[British Columbia Institute of Technology]

[Android Security Suite]

[Major Project Proposal]

Name:	Jivanjot S. Brar
Student ID:	A00774427
Instructor:	Aman Abdulla
COMP8045:	Major Project
Date:	March 5, 2015

BCIT Bachelor of Technology Program

Network Security & Administration

CONTENTS

1 | CURRICULUM VITAE

I. FORMAL EDUCATION

My formal education has been fragmented between countries. My early education took place in India and my secondary and post-secondary education took place in Canada. Upon completing my grade 12, I decided to enrol at BCIT with the intention of completing the Bachelor of Technology Program.

2013 – 2015	**British Columbia Institute of Technology** Bachelor of Technology Program - Network Security & Administration
2010 – 2012	**British Columbia Institute of Technology** Diploma of Technology Program - Technical Programming Option
2005 – 2010	**Sands Secondary School** High School Diploma (Grade 8 – 12)

II. WORK EXPERIENCE

My work experience in the field of Computer Systems has been very little; therefore I decided to go to school in order to further my education in the field of technology. It is also the reason why I have designed a practicum that draws from my interest in programming, network security and mobile development and provide me with some useful experience in each individual field.

2014 - present **DNN Corp.**

Software Support Analyst

- Providing answers to clients by Analyzing and identifying software problems; researching solutions; guiding clients through a series of actions over the telephone or email to help resolve issues or help install and configure software.
- Work continuously on a task until completion (or escalation of ticket to Tier 2 or Tier 3 for further assistance).
- Prioritising and managing many open cases at one time.

2013 - 2013 **XModus Software Inc.**

Contract Web Developer

- Converted PSD files into HTML/CSS pages
- Created interactive plugins such as sliders, slideshows using JavaScript
- Created store locator plugin using JavaScript and PHP.
- Provided HTML/CSS/JavaScript bug fixes

III. AREAS OF SPECIALIZATION

During my education at BCIT, I have learned numerous things. I have always had a passion for programming and always enjoyed learning about new techniques and exploits' hackers or security experts used to compromise a network or a system. Before my education at BCIT, I learned everything from trial and error, but going through Bachelors of Technology program has really polished me as a programmer and gave me a whole new set of skills and knowledge.

I am currently nearing the end of my Bachelors of Technology degree at BCIT; with a dual option in Technical Programming and Network Security. It is with these specializations and my interest in development that I have designed a practicum that draws from each area of expertise and fully make use of what I have learned during my time with the program.

2 | PROJECT INFORMATION

This practicum is primarily an exploration of Android OS with respect to network penetration testing security suite. The main functionality of this suite for the purpose of the practicum will be to provide users with few common security tools that can be used to compromise system; gather information on the network; or craft and send custom packets to analyze their response to the packets.

Throughout my exploration of this practicum, I will attempt to implement the following major features in the Security Suite:

- Network Scanner – scans the network and returns with a list of available hosts on a network, list open ports on a target host.
- ARP Spoofer – used for spoofing the router and (single or multiple) victim machines on a network.
- DNS Spoofer – used for intercepting DNS requests from victim machines and feeding them false information, redirecting them to a fake server.
- TCPDump – used to capture network packets both on mobile and wireless network.
- Packet Crafter – send custom packet to a destination device and capture the response packets.

I. BACKGROUND INFORMATION

A penetration test, or the short form pentest, is an attack on a computer system with the intention of finding security weaknesses, potentially gaining access to it, its functionality and data. (Stephen Northcutt, Jerry Shenk, Dave Shackleford, Tim Rosenberg, Raul Siles, and Steve Mancini;, 2006)

The process involves identifying the target systems and the goal, then reviewing the information available and underlying available means to attain the goal. A penetration test target may be a white box or black box. A penetration test can help determine whether a system is vulnerable to attacks.

Penetration tests are valuable for several reasons:

1) Determining the feasibility of a particular set of attack vectors.
2) Accessing the magnitude of potential business and operational impacts of successful attacks.
3) Testing the ability of network defenders to successfully detect and respond to the attacks.
4) Identifying vulnerabilities that may be difficult or impossible to detect with automated network or vulnerability scanning software.

Network Security experts such as penetration testers use variety of security tools to perform penetration testing on a Network. Some of these tools are specialized OS Distributions, which are geared towards performing penetration testing. Distributions typically contain pre-packaged and pre-configured set of tools (Kali Linux Tool Listing). This is useful because penetration tester does not have to hunt down a tool when it is required. Popular examples of OS distributions are Kali Linux, Pentoo, and WHAX etc. Some popular examples of software tools include Metasploit, nmap, w3af and many more.

There are various types of exploits or attacks that can be used by the penetration testers and these attacks are divided into two categories: Active and Passive attacks. Passive attacks are when network intruder intercepts data traveling though the network, and Active attacks is in which an intruders initiates commands to disrupt the network's normal operation.

Types of attacks include: (Network security, 2014)

- Passive
  - Wiretapping / Packet Sniffing
  - Port Scanning
  - Idle Scan
- Active
  - Denial-Of-Service (DOS) attack

- Spoofing
- Man In The Middle
- ARP Poisoning
- Buffer Overflow
- Stack Overflow
- SQL Injections
- Brute Force Password Guessing

Above mentioned OS distributions, software's are only made for desktops or laptops. For this project, I would like to a take a step further and develop this functionality for a fully mobile platform, Android. I chose Android over other devices because Android's base is Linux which theoretically makes building these applications possible.

II. ASSUMPTIONS

There are few assumptions that development of these Security Tools on Android device is depended on and I have listed these assumptions below.

1) Development of all these application in the suite is based on single and most important assumption that all these Android Applications will run in a similar manner as they do on a Linux machine. The basis of this assumption is that Android Kernel is based on one of the Linux Kernels. Android Operating System uses an underline Linux Kernel in order to access and utilize its hardware components. (Android Operating System)
2) The application will require root access to the device for proper execution. This assumption is derived from the similar behaviour on a Linux machine where application would require a root access in order to access the network card capabilities for packet capture and custom packet crafting purposes.
3) Android device will support frameworks for Packet Sniffing and Packet Injecting.
4) The application will be able to capture packets on a Wi-Fi Network.
5) The application will be able to capture packets on a Device Data Network.
6) The application will be able to inject packets on a Wi-Fi Network.
7) The application will be able to inject packets on a Device Data Network.

III. SCOPE

The application consists of 5 sub-applications: Network Scanner, TCPDump Packet Scanner, Packet Crafter, ARP Spoofer and DNS Spoofer. I have scoped the project down to the basic execution of the 5 sub-applications and main wrapper application. In this section I have described features of individual sub-applications.

1. Network Scanner
    a. Host Discovery – Identifying hosts on a network. For Example, listing the hosts that respond to TCP and/or ICMP requests or have a particular port open.
    b. Port Scanning – Enumerating the open ports on target hosts.
    c. Version Detection – Interrogating network services on remote devices to determine application name and version number.
2. TCPDump Packet Capture
    a. Packet Analyzer that runs under the command line.
    b. Allows user to capture and filter packets over the network.
    c. Displays the captured packet and their information.
    d. Reads captured packet information from a PCAP file.
    e. Writes captured packet information in a PCAP file.
3. Packet Crafter
    a. Tool for auditing Firewalls and networks.
    b. Create and Send custom TCP packets
    c. Create and Send custom UDP packets
    d. Create and Send customer ICMP packets
    e. Captures response packets
4. ARP Spoofer
    a. Sends packets to one Host (Default Gateway) and one Target Machine
    b. Sends packets to one Host (Default Gateway) and many Target Machines
    c. Combines with TCPDump to capture the packets.
    d. Combines with TCPDump to Filter incoming packets
5. DNS Spoofer
    a. Combines with ARP Spoofer to capture packets over the LAN.
    b. Filters incoming DNS Inquiry packets with specific URLs or Addresses
    c. Filters all incoming DNS Inquiry packets
    d. Creates custom DNS response packets

IV. INNOVATION

The applications developed in this project are used as penetration testing tools designed to test strengths of network defense tools such as firewalls etc. Penetration testing tools help determine the weak points of the network and also help testers determine the feasibility and a magnitude of an attack on the network. The tools discussed in this proposal are currently part of Operating System such as Kali and can also be manually installed in Linux OS. However these applications are currently limited to a desktop environment only.

My Innovation for this practicum project involves introduction of penetration testing tools in a mobile platform specifically Android. A penetration testing framework contains many tools, for this project I have focussed on some of the common tools such as sniffers, crafters and scanners.

Android currently lacks frameworks for developing low level networking tools; this project will help me lay down a framework for low level networking tools for Android and will allow me to pave a path for creating a penetration testing or pentesting suite for Android devices.

V. TECHNICAL CHALLENGES

This project allows me to utilize and build upon everything that I have learned in all the Network Security courses. Even though Android's open platform allow us to build these pentesting tools, lack of any previous projects present me with a few extremely difficult technical challenges. Challenges for this project include but aren't limited to following:

- Learning how to develop an Android Application. I have no prior experience for building Android application therefore this aspect of the project is going to be one of the major challenges that I will be facing while working on this project.
- Second most challenging aspect of the project is locating & using appropriate Android Libraries to achieve access to low level resources such as the network card & network stack etc. These libraries are crucial for this project and if no Java Networking Libraries can be located for accessing low level Android resources, project will require following a difficult path for using Native C or C++ code for achieving this functionality.
- If the project does rely on using Native C or C++ code, this will raise another challenge for learning how to develop Android application using Android NDK and running native code in Android.

Above are some of the most difficult challenges that are presented by this project, however overcoming these challenges will provide me with immense experience in Android Application development and also in Network Security application development.

VI. METHODOLOGY

The exploration of this practicum can be divided up into several discrete modules, which can then be integrated. The following sections will briefly outline each module and its function.

1. Packet Capturing

Capturing network packets is an integral part of this application and is used by all 5 sub-applications. This module in application will be responsible for capturing any and all network packets that arrive at the network card of the device. It will then filter the packets based on the sub-application it's incorporated into. The difference here will be the packet filter applied by the sub-application. For example DNS Spoofer will capture all packets however it will only display any incoming DNS packets. Similarly TCPDump will allow user to apply a custom filter which will then be passed to this module and this module will only return the packets that matches the specified filter by the sub-application.

What this means is that this functionality will only be developed once and will then be incorporated into each sub-application and therefore reducing the duplicate code and also reducing the effort to test and fix any bugs.

2. Packet Crafting

One of the other main features of this application is the ability to create and send custom packets over the network. This ability create custom packets is used by all the sub-applications except TCPDump. This feature involves creating custom packets for different protocols such as IP, TCP, UDP, ICMP, DNS, and ARP etc. The purpose of this feature is to allow the application to create custom packets in order to retrieve information about the network.

This module will work alongside with Packet Capturing module, in order to capture the response from the target machines over the network, or to respond to incoming packet from target machine to feed false information. For example DNS Spoofer captures incoming DNS request packets from machine(s) over the network and then sends a false DNS response packet in order to redirect the traffic to intended destination. Similarly Network Scanner sends various TCP, UDP, ICMP packets in order to get more information about the network such as Machine IP addresses, open TCP & UDP ports, services running over the ports of different machines over the network etc.

3. Packet Traversing

The application has the functionality to create & send custom packets and also capture incoming packets. This module will further allow the application to parse the information from the captured incoming packets and display the information for

the user in a readable format. This module will be used by Network Scanner and TCPDump. TCPDump would parse the captured packet to display the user common information such as Source IP & port, destination IP & port, and packet data. Network scanner parses information to retrieve information such as identifying host machines over a network, identifying open ports on a target host, determining the operating system of the target hosts on the network, applications that are running on the target hosts on the network etc.

4. Prototyping

Given the exploratory nature of this practicum, I will be going with an incremental software prototyping approach to development. Small-scale mock-ups of each part of the application outlined above will be developed following an iterative modification process until the prototype evolves to meet the application requirements.

For instance, the first task would be to determine whether assumptions are valid as the completion of the project depends entirely on their validity. This test would require creating a simple packet capturing application in Android and using a stock (non-rooted) device to test the application & similarly also using a rooted device to run the application. If the application runs on the non-rooted device this will determine that there will be no need for a device to be rooted and the application will run on all Android devices. If the application only runs on the rooted device, the application will only be targeted for rooted devices and will also help me determine whether this project can be finished. This will then expand into developing basic functionality of different sub-applications. However, because of the exploratory nature of this practicum, a rigid development and integration plan is not feasible. This is why I will be focusing on creating iterative prototypes, slowly adding features to each iteration until a base level of functionality has been achieved.

VII. TECHNOLOGIES USED

There are various technologies that are going to be used for this project both at the software level for development and hardware level for testing. In this section I will be listing all the technologies that I am going to be using for this project.

1) Operating Systems
    a. Windows 8.1
    b. Fedora Linux
2) Development Tools
    a. Android Studio
    b. Eclipse with Android Developer Tools
    c. Bash Shell
3) Hardware Testing Tools
    a. Nexus 9 Tablet
    b. Nexus 5 Phone
    c. Linux Machine
4) Software Testing Tools
    a. Android Emulators
    b. Wireshark
5) Programming Languages
    a. Java with Android SDK

VIII. TESTING PLAN

The Security Suite application consists of 5 sub-applications. The testing plan for this practicum project includes a set of test cases for each of the sub-applications. In this section I have included set of example test cases for each of the sub-applications. The test cases include a test description, expected result, actual result and whether a test passed or failed. Final Report will include many more test cases that go through the all the features of each sub-application.

The testing format in the Final Report will be similar to the format defined in this section, however two more columns will be added in the report which will include detailed information of the actual results and whether the test passed or failed.

1) Network Scanner

#	Test Case	Expected Results
1	Get a list of open TCP Ports on a Target Machine	The scanner will return port numbers that are open on the target machine. These ports only responded to TCP packets.
2	Get a list of open UDP Ports on a Target Machine	The scanner will return port numbers that are open on the target machine. These ports only responded to UDP packets.

2) TCPDump Packet Capture

#	Test Case	Expected Results
1	Capture all TCP packets	The application captures and displays all TCP packets incoming or outgoing
2	Capture all UDP packets	The application captures and displays all UDP packets incoming or outgoing

3) Packet Crafter

#	Test Case	Expected Results
1	Create a TCP packet for a target machine on port 80	The application creates and sends TCP packets for a specified target host with source port being 80. Since number of packets are not specified packets are sent every seconds until the process is

		manually killed.
2	Create a UDP data packet for a target machine on port 1024	The application creates and sends UDP packets for a specified target host with source port being 1024. Since number of packets are not specified packets are sent every seconds until the process is manually killed.

4) ARP Spoofer

#	Test Case	Expected Results
1	Send ARP packets to a single Target Machine	The application creates and sends an ARP packet to the target machine every second. The application will includes its own IP as the source IP in order to trick the target machine into thinking the device is the router
2	Send ARP packets to multiple Target Machine	The application creates and sends an ARP packet to the all the target machines every second. The application will includes its own IP as the source IP in order to trick the target machines into thinking the device is the router

5) DNS Spoofer

#	Test Case	Expected Results
1	Detect DNS request packets	The application captures all DNS packets and filters only DNS request packets
2	Send DNS response packets	The application parses DNS request packets and send a response packet back to the source machine

IX. SCHEDULED ESTIMATES

In this section I have included a Gantt chart describing the various activities/tasks and allocated duration for each of the tasks. This project uses an incremental software prototyping approach to development where the product is designed, implemented and tested incrementally (a little more is added each time) until the product is finished. It involves both development and maintenance. I have allotted a time frame of 154 days to the completion of this project. Each day consists of minimum of 3 hrs of works which adds up to 462 hours in total.

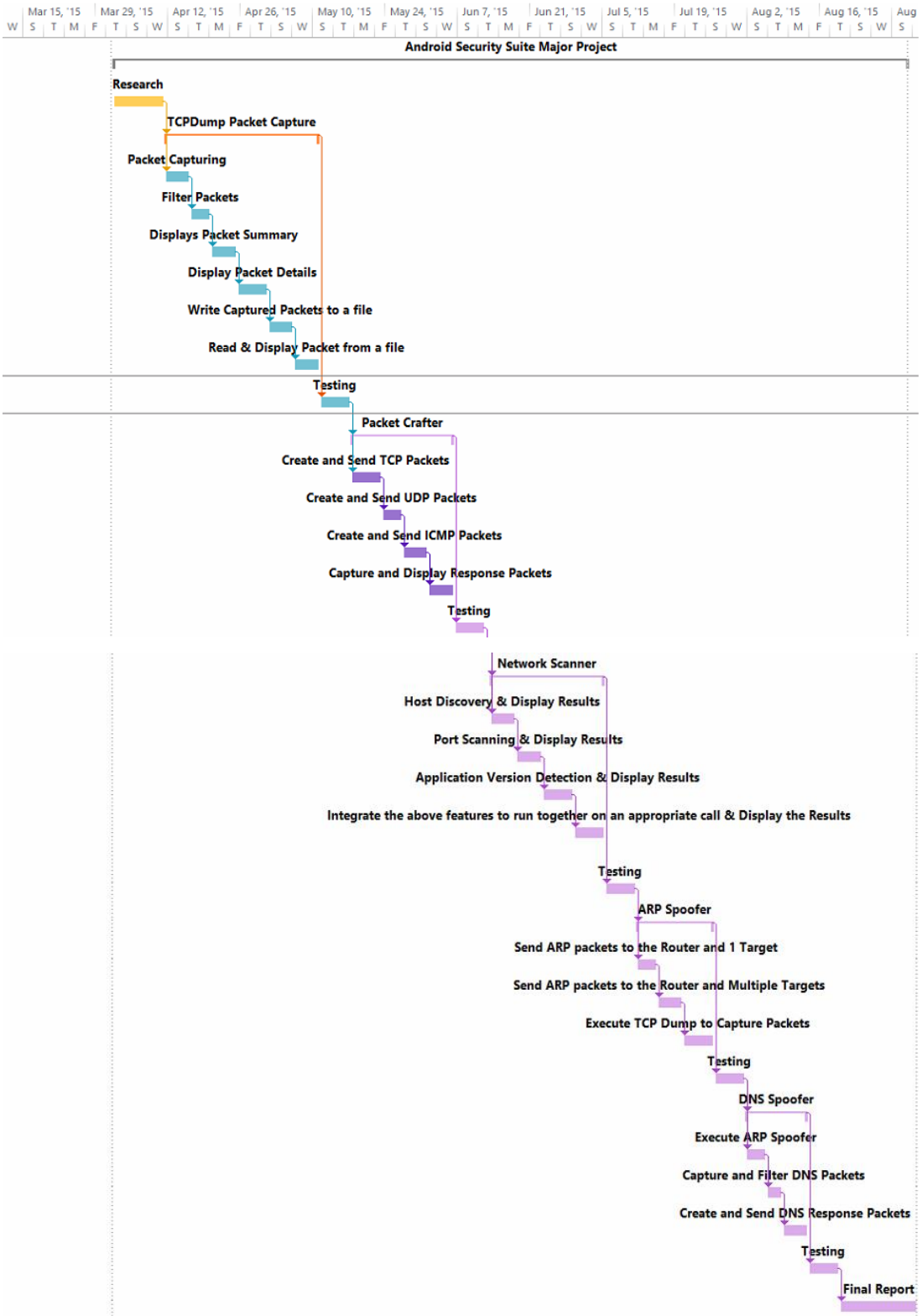
The Gantt chart below is divided based on each sub-application and various tasks for each sub-application. Upon completion of all the tasks for the sub-application, a complete Use case testing is done on the entire sub-application.

GANTT CHART

	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
0		**Android Security Suite Major Project**	**154 days**	**Wed 4/1/15**	**Tue 9/1/15**		
1		Research	10 days	Wed 4/1/15	Fri 4/10/15		
2		**TCPDump Packet Capture**	**30 days**	**Sat 4/11/15**	**Sun 5/10/15**	**1**	
3		Packet Capturing	5 days	Sat 4/11/15	Wed 4/15/15	1	
4		Filter Packets	4 days	Thu 4/16/15	Sun 4/19/15	3	
5		Displays Packet Summary	5 days	Mon 4/20/15	Fri 4/24/15	4	
6		Display Packet Details	6 days	Sat 4/25/15	Thu 4/30/15	5	
7		Write Captured Packets to a file	5 days	Fri 5/1/15	Tue 5/5/15	6	
8		Read & Display Packet from a file	5 days	Wed 5/6/15	Sun 5/10/15	7	
9		Testing	6 days	Mon 5/11/15	Sat 5/16/15	2	
10		**Packet Crafter**	**20 days**	**Sun 5/17/15**	**Fri 6/5/15**	**9**	
11		Create and Send TCP Packets	6 days	Sun 5/17/15	Fri 5/22/15	9	
12		Create and Send UDP Packets	4 days	Sat 5/23/15	Tue 5/26/15	11	
13		Create and Send ICMP Packets	5 days	Wed 5/27/15	Sun 5/31/15	12	
14		Capture and Display Response Packets	5 days	Mon 6/1/15	Fri 6/5/15	13	
15		Testing	6 days	Sat 6/6/15	Thu 6/11/15	10	

GANTT CHART

16		**Network Scanner**	**22 days**	**Fri 6/12/15**	**Fri 7/3/15**	**15**
17		Host Discovery & Display Results	5 days	Fri 6/12/15	Tue 6/16/15	15
18		Port Scanning & Display Results	5 days	Wed 6/17/15	Sun 6/21/15	17
19		Application Version Detection & Display Results	6 days	Mon 6/22/15	Sat 6/27/15	18
20		Integrate the above features to run together on an appropriate call & Display the Results	6 days	Sun 6/28/15	Fri 7/3/15	19
21		Testing	6 days	Sat 7/4/15	Thu 7/9/15	16
22		**ARP Spoofer**	**15 days**	**Fri 7/10/15**	**Fri 7/24/15**	**21**
23		Send ARP packets to the Router and 1 Target	4 days	Fri 7/10/15	Mon 7/13/15	21
24		Send ARP packets to the Router and Multiple Targets	5 days	Tue 7/14/15	Sat 7/18/15	23
25		Execute TCP Dump to Capture Packets	6 days	Sun 7/19/15	Fri 7/24/15	24
26		Testing	6 days	Sat 7/25/15	Thu 7/30/15	22
27		**DNS Spoofer**	**12 days**	**Fri 7/31/15**	**Tue 8/11/15**	**26**
28		Execute ARP Spoofer	4 days	Fri 7/31/15	Mon 8/3/15	26
29		Capture and Filter DNS Packets	3 days	Tue 8/4/15	Thu 8/6/15	28
30		Create and Send DNS Response Packets	5 days	Fri 8/7/15	Tue 8/11/15	29
31		Testing	6 days	Wed 8/12/15	Mon 8/17/15	27
32		**Final Report**	15 days	Tue 8/18/15	Tue 9/1/15	31


Mar 15, '15
Mar 29, '15
Apr 12, '15
Apr 26, '15
May 10, '15
May 24, '15
Jun 7, '15
Jun 21, '15
Jul 5, '15
Jul 19, '15
Aug 2, '15
Aug 16, '15
Android Security Suite Major Project
Research
TCPDump Packet Capture
Packet Capturing
Filter Packets
Displays Packet Summary
Display Packet Details
Write Captured Packets to a file
Read & Display Packet from a file
Testing
Packet Crafter
Create and Send TCP Packets
Create and Send UDP Packets
Create and Send ICMP Packets
Capture and Display Response Packets
Testing
Network Scanner
Host Discovery & Display Results
Port Scanning & Display Results
Application Version Detection & Display Results
Integrate the above features to run together on an appropriate call & Display the Results
Testing
ARP Spoofer
Send ARP packets to the Router and 1 Target
Send ARP packets to the Router and Multiple Targets
Execute TCP Dump to Capture Packets
Testing
DNS Spoofer
Execute ARP Spoofer
Capture and Filter DNS Packets
Create and Send DNS Response Packets
Testing
Final Report

X. DELIVERABLES

The deliverables for this project will include but are not limited to the following items:

- Application Source Code
- Application APK file for quick installation
- User Manual
- Testing Documentation
- Test Supporting documents or scripts
- Project Proposal
- Project Final Report
- Subject Expert Approval Form

3 | REFERENCES

Android Operating System. (n.d.). Retrieved March 2015, from Wikipedia: http://en.wikipedia.org/wiki/Android_%28operating_system%29

Kali Linux Tool Listing. (n.d.). Retrieved March 2015, from Kali Linux: http://tools.kali.org/tools-listing

Network security. (2014, August 22). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Network_security

Stephen Northcutt, Jerry Shenk, Dave Shackleford, Tim Rosenberg, Raul Siles, and Steve Mancini;. (2006, June). Retrieved March 2015, from SANS: http://www.sans.org/reading-room/whitepapers/analyst/penetration-testing-assessing-security-attackers-34635