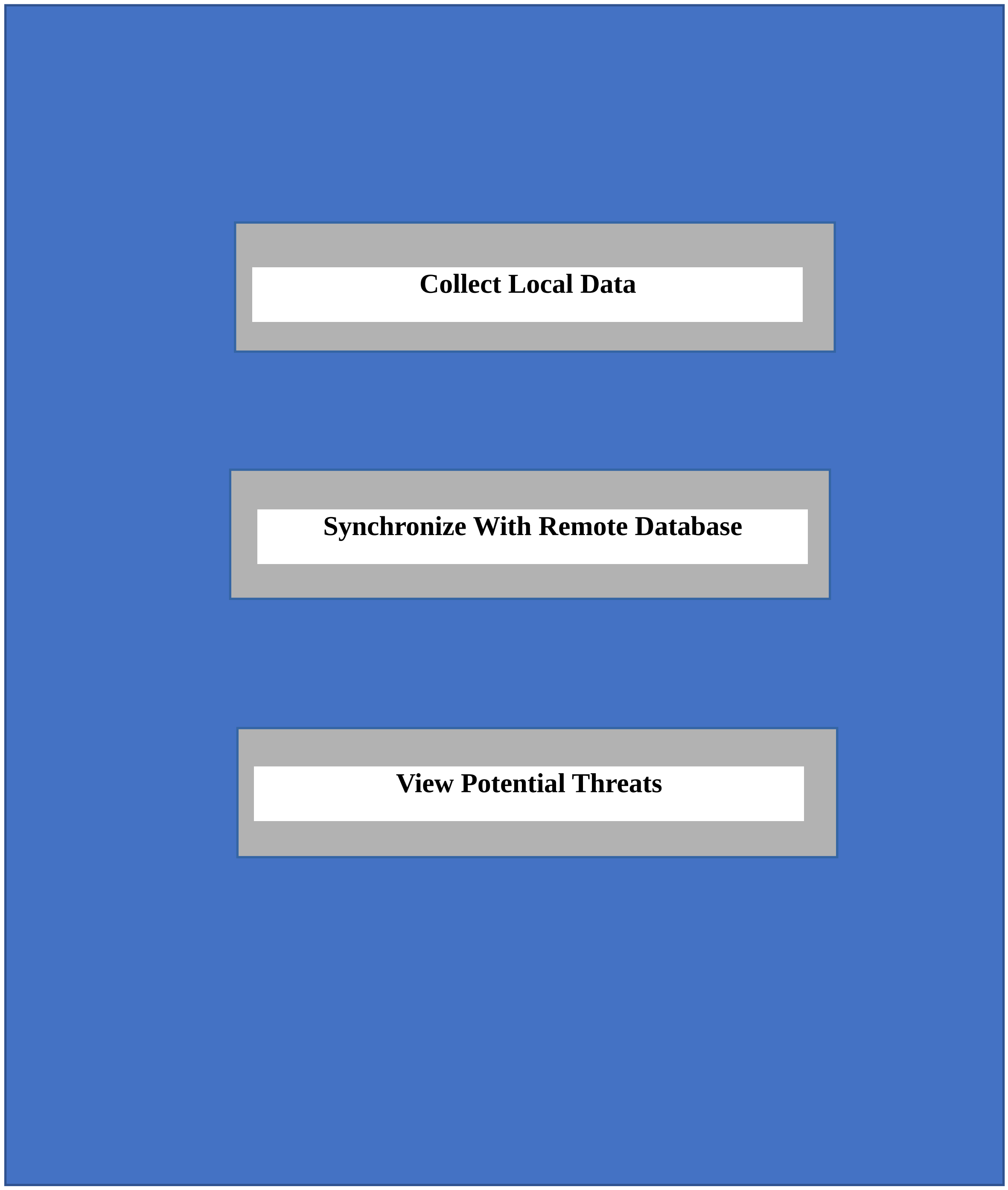
The design philosophy of this project is to divide and conquer. The project will be divided up into separate stand alone components. In this way the complexities of the larger project can be broken into manageable pieces. The main project will be broken into three general categories. The first being the collection of information. The second being the analysis of the information. The last component is going to be the processing of this information into actionable information.

Top Level Overview


Janus
User interface that will offer three options. Collect/Synchronize/View Results. These three options will launch a seperate stand alone application.
Collect
Collects local 802.11 traffic and data on the local nodes and converts it into a local JSON file, that is shared by all applications.
Synchronize Data
Allows the application to synchronize its data with the remote database server..
View Report
Gives the tech in the field information that can now be acted on.
Local JSON file
Remote Firebase Database
Administrative Controls

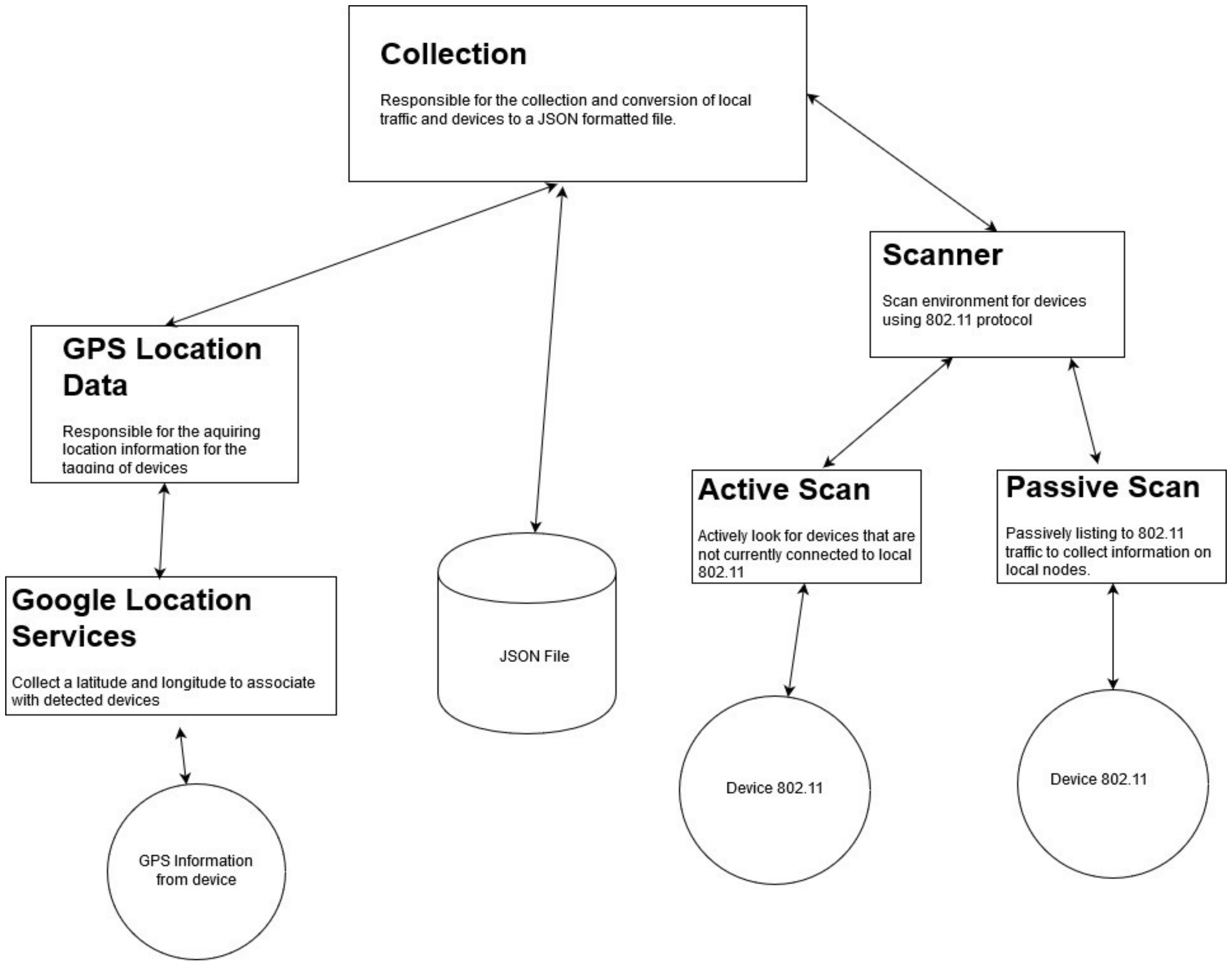
The modular design will allow for the easy replacement of a component, without the redesign of the entire project. The design will also allow for future growth. It is oped that this basic platform to be built upon and customized to fit environmental needs.

Janus User Interface



This application serves as a interface with the user to allow for the selection of the proper application to complete the task. There is not a large amount of code involved in the operation of this application.

<u>Collection</u> (802.11 protocol)


Collection
Responsible for the collection and conversion of local traffic and devices to a JSON formatted file.
Scanner
Scan environment for devices using 802.11 protocol
GPS Location Data
Responsible for the aquiring location information for the tagging of devices
Active Scan
Actively look for devices that are not currently connected to local 802.11
Passive Scan
Passively listing to 802.11 traffic to collect information on local nodes.
Google Location Services
Collect a latitude and longitude to associate with detected devices
JSON File
Device 802.11
Device 802.11
GPS Information from device

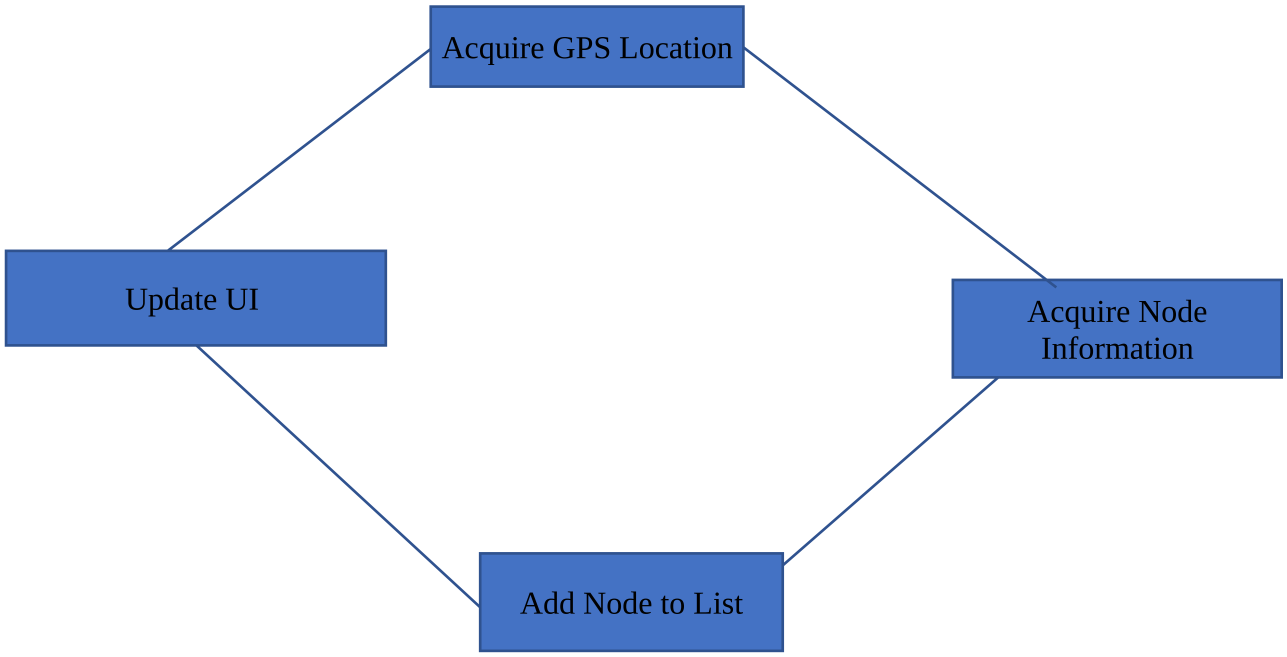
Collection scans the environment for devices, and tags their traffic and ID. This data is collected and formatted in JSON format and stored in a common file for use by other components of the platform.
GPS Location Data will use the device GPS to pin a location that the data is being collected.
Active Scan, will attempt to ping addresses that seem inactive to see if there is a response. Look for sleeping devices. (Application would be to look for devices connected but inactive. The idea springs form a tablet that was once found in an office that no one knew about. It was running Android 2 and connected to the network, with no security.)
Passive Scan, puts the phone into passive mode and collects traffic in the area. This by far will be the primary method of collecting data.

The stored data will be:

Location:
- Lattitude: Double
- Logitude: Double

Device:
- Device ID: 64 bit, first 16bits will be zero for older devices
- Destination IP: 32bits for the address that the device is talking to
- Source IP: 32 bits
- Date: Date
- Time Start: Time
- Time End:(Duration of the Communication to the site. Extremely short could be a sign of redirection to a Malware location)

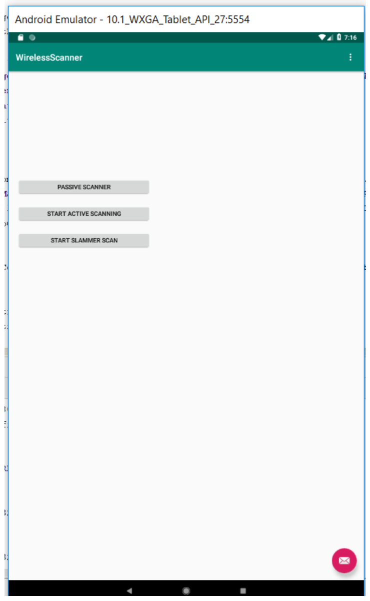
<u>Data Collection Cycle</u>



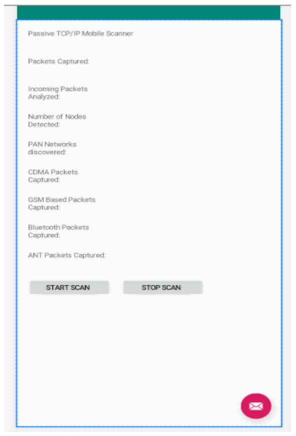
Updates Since the Last Design Document

From a design perspective there has been little design changes since the last design document. The focus has been the implementation and refinement of the design planed in the previous report. The UI for the passive scanner has been finalized. A lot of the features in the layout will not be implemented in this version. Pressing these buttons will result in a toast message that tells the user that these elements will be added in a future version. Most of these elements are not going to be implemented due to budget. I am a broke college student and can not afford the antennas and hardware needed to support these features. Slammer scan was added in this design, it is a brute force scan of the local nodes. Of course this will have a negative effect on local traffic, so it is highly discouraged. This design came form a lecture in CS 455 about determining the local nodes through polling. The intent is that a search like this will be carried out during non-business hours. I am also getting some artwork to make the project look better. The database of node information will be uploaded to a database named “MobileWirelessScanning”. Some prelimanry data has been collected to see what the data results would look like.

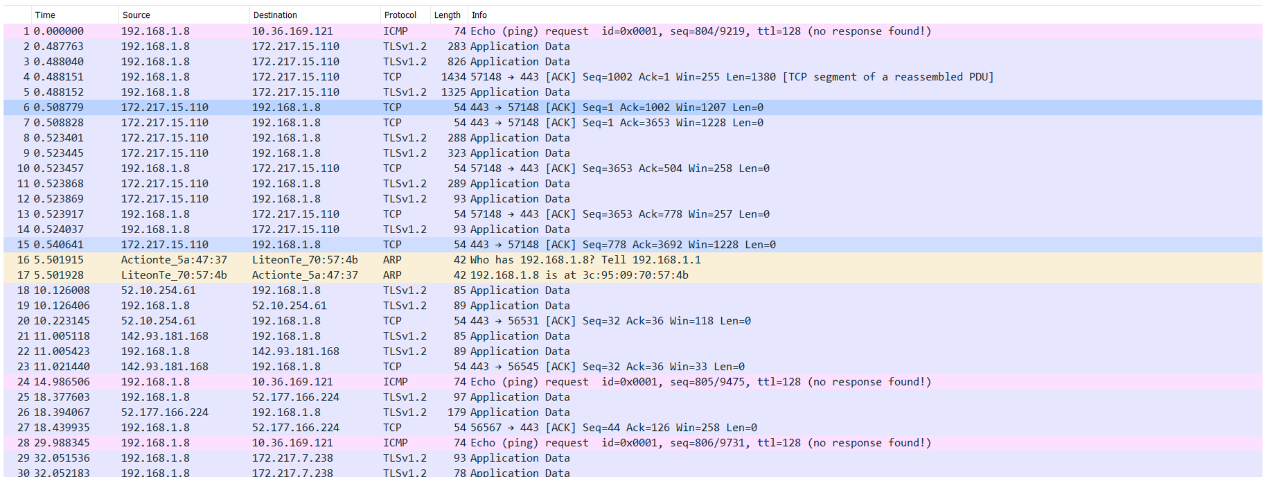
Scanner Main Screen:



Passive Scanner:


Passive TCP/IP Mobile Scanner
Packets Captured:
Incoming Packets Analyzed:
Number of Nodes Detected:
PAN Networks discovered:
CDMA Packets Captured:
GSM Based Packets Captured:
Bluetooth Packets Captured:
ANT Packets Captured:
START SCAN
STOP SCAN

Collected Data:



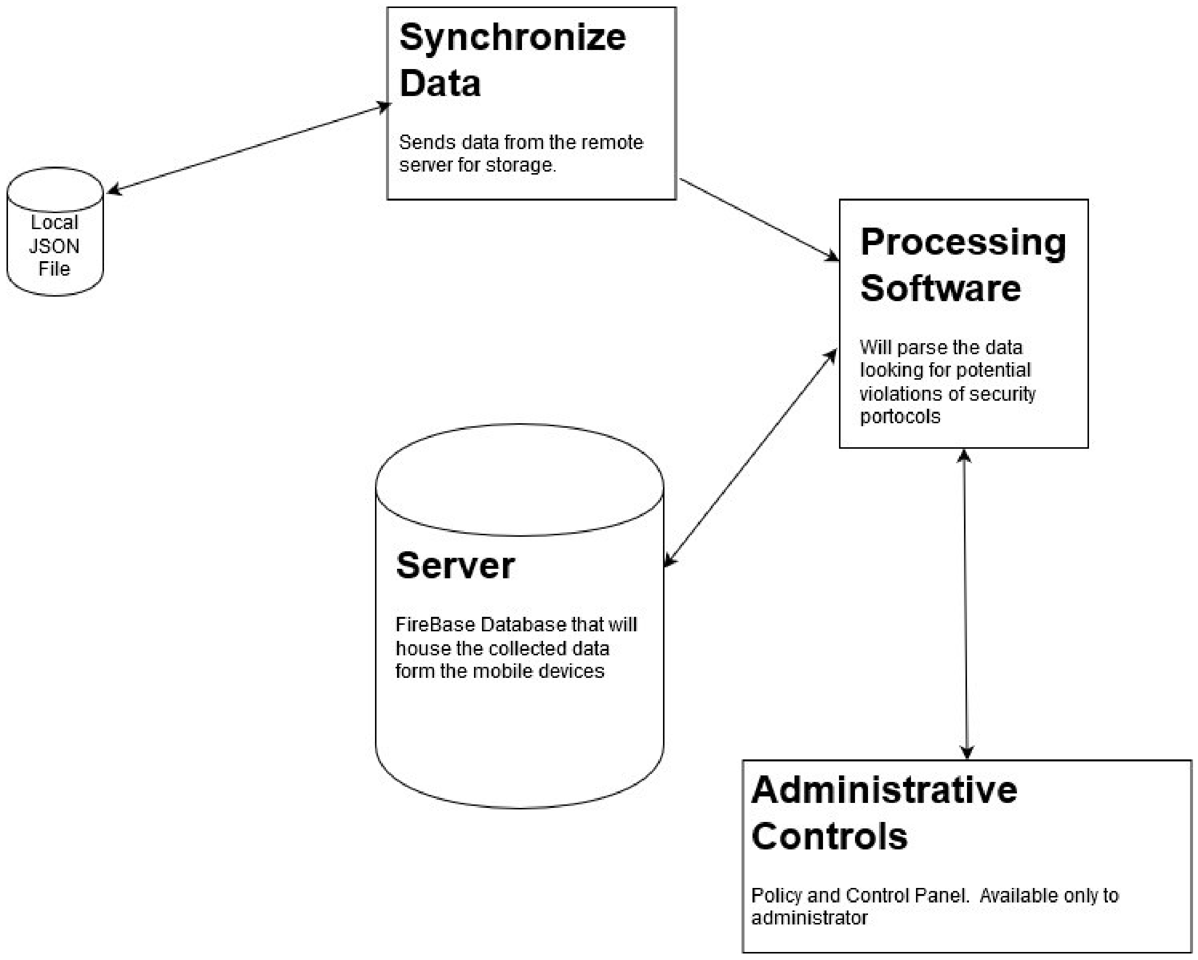
	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.8	10.36.169.121	ICMP	74	Echo (ping) request id=0x0001, seq=804/9219, ttl=128 (no response found!)
2	0.487763	192.168.1.8	172.217.15.110	TLSv1.2	283	Application Data
3	0.488040	192.168.1.8	172.217.15.110	TLSv1.2	826	Application Data
4	0.488151	192.168.1.8	172.217.15.110	TCP	1434	57148 → 443 [ACK] Seq=1002 Ack=1 Win=255 Len=1380 [TCP segment of a reassembled PDU]
5	0.488152	192.168.1.8	172.217.15.110	TLSv1.2	1325	Application Data
6	0.508779	172.217.15.110	192.168.1.8	TCP	54	443 → 57148 [ACK] Seq=1 Ack=1002 Win=1207 Len=0
7	0.508828	172.217.15.110	192.168.1.8	TCP	54	443 → 57148 [ACK] Seq=1 Ack=3653 Win=1228 Len=0
8	0.523401	172.217.15.110	192.168.1.8	TLSv1.2	288	Application Data
9	0.523445	172.217.15.110	192.168.1.8	TLSv1.2	323	Application Data
10	0.523457	192.168.1.8	172.217.15.110	TCP	54	57148 → 443 [ACK] Seq=3653 Ack=504 Win=258 Len=0
11	0.523868	172.217.15.110	192.168.1.8	TLSv1.2	289	Application Data
12	0.523869	172.217.15.110	192.168.1.8	TLSv1.2	93	Application Data
13	0.523917	192.168.1.8	172.217.15.110	TCP	54	57148 → 443 [ACK] Seq=3653 Ack=778 Win=257 Len=0
14	0.524037	192.168.1.8	172.217.15.110	TLSv1.2	93	Application Data
15	0.540641	172.217.15.110	192.168.1.8	TCP	54	443 → 57148 [ACK] Seq=778 Ack=3692 Win=1228 Len=0
16	5.501915	Actionte_5a:47:37	LiteonTe_70:57:4b	ARP	42	Who has 192.168.1.8? Tell 192.168.1.1
17	5.501928	LiteonTe_70:57:4b	Actionte_5a:47:37	ARP	42	192.168.1.8 is at 3c:95:09:70:57:4b
18	10.126008	52.10.254.61	192.168.1.8	TLSv1.2	85	Application Data
19	10.126406	192.168.1.8	52.10.254.61	TLSv1.2	89	Application Data
20	10.223145	52.10.254.61	192.168.1.8	TCP	54	443 → 56531 [ACK] Seq=32 Ack=36 Win=118 Len=0
21	11.005118	142.93.181.168	192.168.1.8	TLSv1.2	85	Application Data
22	11.005423	192.168.1.8	142.93.181.168	TLSv1.2	89	Application Data
23	11.021440	142.93.181.168	192.168.1.8	TCP	54	443 → 56545 [ACK] Seq=32 Ack=36 Win=33 Len=0
24	14.986506	192.168.1.8	10.36.169.121	ICMP	74	Echo (ping) request id=0x0001, seq=805/9475, ttl=128 (no response found!)
25	18.377603	192.168.1.8	52.177.166.224	TLSv1.2	97	Application Data
26	18.394067	52.177.166.224	192.168.1.8	TLSv1.2	179	Application Data
27	18.439935	192.168.1.8	52.177.166.224	TCP	54	56567 → 443 [ACK] Seq=44 Ack=126 Win=258 Len=0
28	29.988345	192.168.1.8	10.36.169.121	ICMP	74	Echo (ping) request id=0x0001, seq=806/9731, ttl=128 (no response found!)
29	32.051536	192.168.1.8	172.217.7.238	TLSv1.2	93	Application Data
30	32.052183	192.168.1.8	172.217.7.238	TLSv1.2	78	Application Data

Collected through WireShark (to get an idea of the network traffic and data size)

Future Upgrades

The collection of cell traffic. This will allow the identification of devices connected to both the network and cellular data. This situation enables communication that are not filtered by the network firewall protocols. Another addition in the future will be a line of sight detection protocol. This will allow the user to look for signal scattering and know if the device is in a direct line of sight. The addition of deployable remote stations in in development for future additions. The architecture being based on the Arduino WiFI package. There is also a design being worked on to attack common encryption used in wireless communications. (Note: this uses a lot of math I do not understand, I am only putting in the interface)

Synchronize Data

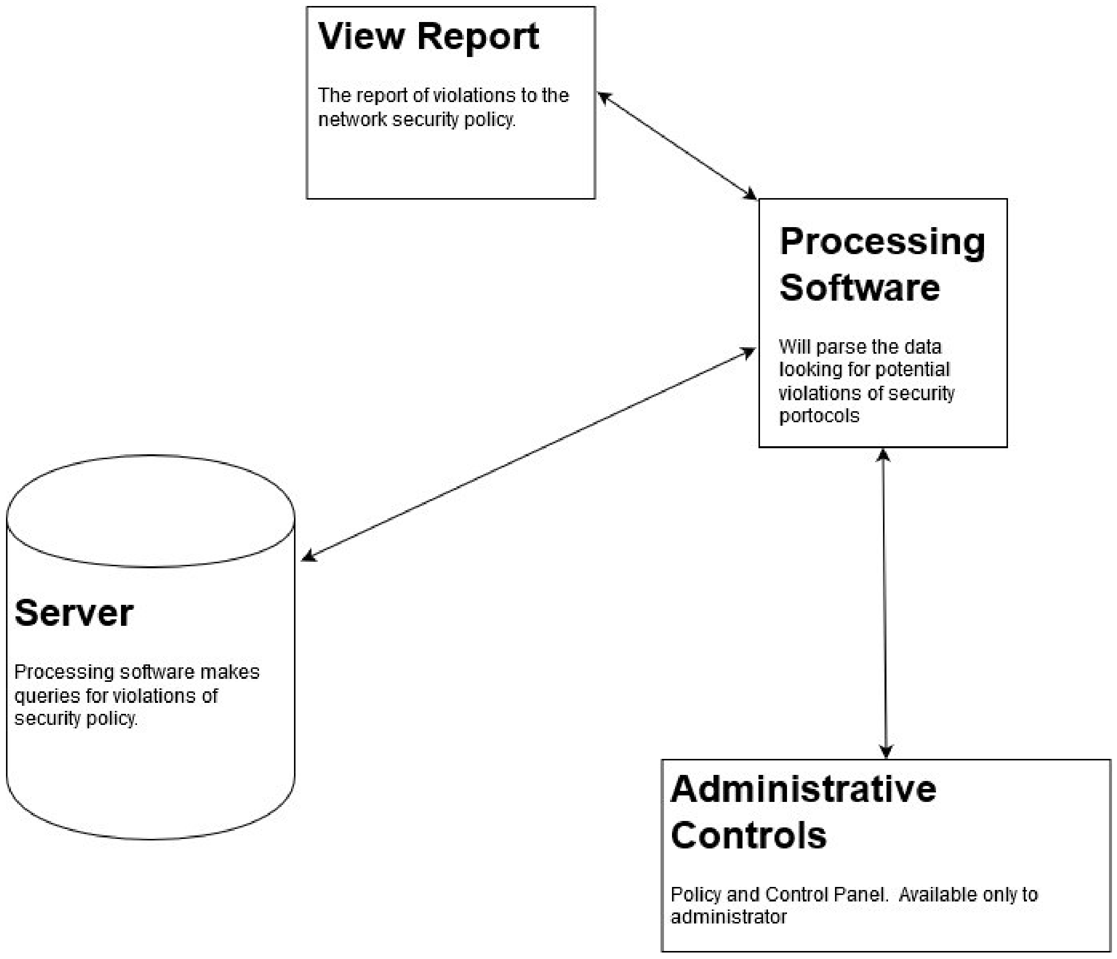

Synchronize Data
Sends data from the remote server for storage.
Local JSON File
Processing Software
Will parse the data looking for potential violations of security portocols
Server
FireBase Database that will house the collected data form the mobile devices
Administrative Controls
Policy and Control Panel. Available only to administrator

The goal is to be able to store the collected information and to process the data into something meaningful. It is this system that will need the strictest security policies, since it will have the repository of traffic data. The design principle will follow the “Least Privilege” model. The processing that will be implemented will add the owner and signature of the sites being visited. So for example:

www.gmu.edu 129.174.1.59 SSL Certificate Issued by InCommon RSA Server CA

The server can authenticate the IP as belonging to George Mason University. The certificates were updated in September. The site is not on a black list. This site would end up on a log just in case of an incident (Our version of the circular filing cabinet). The location would not be flagged.

View Report


View Report
The report of violations to the network security policy.
Processing Software
Will parse the data looking for potential violations of security portocols
Server
Processing software makes queries for violations of security policy.
Administrative Controls
Policy and Control Panel. Available only to administrator

After the data has been processed the report of violations can be requested by the user. The report will be of the form of date, location, MAC, IP Violation. Administrative controls will set the policies, and update information. The ideal situation is a live update on black list sites, and traffic monitoring. This first design is to be more of a proof of concept, and the data will be of a best effort form. In the future, modules will be replaced to improve the performance of the system.

Hunter

Start Date: 9/12/2019

Objective:

The goal of this project is to produce a tool that will allow for the analysis of the dynamic nature of WiFi forests. On a campus or in an organization it is often difficult to define your perimeter, or even the nodes on your network. The very nature of the network is dynamic and in a constant state of change. The tool will be broken into two distinct parts. The first being the hunter. Its role is to locate and tag all of the nodes in the network. This will be accomplished through the tagging and locating of the nodes to a geographical region. This is where the use of mobile applications will come in. A mobile device will be used to sniff out wireless packets and tag them to a geographical location. This data will then be stored and uploaded to a remote server for analysis. From this point the second portion of the program takes effect. This program can then be used to analyze the data and produce a baseline map of the environment. From this baseline, an analysis can be made of the nature of the network and requirements for locking down the network. This information is then be passed back to the system for local investigation. Through this dynamic analysis of the network a better security model can be constructed.

The first component of the application is the sniffer element. The mobile application will go into promiscuous mode, allowing for the collection of traffic conforming to the IEEE 802.11 protocol. Instead of only looking at traffic that is meant for the mobile device, the device collects all transmissions in the area. From using the 802.11 protocol the application can extract information from these transmissions. The application will also be testing a prototype algorithm that will allow for hidden, and out of sight nodes.

In the above image the sniffer does not have the range on its own to reach the hidden user. Through the exploitation of the IEEE 802.11 protocol, the algorithm will attempt to use the router and the users own device to bridge the gap. The algorithm will not require that the hidden user's device be connected to the router being used. In addition this algorithm can be reversed so the hidden user who is using a private network can be exploited to expose the existence of the hidden access point. It is hoped that this algorithm will be compatible for integration with an existing tool called Hydra. If these tools can be integrated then the effective range of the system would cover the entire inter-networked area. This would mean that on the GMU campus IT could monitor for problems on everyone's devices for the entire campus. (This is of course not an objective of the project, for practical reasons. Bandwidth requirements, invasion of privacy, and Sub-netting). The primary goal of the first part of the application is the collection of information for the second part of the application.
Processing of this data is the next step of the application pipeline. This data is grouped into sort-able groups and processed. The information will be of the following structure:

{ MAC Address, GPS Location, {Were they went}, {Security Protocols}, {Security Violations}, {links to other contacts}}

The MAC address is the key that will be used to sort the devices by. This provides also a unique way of identifying users. The GPS location, will provide a type of home range for the device. This will also allow the user to identify known devices seen in the wild (ie. not on campus. Is this a device that a user brings from home? Does he/she only have it at work? Is the device a professors laptop, that never leaves the lab?) . While these questions on the surface do not seem to be important, the nature of the security that is needed to protect the devices differs with the way it is used. The next piece of information serves two purposes. The first is to notify the administrators of accesses to black listed locations (malware servers, pirate servers, inappropriate locations, and to link with the protocols used). The protocols is to identify nodes in the network that might need correction on security protocols. For example, going to a bank site and sending credentials un-encrypted. This information is then collected and added to the security violations, where it can be retrieved through a SQL search. This information is then processed and returned back to the mobile device as both a report, and tags.

This leads to the shared aspect of the project. Eventually the goal is to make this a distributed system. Deployable sensors can be logging nodes, and transmitting data to servers for analysis. This information can then be disseminated to company network administrators to take action. The quicker this loop in information can take place, the shorter the window for unauthorized access. There is also a plan for the integration of network security tools into the security package.

At this point the tags, can be used to track down the devices that need to be addressed. Through this process a better security model can be developed. The first element of security is to know what you have. In a dynamic network this is a challenge. Only when you know the environment being dealt with can you develop a plan to protect it. There is also an additional side effect of this project. If people know what kind of information that they broadcast, they might take greater measures to secure the data. The primary goal of this system is to be able to detect and act on threats to the security of the network. The concept is to have a lightweight and mobile system, that can shift with the changing needs of the industry. In effect the searches on the database can be updated on the fly, to reflect the changing face of security. The reports and data flowing to the administrative staff can be changed at a server level allowing for quicker changes to the system. The real challenge of this system will be making tools developed for running on a laptop, and porting them down to an android device.
The goal is that a sensor or audit device will detect a violation of security policy, and forward it to the database. The database detects the violation and forwards it to the network administrator. The administrator quarantine the node, and takes action. This action is supported now by the sensors allowing the administrator to know the location of the offending device. This is where the name Janus comes from. Janus was the Greek god of portals and decisions. The program is to be a portal into the shifting nature of the wireless network, and make decisions to secure the network.

The testing of the final application will have two parts. The first will center around unauthorized users on the network. I am going to recruit a few people to play the part of unauthorized users. The goal of the system is to detect them and to locate them on campus. He next phase will be to define a set of websites, and actions as security violations. Then the goal will be to identify nodes on the network that are committing these violations. This will take the form of listing of a site like Amazon as a black listed server, and detecting nodes accessing the server.