Foundstone
Professional Services A DIVISION OF McAFEE

Thick Client (In)Security

Neelay S Shah

March 24, 2010

Introduction

Goal

- Educate the audience about the various strategies that can be used to test thick client applications from a security perspective

Agenda

- Motivation
- Scope
- Types of thick client applications
- Tools and techniques for security testing
- Questions

Motivation

- Tendency to trust the client
  - Development team themselves wrote the client as well
  - Performance / Speed optimization

Scope

- What do you mean by security testing?
  - Configuration management, authentication, authorization, data validation, user and session management, error handling, logging testing etc.
- For today's presentation
  - Bypass client side validation checks
    - Data validation, authorization testing etc.

Bypass client side validation checks

- Man-In-The-Middle Attack
  - Intercept the client – server communication
  - Do NOT need to understand / modify the application code
  - Typically is the fastest way of security testing the application

Bypassing client side validation

- Reverse engineer
  - Understand the client - server communication code
  - Disable the client side validation checks
  - Can be very tedious and time consuming depending on the application technology

Bypassing client side validation

- Write a new client
  - Understand the client – server communication
  - Write up a new client simulating the same control / communication flows
  - Can be very time consuming based on the scale of the application at hand
  - Typically needs knowledge of some scripting language such as Perl, Python, Tcl etc.

Scope

- For today's presentation
  - Man-In-The-Middle attacks / Intercept the client – server communication

Types of Thick Client – Server Applications

- Thick client and server using HTTP to communicate
- Thick client and server using HTTP over SSL to communicate
- Thick client and server using a proprietary TCP protocol to communicate (without any encryption)

Types of Thick Client – Server Applications

- Thick client and server using a proprietary TCP protocol over SSL to communicate

- Thick client and server using a proprietary TCP protocol and shared key / custom cryptography to communicate

Thick client – server using HTTP to communicate - Techniques

- Network Sniffing
- HTTP proxy should work
- Configuring the HTTP proxy
  - Does the application support configuring a proxy through a configuration file?
  - Does the application respect the browser proxy settings?
  - If it is a Java application, does it respect the Java proxy settings?
  - Use the “hosts” file to setup the HTTP proxy

Thick client – server using HTTP over SSL to communicate - Techniques

- Network sniffing will NOT help
- HTTP proxy should work
- Configuring the HTTP proxy
  - Does the application support configuring a proxy through a configuration file?
  - Does the application respect the browser proxy settings?
  - If it is a Java application, does it respect the Java proxy settings?
  - Use the “hosts” file to setup the HTTP proxy

Thick client – server using HTTP over SSL to communicate - Techniques

- Configuring the server's certificate
  - Install the proxy's SSL certificate in the trusted certificate authority store
    - Trusted certificate authority store can be accessed from "Start → Control Panel → Administrative Tools" or type "certmgr.msc" on the Run prompt
  - For Java applications
    - Add the proxy's certificate to the Java certificate "User" store accessible from the Java control panel applet
    - Add the proxy's certificate to the Java "System" store which is a file on the local file system using the keytool application

Thick client – server using HTTP over SSL to communicate - Techniques

- Configuring the server's certificate
  - If the client ships with the server's certificate (in the install directory or another location on the file system), replace it with the proxy's certificate
  - Generating a certificate
    - Openssl
      - openssl req -x509 -newkey rsa:1024 -keyout <private_key_file> -out <certificate_file>
    - Java keytool
    - Fiddler HTTP proxy
      - Automatically generates the certificate
      - keytool.exe -import -alias <cert_alias> -file <cert_file> -trustcacerts -storetype jks –<file_system_key_store_location>

Thick client – server using HTTP over SSL to communicate - Techniques

- Configuring the server's certificate
  - If the Java client application ships with the server's certificate as part of the (signed) JAR, then you will need to decompile, modify the JAR, recompile and resign the JAR
  - Decompile the JAR
    - Extract the JAR
    - Use a Java decompiler such as Jad to decompile the .class files
  - Modify the code to update the server's certificate
  - Recompile and Resign the JAR
    - Remove the META-INF folder
    - Create the Jar file from the modified code
      - jar.exe –cvf <Jar_Name> .

Thick client – server using HTTP over SSL to communicate - Techniques

- Configuring the server's certificate
  - Recompile and Resign the JAR
    - Create a new signing key-pair
      - keytool.exe" -genkeypair -alias <keypair_alias> -keystore <file_system_key_store_location> -storepass <store_password> -validity 500 -dname <Name_Details>
    - Sign the Jar file
      - jarsigner.exe -keystore <file_system_key_store_location> -storepass <store_password> -keypass <key_pass> <Jar_name> <keypair_alias>
    - Verify the signed Jar file
      - jarsigner.exe -verify <Jar_name>

Thick client – server using proprietary TCP protocol to communicate (without encryption)

- Network Sniffing
- HTTP proxy will NOT help
- TCP Proxy such as EchoMirage should work
  - Hooks into the Windows socket library
  - Limited ability to modify data

Thick client – server using proprietary TCP protocol to communicate over SSL

- Network sniffing will NOT help
- HTTP Proxy will NOT help
- TCP Proxy like EchoMirage should help
  - Hooks into the Windows Sockets library
  - Limited ability to modify data

Thick client – server using proprietary TCP protocol over custom / shared key cryptography to communicate

- Network sniffing will NOT help
- HTTP proxy will NOT help
- TCP proxy will NOT help
- "Detours" will help
  - Provides the ability to hook into arbitrary Win32 calls

Summary

- No one-size fits all methodology
- Need to understand the development technology and the communication protocols used by the thick client

References

- Fiddler HTTP Proxy - http://www.fiddler2.com/fiddler2/
- EchoMirage - http://www.bindshell.net/tools/echomirage
- Microsoft Detours - http://research.microsoft.com/en-us/projects/detours/
- Keytool command - http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/keytool.html
- Openssl command - http://www.openssl.org/docs/apps/req.html#EXAMPLES

Questions

Foundstone
Professional Services A DIVISION OF McAFEE

Foundstone
Professional Services A DIVISION OF McAFEE

Thick Client (In)Security

Neelay S Shah

March 24, 2010