Argus

By

Benjamin Fraley, Tyler Hopperton, Tyler Jones

Submitted to
The Faculty of the School of Information Technology
In Partial Fulfillment of the Requirements for
The Degree of Bachelor of Science
In Information Technology

Benjamin Fraley	4/17/17 Date
Tyler Hopperton	4/17/17 Date
Tyler Jones	4/17/17 Date
Bogdan Vykhovanyuk	4/17/17 Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services

April 2017

Table of Contents

Table and Figures

Acknowledgements

The team at Argus would like to thank all those involved in the University of Cincinnati Senior Design Class of 2017. More specifically, the professors who devote countless hours of teaching, revising, assisting, and providing feedback on each project proposal. Without the assistance and contributions of the faculty, staff, peers, and friends, Argus would not be what be what it is today.

Abstract

Every day, our personal devices communicate with many different networks and other devices. Ordinarily this does not cross anyone's mind, however the circumstances change when your device could be interacting with someone or something that has malicious intent. Often, users are not sure who or what their devices are talking to. Argus saw an opportunity to help secure our devices and empower the users to gather intelligence on personal devices and their communications. Argus is a network analysis tool that is cross-platform and written in Python. Argus remotely captures packets and collects them for analysis. By allowing users to understand who or what their devices are talking to, we can help protect them and ensure their information is safe.

Introduction/Description

Argus is designed to be a simplistic tool to help anyone gather network information about their networks and devices. The tool allows users to gather accurate, real-time information about the device the tool is running on. It does not require any major changes to existing infrastructure and can be scaled to fit any size company. Similar solutions are very rare, those that do exist are not user friendly. Our solution is one that is light-weight, user friendly, and incredibly versatile to fit whatever is needed.

Design Objectives

Argus, founded and developed in Cincinnati, Ohio, is a company focused on remotely capturing packets. Argus's clients comprise both home users and small-large organizations. When a user believes they are being maliciously targeted online it is crucial that results are fast and accurate, so Argus is intuitively simple to use.

A problem that most companies and users face often revolves around missing information or important context when trying to solve an issue. Argus aims to end that age-old problem. Argus allows any user to immediately and accurately see with what or whom their devices are communicating with. This can help an incident of a suspected compromise or companies looking to monitor network traffic for abnormalities. This information will better help IT personnel when attempting to solve problems.

User Profile

Argus is designed to be simple to use and require user interaction from as little people as possible. Table 1 details all necessary interactions between Argus and those using the tool.

User Profile Form – Help Desk Technician
Use: Distribute the tool to users when the internal Help Desk process deems it necessary
Software and Interface Experience: Argus will be delivered via a single script that will run invisible to the user. The user only needs to be familiar with executing a file or program.
Experience with Similar Applications: Help Desk technicians are already comfortable interacting with the end user, whether it is walking them through how to complete a task, remoting into their system, or providing troubleshooting tools for them to run. Argus will seamlessly integrate into the Help Desk process, as it will be delivered via a single script that needs only to be ran by a user. The technician, through whatever process is already in place to interact with the end user, will provide them with the script, which as far as the interaction need go.
Frequency of Use: Argus is meant to be used on an as-needed basis. When the technician determines that the analysis of network traffic will benefit the investigation of a trouble ticket, they need only to distribute it to the end user.
User Story: Steve receives a call from a user complaining that their computer is acting strangely. After talking with the user, Steve is able to gather that the user went to download malware.totallylegit.evil and their computer subsequently began running slowly. Being that the user is working remote today, Steve, via their software repository, sends the user Argus and instructs them to double-click the file. Steve explains to the user that the security team will now have everything they need to determine what is wrong with their computer, and will be able to assist the user further.

Table 1: User Profile

User Profile Form – Security Analyst
Use: Monitor back-end server.
Software and Interface Experience: The back-end component of Argus will store packets in PCAP format viewable by any PCAP-aware tool. The security analyst will need a familiarity with Linux directory transversal, as well as their preferred packet analysis tool.
Experience with Similar Applications: Security analysts are familiar with network analysis with packets. Argus is meant only as a tool to gather such packets from a new source, and will not affect how the security analyst interacts with them.
Frequency of Use: Argus is meant to be used on an as-needed basis. The security analyst will only need to take action when they are notified that packets are coming in, with an explanation of what they are looking for.
User Story: Sarah receives a call from the Help Desk stating a user called in with computer issues, and believes the computer may have been compromised. The Help Desk technician provided the user Argus, and tells Sarah that it is currently running on the user's computer. Sarah checks the Argus collection server to find a new source, and begins investigating for potential causes.

Table 1: User Profile

Use Case Diagram

Argus is built to be extremely user friendly for all parties involved. Figure 1 shows how Argus can be utilized to assist an organization troubleshoot an employee's computer problems.

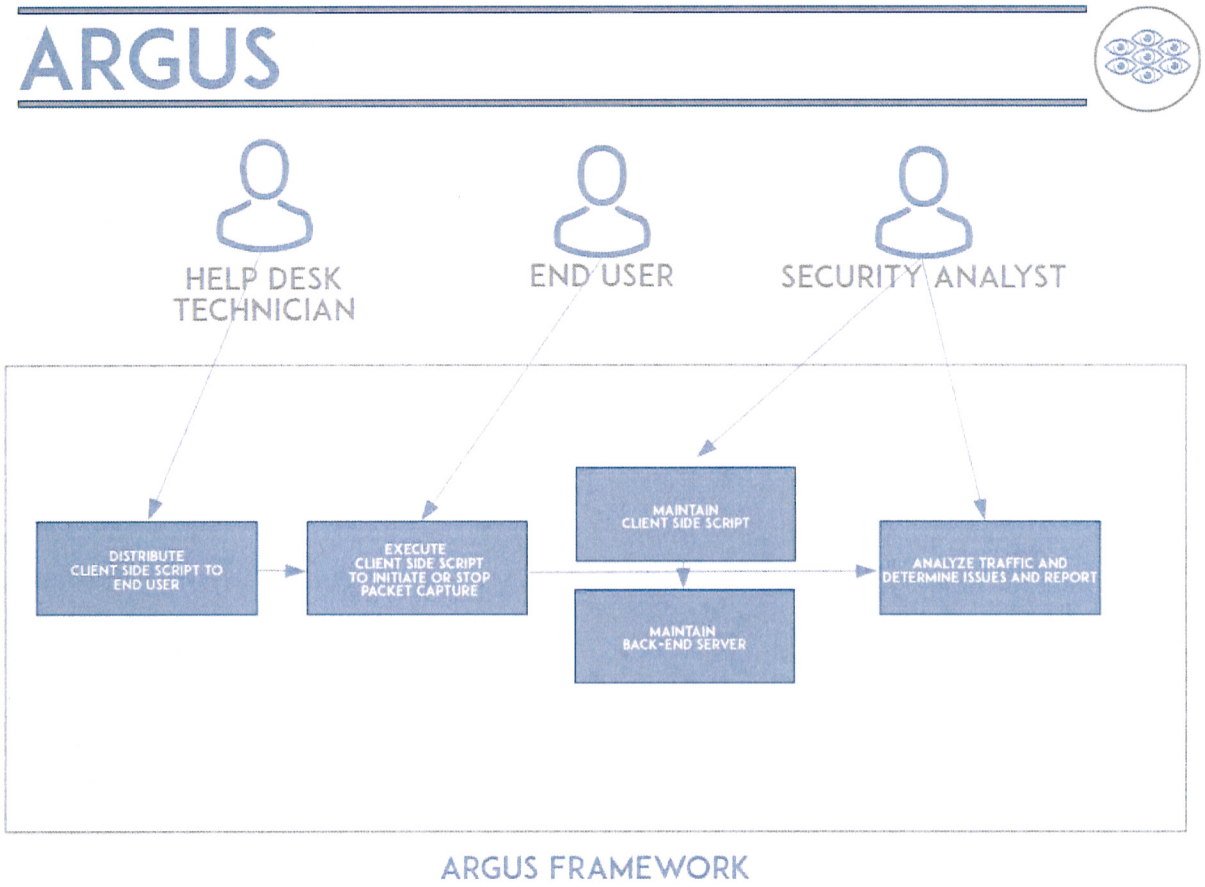


Figure 1: Use Case Diagram

Budget

The budget in Table 2 reflects the real-world costs of planning and executing this project. Parts and labor and other factors are considered to give an accurate representation of the cost of this project.

No.	Item	Units	Unit Price ($)	Line Item Total
Hardware/Software				
1.	**Parts (Client and Server)**	**2 (Client and Server)**	**$75**	**$150**
2.	**Networking**	**~4**	**$10**	**~$40**
3.	**Misc.**	*	*	**~$200**
Labor				
1.	**Labor**	**240 (hours)**	**$50/hr.**	**$12,000**
Total:				
				$12,390

Table 2 Budget

Testing

Overview

This section will explain the testing methodology for Argus and should be followed as a guide. The following individuals should utilize this section:

- Project Managers
- Developers
- Team Members

Scope

The scope of testing is to test the operation of Argus on Windows, Linux, and Mac OS X. The test will be organized based on the requirements of Argus.

Objective

The goal of testing is to verify that Argus is functioning as expected. These tests are designed to test components of Argus, and the entirety of Argus. Developers will run the tests for the components they are working on.

Testing Plan

Using the Agile Methodology, we test regularly throughout the development process. Every change, major or minor, is tested for accuracy. Argus is tested using our own equipment and systems, and volunteers.

Windows Testing

Functional Requirements

1. Once Argus is installed, it should be able to check the endpoint system for the presence of a packet capture program (WinPcap).
2. The Back-end Server should capture the packets of the remote endpoint system.
3. The packet captures should be stored and sorted in the Back-end Server.
4. Argus should run silently on the endpoint system.

Req. #	Input/Action	Expected Outcome	Actual Outcome	Pass/Fail	Explanation
1	Install Argus and attempt to run	Argus runs smoothly and correctly because it was able to locate WinPcap on remote endpoint system	Argus ran smoothly when WinPcap was installed on remote system	Pass	When WinPcap is installed on the endpoint, Argus works as expected
2	Security Analyst attempts to view packet captures	Packets are saved and viewable	Packets were viewable from the Back-end Server	Pass	The packets were able to be viewed
3	Security Analyst attempts to locate packet capture from specific IP	Packets are saved in Back-end Server with unique filenames	Packets are viewable in real time but were not saved	Fail	The packets were not able to be saved off automatically
4	Help Desk technician wants to run Argus without interrupting end user	Argus runs silently	Argus ran silently	Pass	Argus ran with no sign of it doing so

Table 3 Windows Functional Requirements

Linux/Mac OS X Testing

Functional Requirements

1. Once Argus is installed, it should be able to check the endpoint system for the presence of a packet capture program (libpcap).
2. The Back-end Server should capture the packets of the remote endpoint system.
3. The packet captures should be stored and sorted in the Back-end Server.
4. Argus should run silently on the endpoint system.

Req. #	Input/Action	Expected Outcome	Actual Outcome	Pass/Fail	Explanation
1	Install Argus and attempt to run	Argus runs smoothly and correctly because it was able to locate libpcap on remote endpoint system	Argus ran smoothly when libpcap was installed on remote system	Pass	When libpcap is installed on the endpoint, Argus works as expected
2	Security Analyst attempts to view packet captures	Packets are saved and viewable	Packets were viewable from the Back-end Server	Pass	The packets were able to be viewed
3	Security Analyst attempts to locate packet capture from specific IP	Packets are saved in Back-end Server with unique filenames	Packets are viewable in real time and able to be saved	Pass	The Packets were able to be saved automatically
4	Help Desk technician wants to run Argus without interrupting end user	Argus runs silently	Argus ran silently	Pass	Argus ran with no sign of it doing so

Table 4 Linux/Mac OS X Functional Requirements

Schedule

Testing is an important component to ensuring Argus can operate as intended. The table below details the testing schedule Argus went through to ensure all use cases have been met.

Team Member	Timeline to be Completed	How Often?
Project Manager	10/17/2016 to 04/10/2017	Weekly
Developer	10/17/2016 to 04/10/2017	Weekly
Team Members	10/17/2016 to 04/10/2017	Weekly

Table 5 Schedule

Gantt Chart

Argus took some time and resources to be fully developed and deployed. Figure 2 lays out the timeline and the objectives that were completed to prepare Argus for IT Expo.

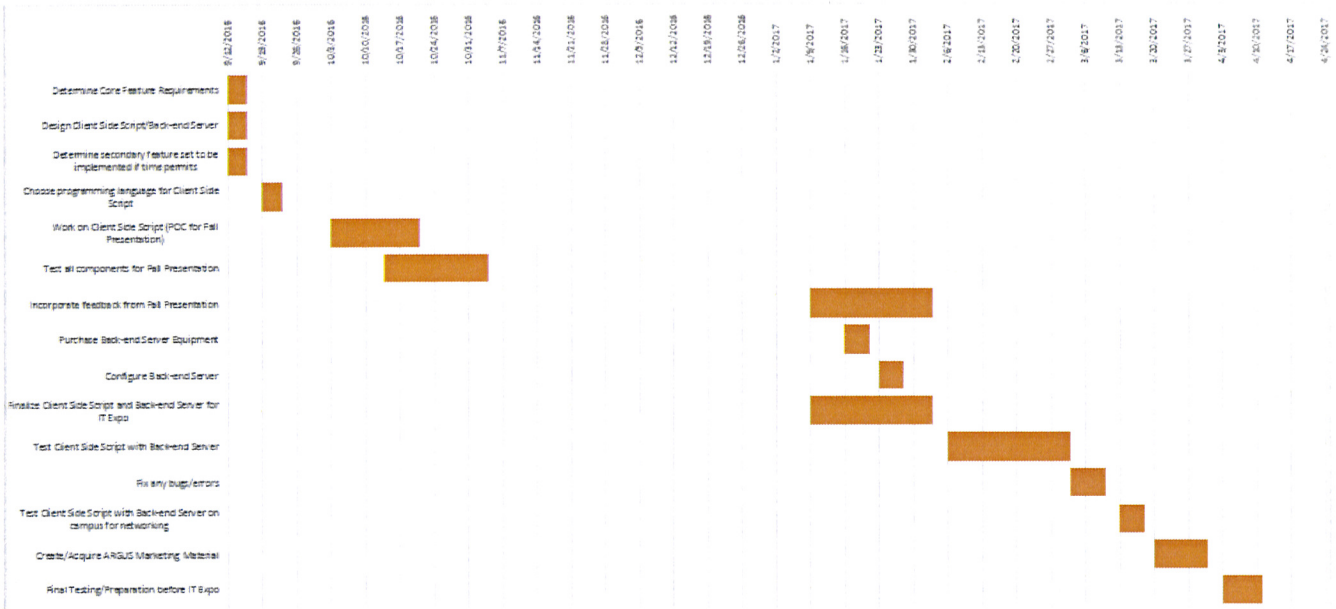


Figure 2: Gantt Chart

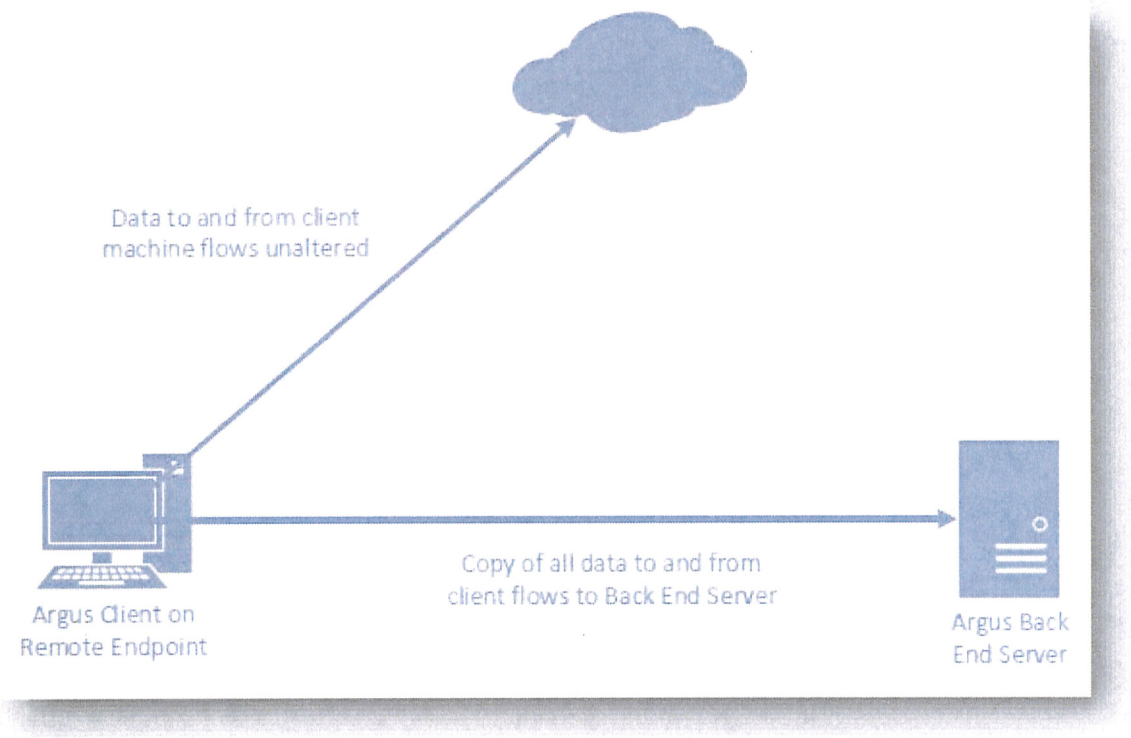
Conclusion

Throughout the entirety of this process the Argus team has learned many different things. We learned a lot about programming in Python and scripting. Specifically, we learned about the overall flexibility and versatility of the Python language. We learned a great deal about project management, team work and working in a real world environment. As a team we also had overcome the challenge of developing our own budget and continually reaching milestones. Overall it was an incredibly positive learning experience and we are not only better IT professionals but better people because of our time spent on Argus.

Work cited

1. Top 10 Metrics to Monitor the Health of Your Hel (n.d.): n. pag. Zendesk.com. Zendesk.com. Web. 6 Nov. 2016.
2. "IBM | Ponemon Cost of Data Breach 2016." IBM | Ponemon Cost of Data Breach 2016. IMB, n.d. Web. 06 Nov. 2016.
3. Lutz, Mark. Programming Python. Sebastopol, CA: O'Reilly, 2006. Print.
4. Chappell, Laura A. Wireshark 101: Essential Skills for Network Analysis. San Jose, CA: Protocol Analysis Institute, Chapell U, 2013. Web. 8 Nov. 2016.

Appendix



Argus Diagram 1

ARGUS

BENJAMIN FRALEY, TYLER HOPPERTON, TYLER JONES

COLLEGE OF EDUCATION, CRIMINAL JUSTICE & HUMAN SERVICES
SCHOOL OF INFORMATION TECHNOLOGY
ADVISOR - BODGAN VYKHOVANYUK

40-70 HOURS
BETWEEN DISCOVERY AND RESPONSE

LACK OF INFORMATION
DURING AN INCIDENT

NO PHYSICAL ACCESS
TO AFFECTED SYSTEM

205 DAYS UNTIL
COMPROMISE HAS BEEN DISCOVERED

CURRENT SOLUTIONS:
EXPENSIVE, UNWIELDY, COMPLEX

ARGUS IS A REMOTE PACKET CAPTURE UTILITY
THAT IS CROSS-PLATFORM AND DESIGNED TO GIVE INSIGHT ON
WHO/WHAT DEVICES ON YOUR NETWORK ARE COMMUNICATING WITH

INTIAL INCIDENT OCCURS

ARGUS IS ACTIVATED AND CREATES AN ALERT

PERSONELL SEVER THE CONNECTION

TECHNOLOGY USED:

UNIVERSITY OF Cincinnati

Benjamin Fraley, Tyler Hopperton, Tyler Jones

ARGUS

- Problem
- Solution
- Technology
- Demonstration
- Benefits
- Conclusion

Problem

40-70 Hours
Between Detection and Response

Lack of Information
During an Incident

No Physical Access
To Affected System

205 Days Until
Compromise has been Discovered

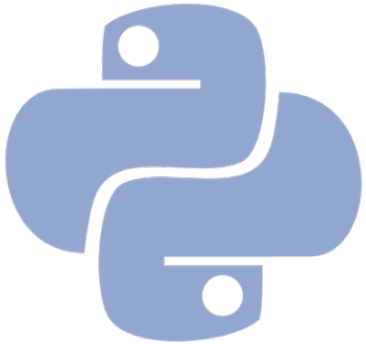
Current Solutions:
Expensive, Complex, Unwieldy

Solution

Argus-

Argus is a remote packet capture tool written in Python and designed to be lightweight and cross platform. Incredibly pliable and robust in its uses. It can be used by help desk technicians, data analysts, security personnel etc.

Technology



Demo

Benefits

Quicker time to resolution for incidents

Massive quantities of real-time network information

Help reduce number of large scale incidents and cut costs

Increased Incident Reporting

Conclusion

-Scope

-Initial goal

-Argus opportunities

Questions?