BITS Pilani Presentation

BITS Pilani
Pilani Campus

Jagdish Prasad
WILP

innovate achieve lead

BITS Pilani
Pilani Campus

SSZG575: Ethical Hacking
Session: 05 (Mobile Application Security)

Agenda

innovate achieve lead

- Ghidra Reverse Engineering Tool
  - Introduction
  - Platforms
  - Framework components
  - Features
  - How to use?
- Mobile Application Security
  - Android Security: kernel and applications
  - IOS Security: kernel, applications
  - Rooting and Jailbreaking, File system level access, Super-user, Malware
  - Countermeasures: Strategies, Scenarios

Ghidra Tool

What is Binary Reverse Engineering?

- Reverse engineering is the process of uncovering principles behind a piece of hardware or software, such as its architecture and internal structure.
- Binary Reverse engineering is a process that hackers use to figure out a program's components and functionalities in order to find vulnerabilities in the program.
- Original software design is recovered by analyzing the code or binary of the program, in order to hack it more effectively.

Why Binary Reverse Engineering?

- Research network communication protocols
- Find algorithms used in malware such as computer viruses, trojans, ransomware, etc.
- Research the file format used to store any kind of information, for example emails databases and disk images
- Check the ability of your own software to resist reverse engineering
- Improve software compatibility with platforms and third-party software
- Find out undocumented platform features

Key Reverse Engineering Terms

- Binary Auditing
  - Binary Auditing deals with the analysis of binary files
  - developing strategies to understand, analyze and interprete native code
- De-compiler
  - A de-compiler represents executable binary files in a readable form.
  - It transforms binary code into text that software developers can read and modify.
- Disassembler
  - Carries out one-to-one mapping of processor binary instruction codes into instruction mnemonics.
- De-complilers and Disassemblers both generate human readable text from binaries however De-compilers generate much higher level text from understanding point of view.

What is GHIDRA?

- A Software Reverse Engineering (SRE) framework created by National Security Agency (NSA) of the USA.
- Includes a set of tools that help analyze compiled code on a variety of platforms including Windows, MacOS and Linux.
- Capable of disassembly, assembly and de-compilation.
- In development for 20 years.
- Primarily written in Java.
  - Some C/C++.
  - Can write scripts in Python.
  - Requires a Java Runtime and Development Kit (not all versions of Java supported).
- Designed for customizability and extensibility.

What is GHIDRA?

- Ghidra 9.0 publicly released March 2019.
- Source code released on Github April 2019.
- Created as a platform to solve hard cyber security problems.
- Install Java on Linux systems
- Add path of JDK to PATH variable
  - export PATH=<path of JDK dir>/bin:$PATH
- www.ghidra-sre.org
- https://github.com/NationalSecurityAgency/ghidra

What is GHIDRA?

GENERIC

HEXIDECIMAL

INTEGRATED

DECOMPILING

REVERSE-ENGINEERING

ARCHITECTURE

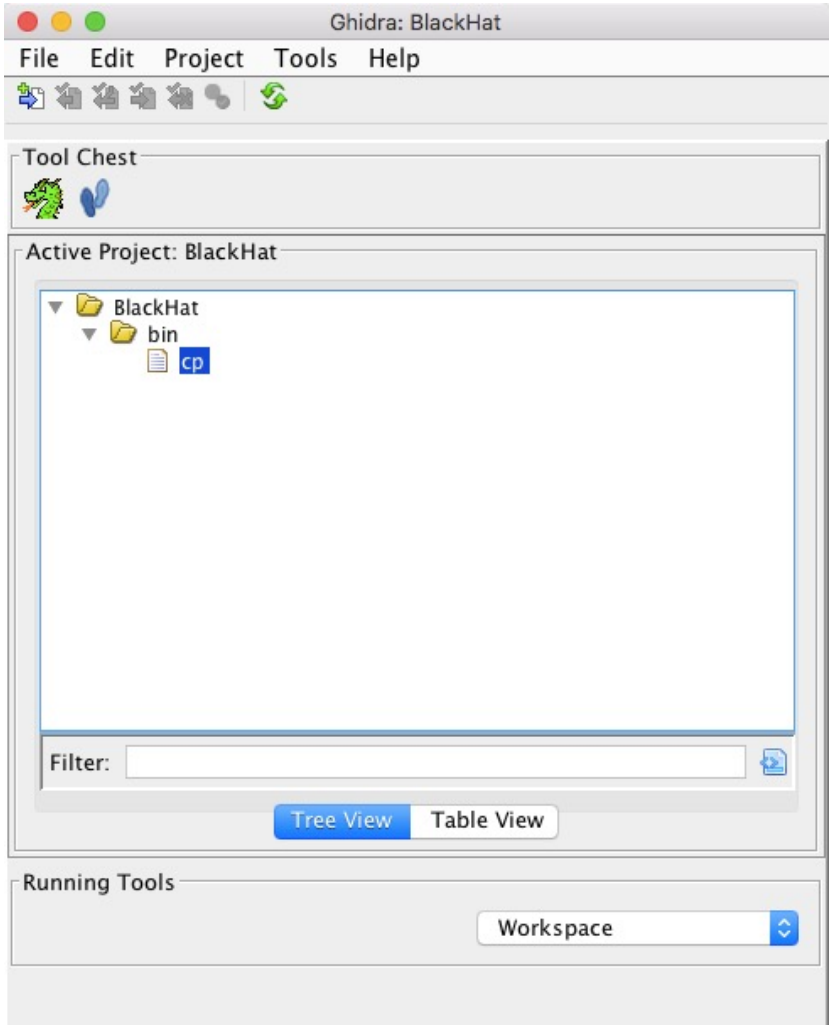
Ghidra Platform Requirements

- **Platforms Supported**
  - Microsoft Windows 7 or 10 (64-bit)
  - Linux (64-bit, CentOS 7 is preferred)
  - MacOS (OS X) 10.8.3+ (Mountain Lion or later)
  - **Note:** All 32-bit OS installations are now deprecated.
- **Minimum Requirements**
  - **Hardware**
    - 4 GB RAM
    - 1 GB storage (for installed Ghidra binaries)
    - Dual monitors strongly suggested
  - **Software**
    - Java 11 64-bit Runtime and Development Kit (JDK)

Ghidra Features

- Main UI
- Listing view
- Function graph
- Decompiler
- File system browser
- Script manager
- Integrated windows
- Other features

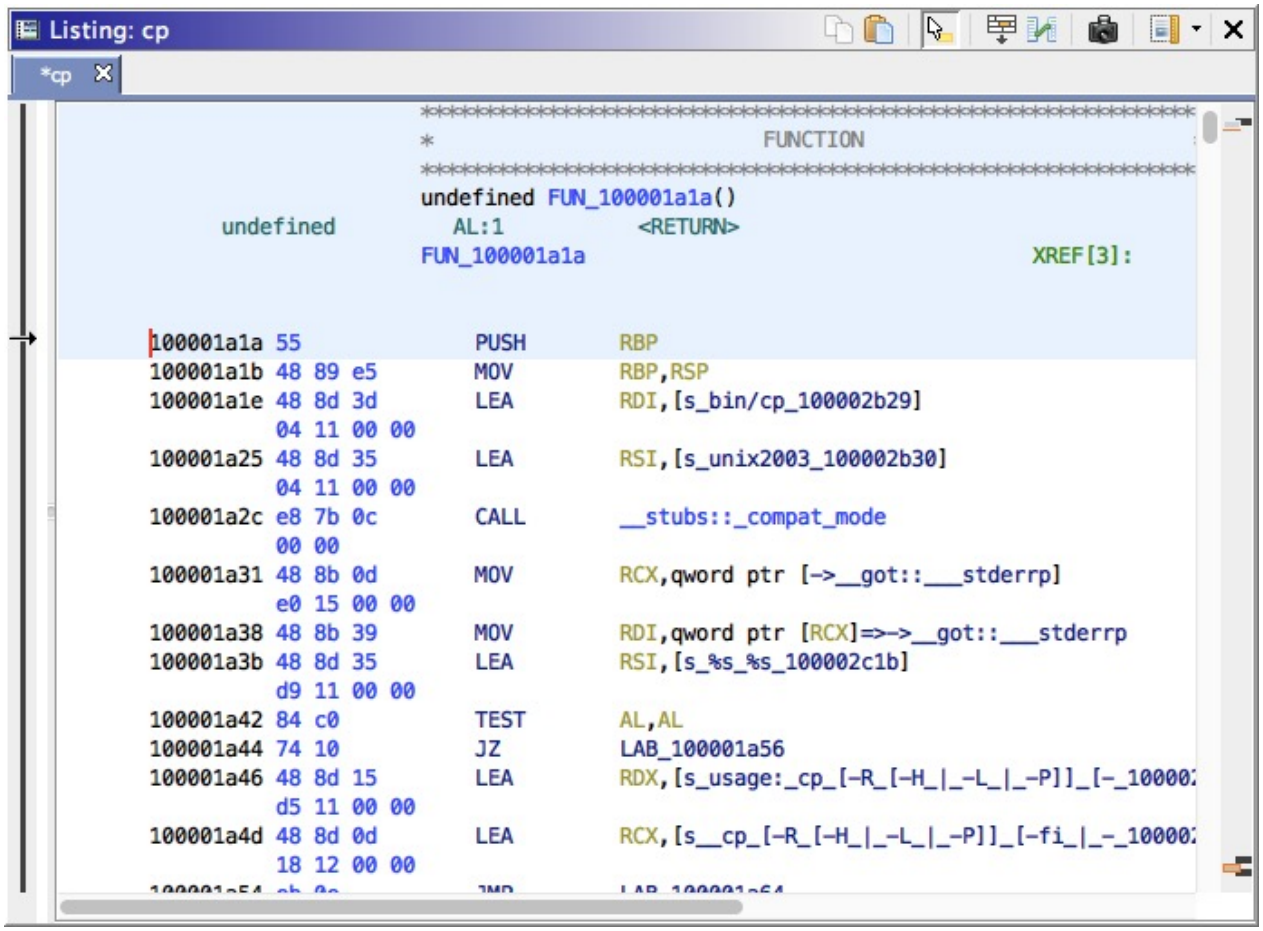
Ghidra Features: Main UI



- Ghidra is project based
- Project window

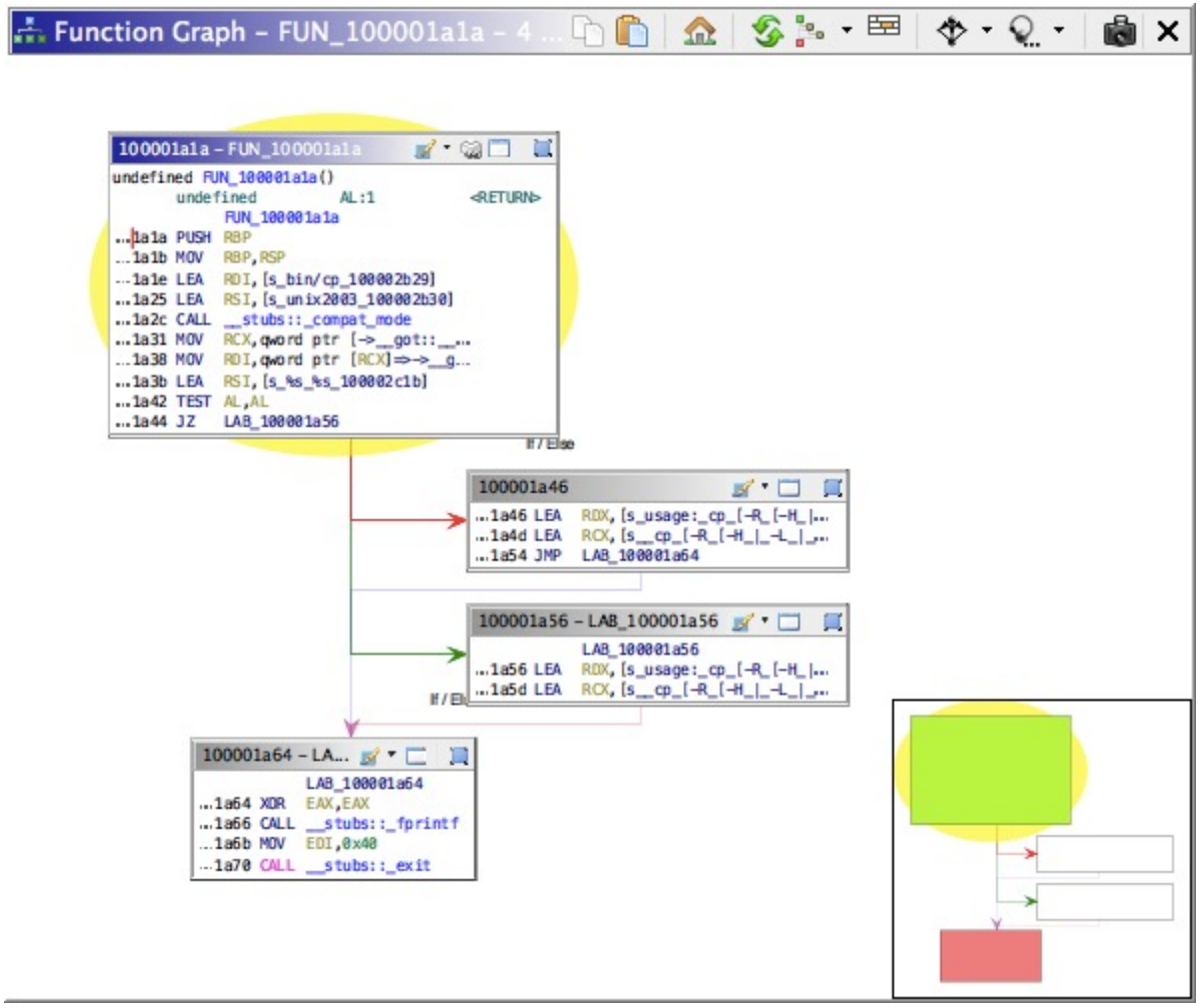
Ghidra Features: Listing View

innovate achieve lead



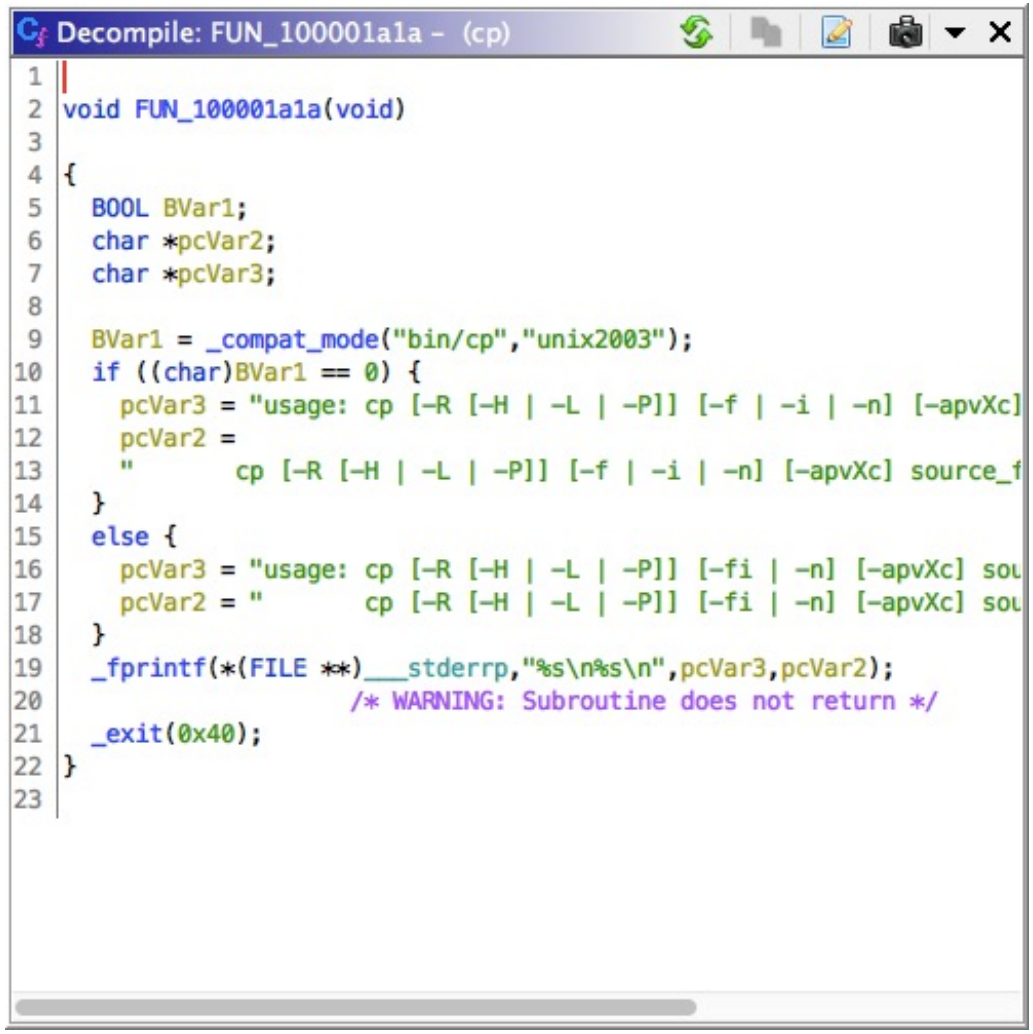
- Annotations
- Instructions
- Data
- Comments

Ghidra Features: Function Graph



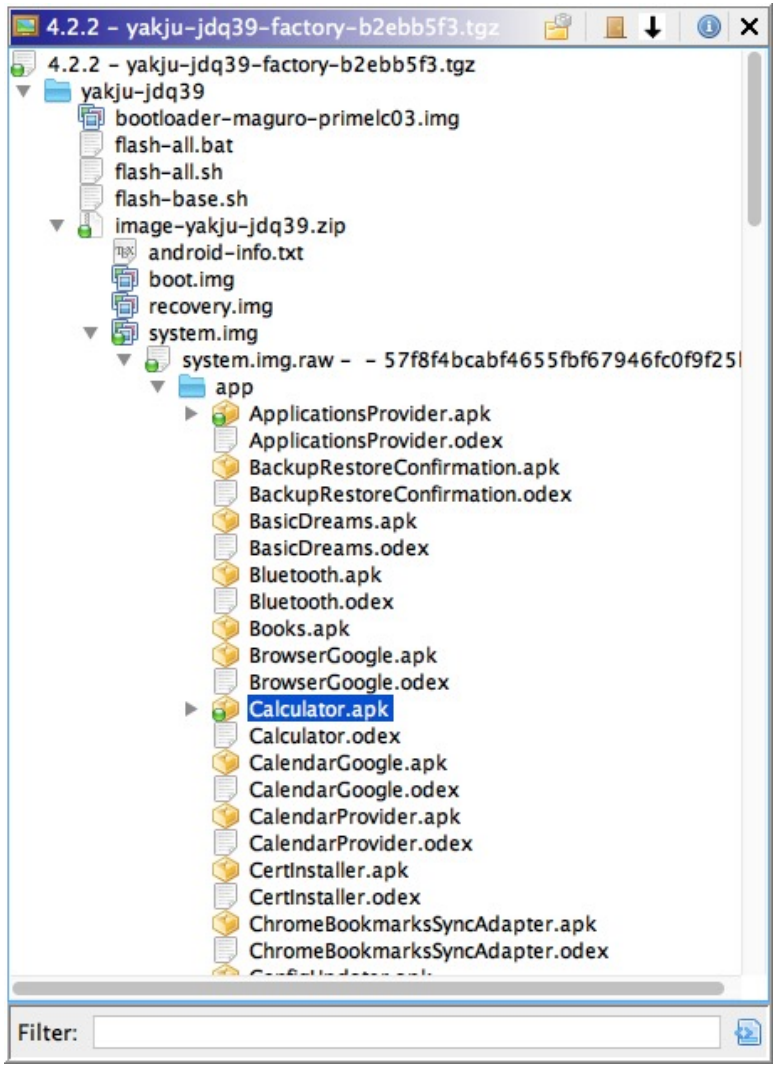
- Displays functions as code block

Ghidra Features: Decompiler



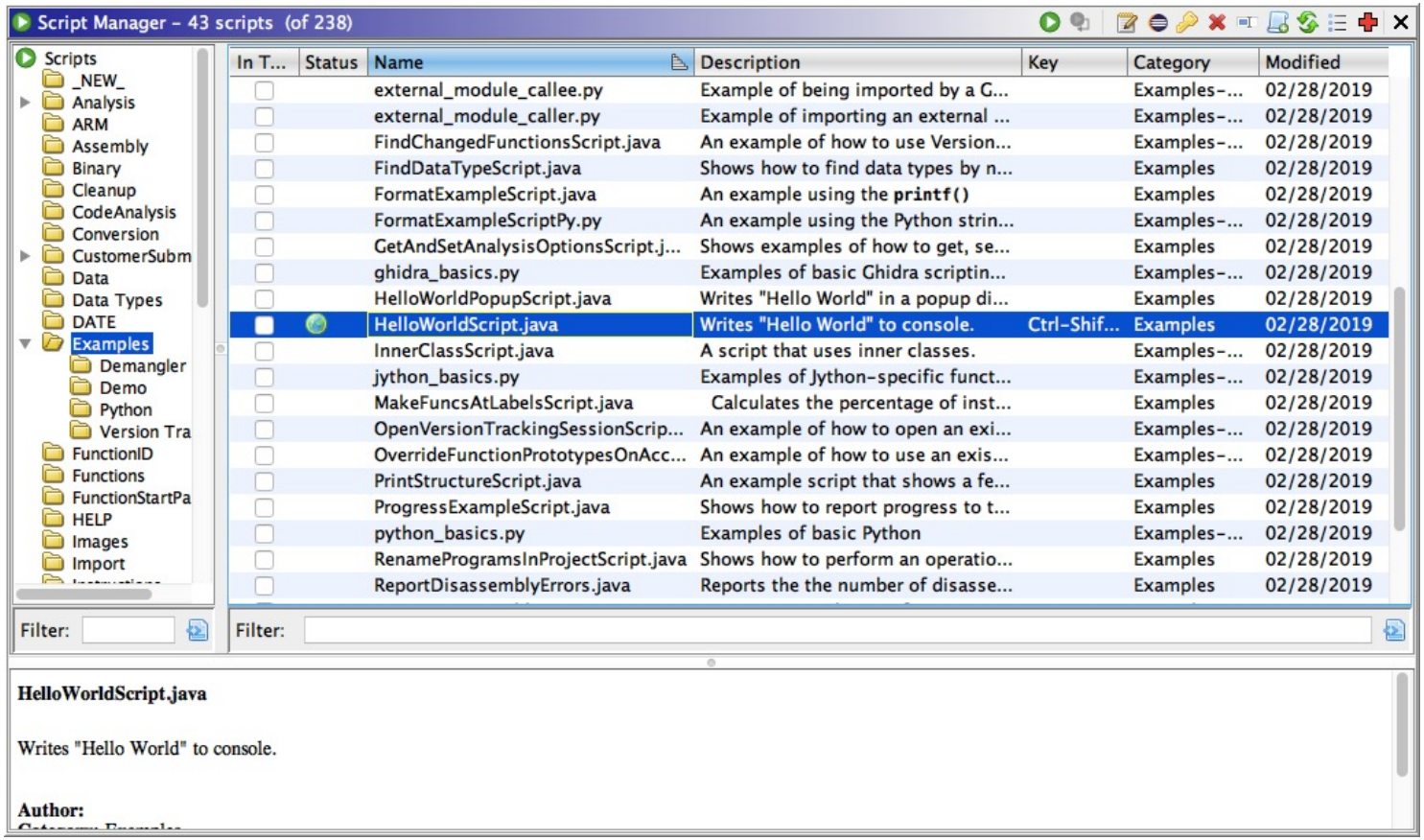
- Works for all processors with **SLEIGH** specs
  - **SLEIGH**, a machine language translation and dissassembly engine.
  - SLEIGH generates **pcode**, a reverse engineering Register Transfer Language (RTL), from binary machine instructions.
  - SLEIGH is based on **SLED**, a *Specification Language for Encoding and Decoding.*
  - SLEIGH extends SLED by providing semantic descriptions (via the RTL) of machine instructions and other enhancements for real world reverse engineering.
  - SLEIGH is part of Project **GHIDRA**.
    - Provides GHIDRA disassembler, data-flow and decompilation analysis.

Ghidra Features: File System Browser



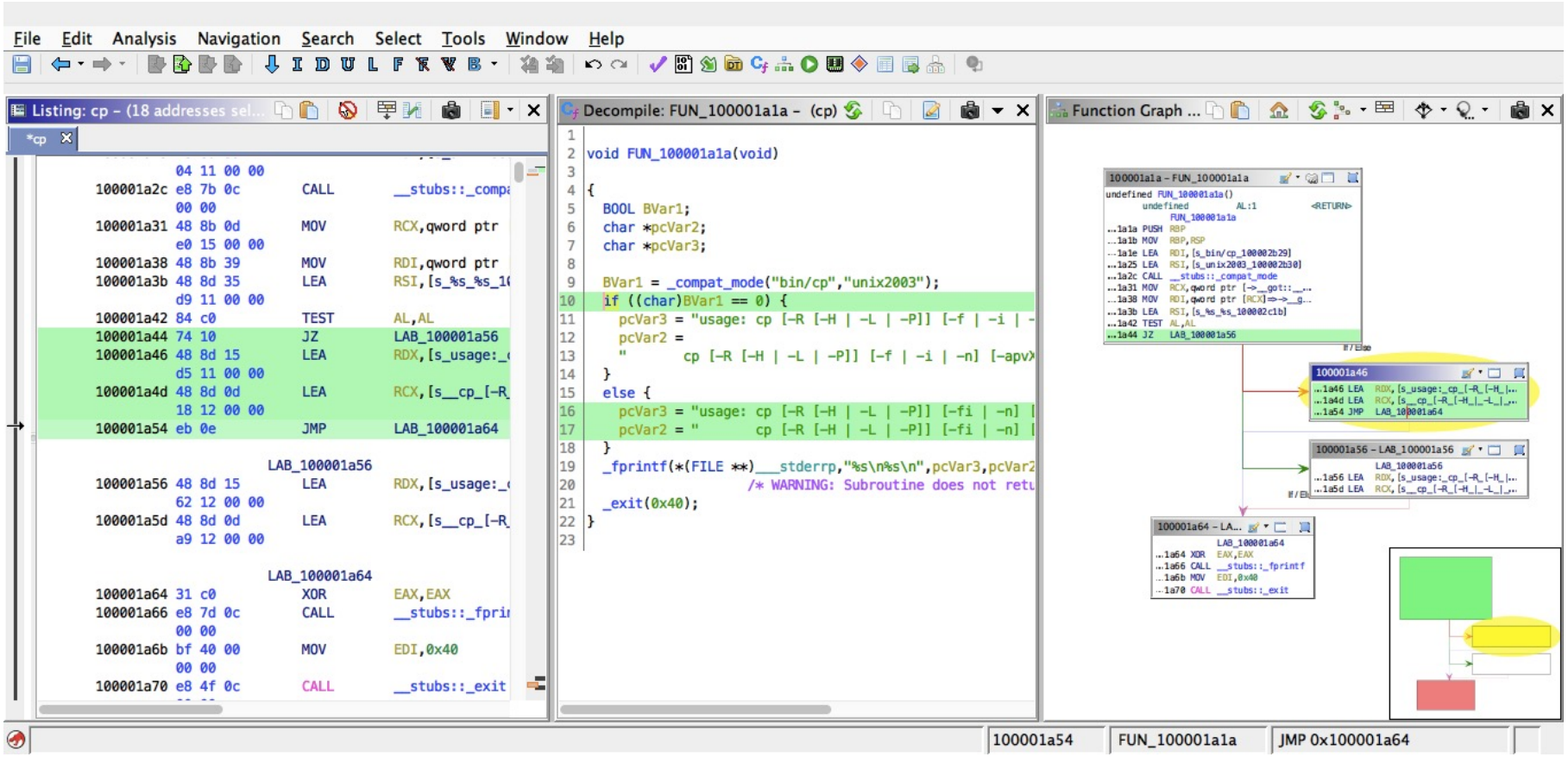
- Allows drilling down into firmware bundles
- Many formats supported like tar, zip, etc

Ghidra Features: Script Manager



- Support Java & Python

Ghidra Features: Integrated Windows



- All windows track selection and location changes

Ghidra Features: Other Features

- Assembler
- Bookmarks
- Byte Viewer (Hex Editor)
- Data Type Manager
- Entropy
- Navigation
- Searching (bytes, strings, comments)
- Version Tracking
- Version Control (Multi-user)
- And more…

Framework Components

innovate achieve lead

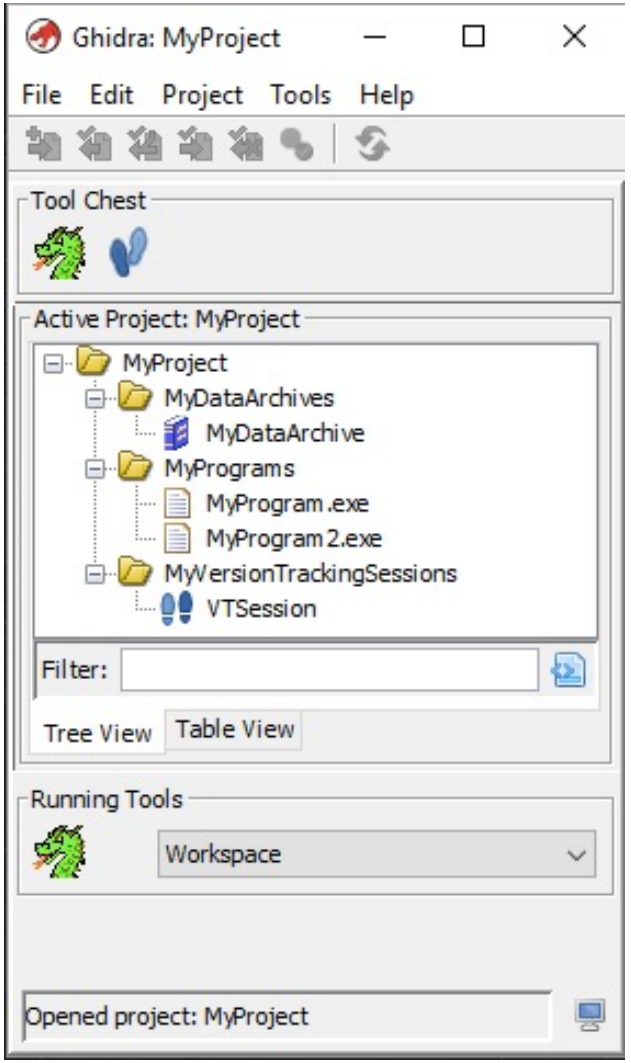
- Database
- Model
- Software Modelling
- Plugins
- Docking Windows

Ghidra: Database

- Custom database to hold Domain Object data
- Tables, Rows, Cols
- Defined in package db.*
- Created to support
  - Undo/Redo
  - Version Control (Multi-User)

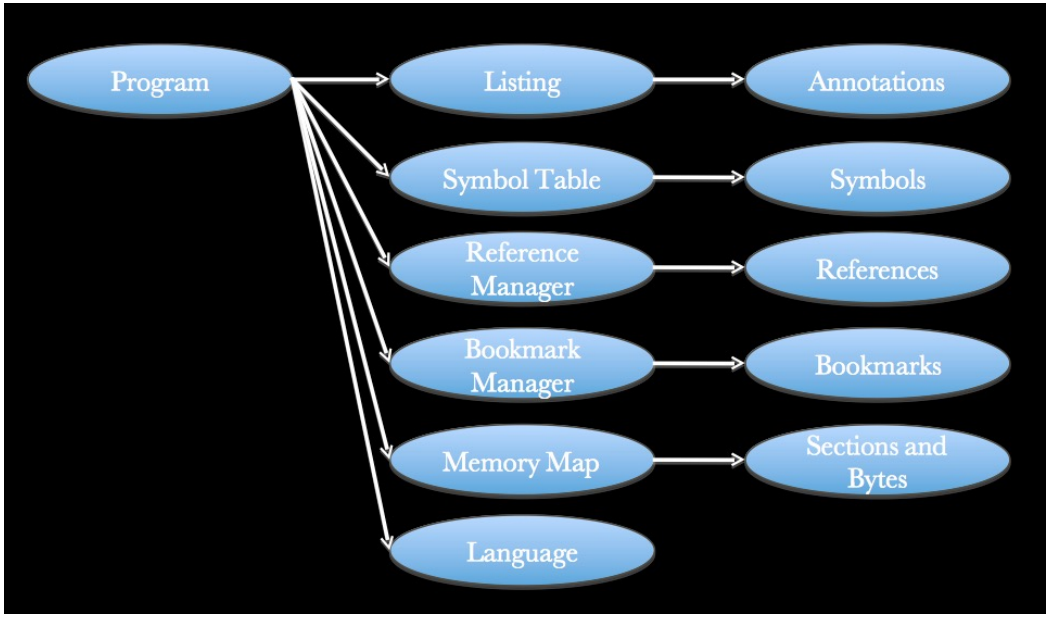
Ghidra: Model

- Data stored within the project
- Project
  - Domain Folder
    - Domain File
      - Domain Object
- Implementations supported:
  - Program
  - Data Type Archive
  - Version Tracking Session



Ghidra: Software Modelling

- Classes to represent a Program
- Program
- Memory
  - Memory Blocks, Bytes
- Symbol Table
  - Symbols
- Listing
  - Instructions, Data
- Bookmark Manager
  - Bookmarks



Ghidra: Plugins

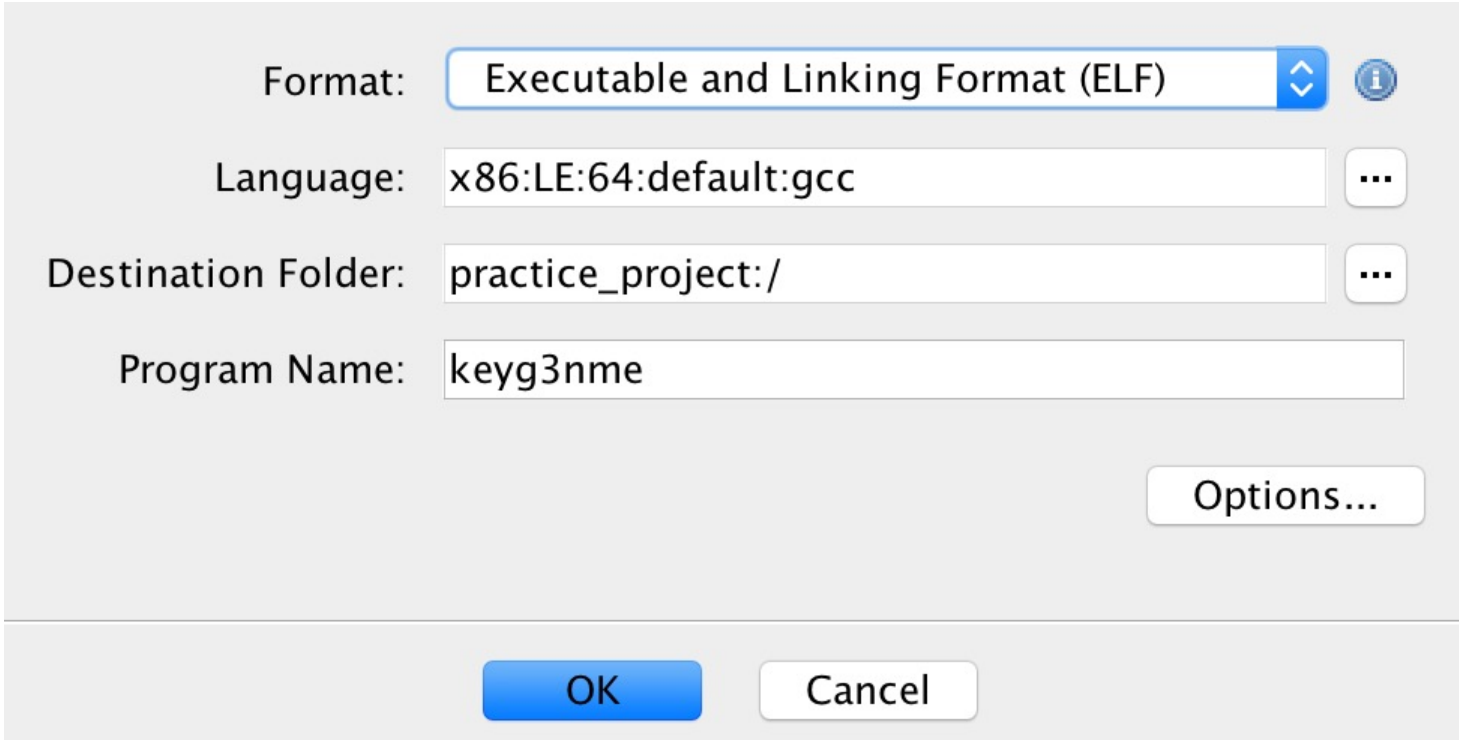
- Extends “Plugin” base class
- Produces and Consumes
  - Events (PluginEvent)
    - Selections, Locations
  - Services
    - Goto, Program Manager
- Creates
  - Menu and button actions
  - GUI components

Ghidra: Docking Windows

- Customized Java GUI components
  - GTree, GTable
- Improved TableModel and TreeModel
  - Generic Threaded Loading, Sorting, Filtering
- Uniform Look-and-Feel
- Overcome Java Swing limitations, issues, and general wonkiness

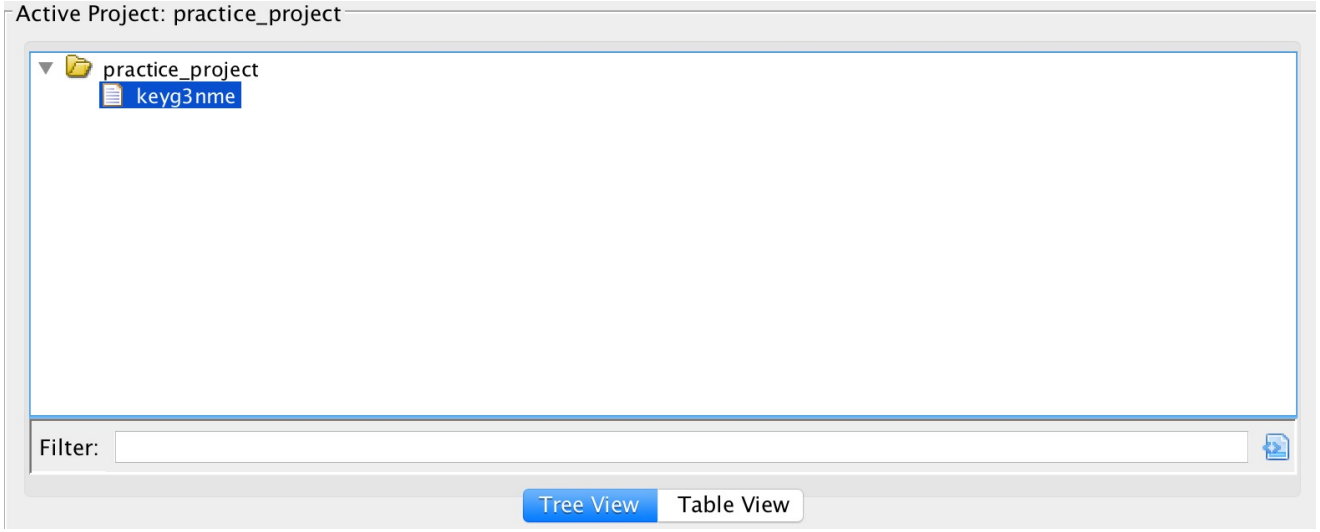
Open a Binary

- Create a new project by going to File > New project
- Go to File > Import file to import the binary file that you want to analyze



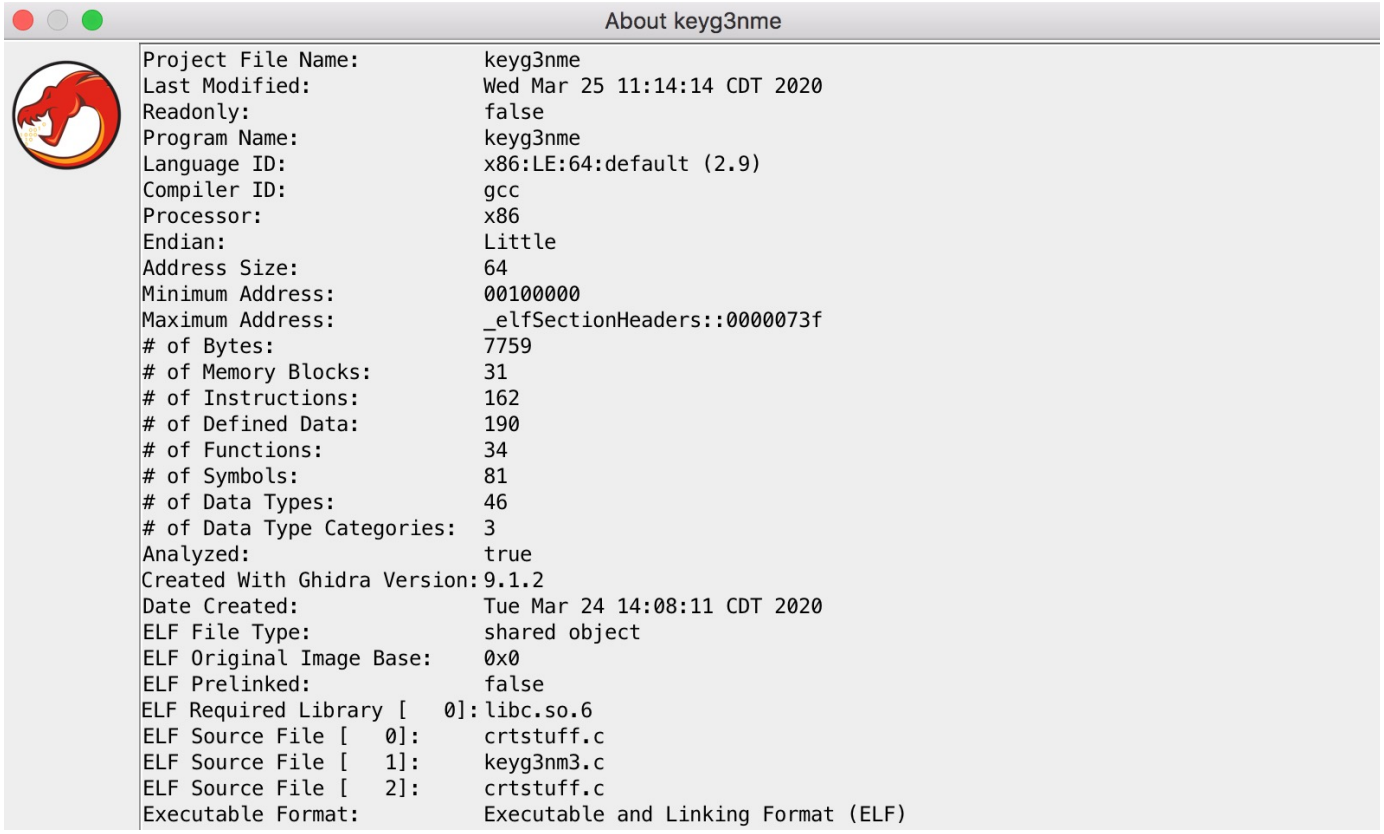
Open a Binary

- After importing the file, you will see a window like this one:



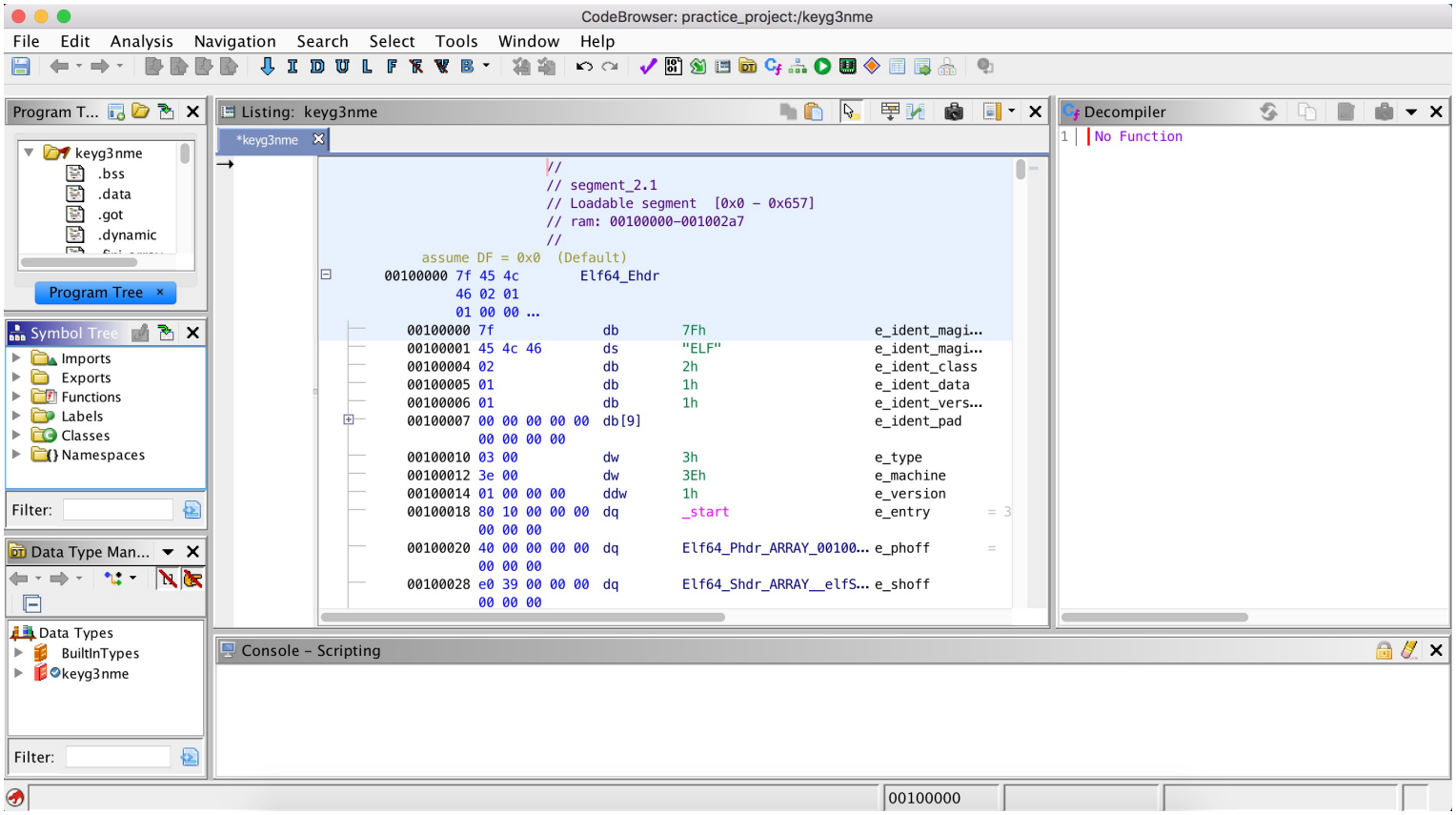
Open a Binary

- Basic info about the program being analyzed is displayed

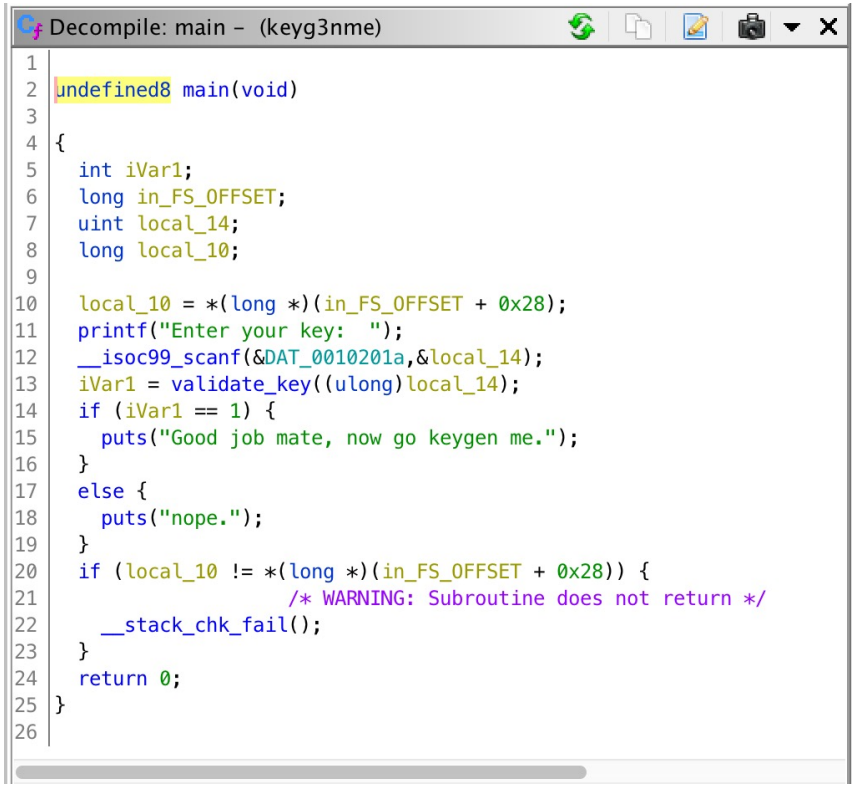


Features of Ghidra

- Double click on the file to analyze
- A new window opens with binary & hex code snippets:

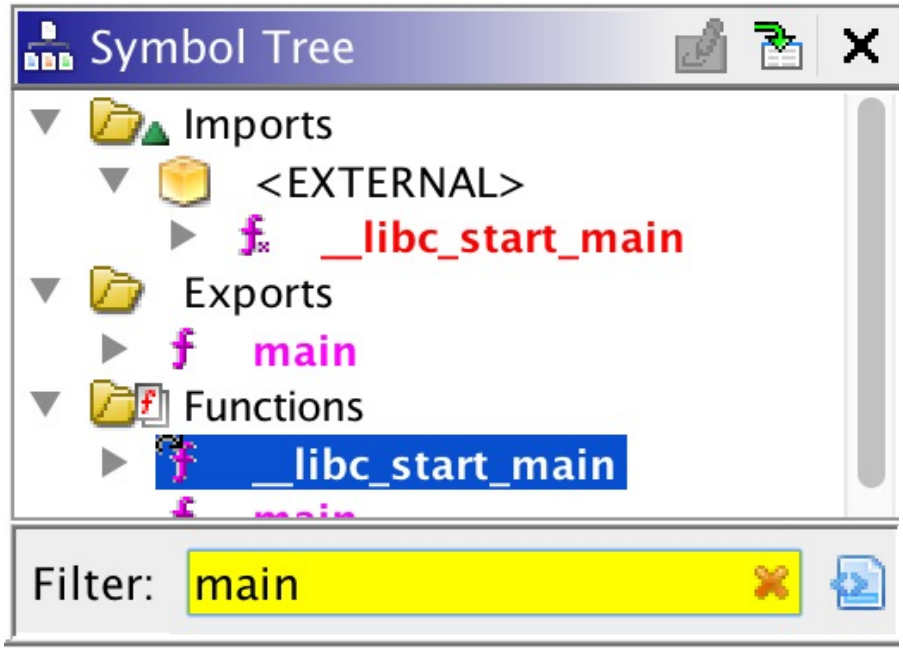


Features of Ghidra



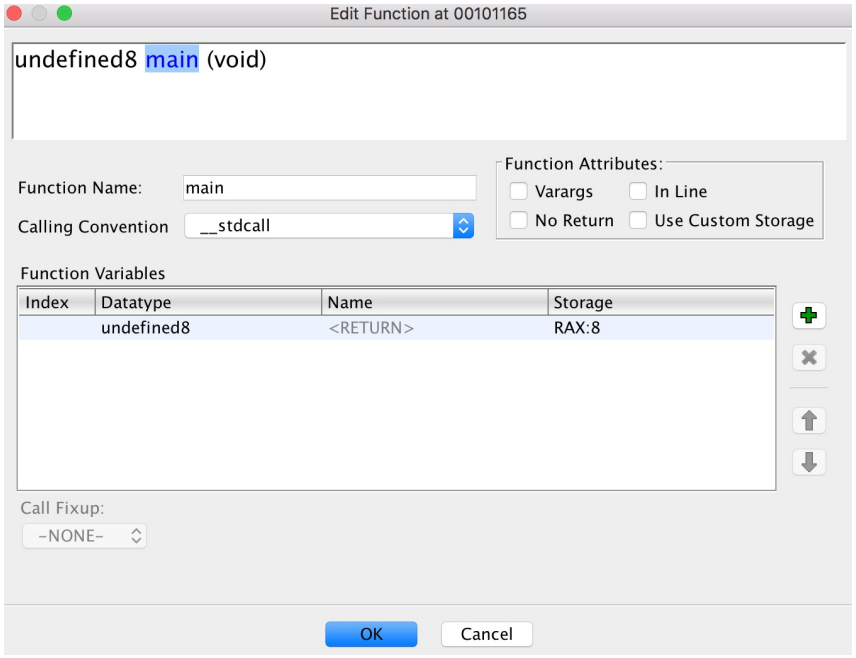
- List of symbols in the Symbol Tree view on the left
- Symbols are references to some type of data like an import, a global variable, or a function.
- Listing view in the middle shows typical assembly code fields like addresses, bytes, operands, and comments, etc.
- Decompiler on the right converts assembly back to C code.
  - double click on the function that you want to analyze in the symbol tree view.

Finding a Function



- Symbol Tree helps navigate around the binary and search for individual functions.
- Example: How do we find the “main” function of a program?
  - Try to search for the function in the symbol tree view

Edit Program During Analysis



- Can edit the program during analysis in Ghidra.
- Can edit a function by right-clicking on the function name in either the symbol tree, the listing window or the decompiler window, then going to the “Edit Function” option.
- Retype and rename variables by right-clicking on the variable names and then going to the “Retype Variable” or “Rename Variable” option.

Ghidra Demo Video

1. Ghidra Installation and Getting Started Video:

   https://ghidra-sre.org/GhidraGettingStartedVideo/GhidraGettingStartedVideo.mp4

2. Ghidra Presentations

   https://github.com/NationalSecurityAgency/ghidra/wiki/files/recon2019.pdf

   https://github.com/NationalSecurityAgency/ghidra/wiki/files/blackhat2019.pdf

Mobile Application Security

Mobile Operating Systems

innovate achieve lead

- **Android:** Operating system from Google licensed to various mobile device manufactures
  - Windows equivalent for mobile devices

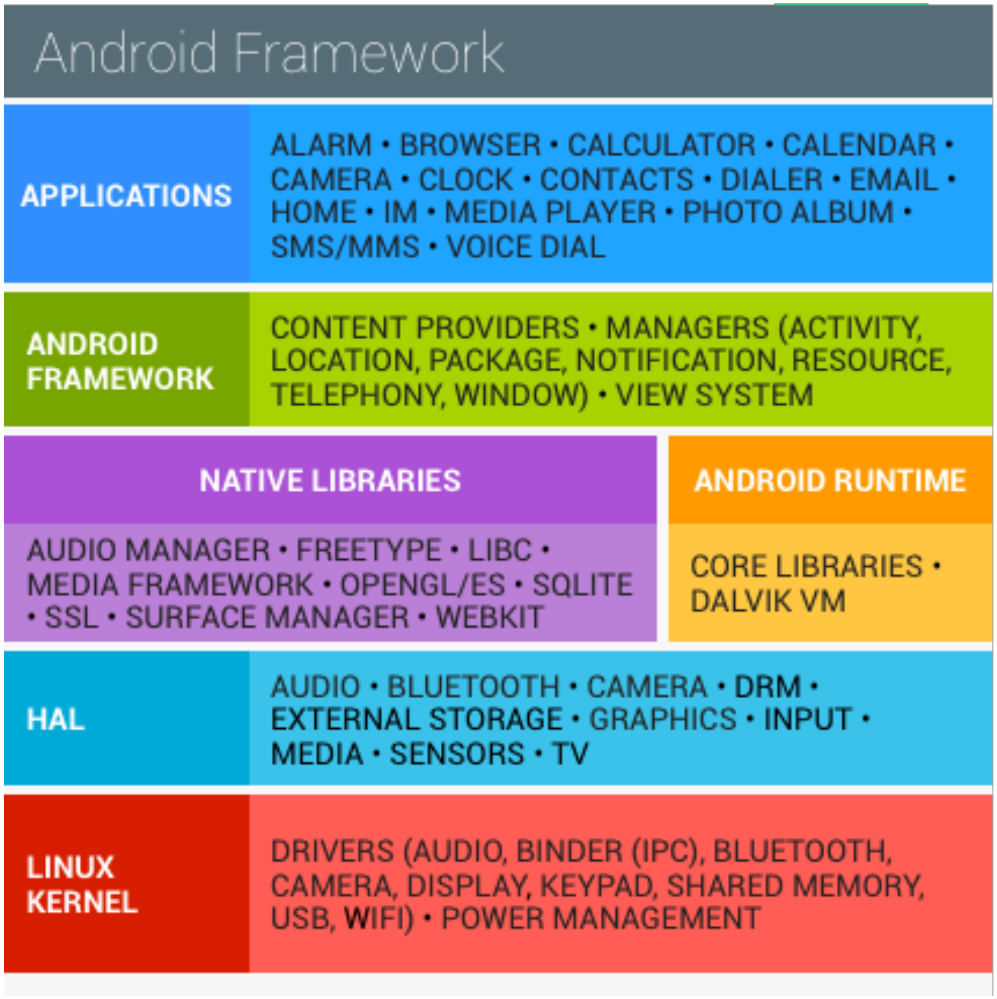
- **iOS:** Operating system for Apple iPhone and other Apple mobile devices.

Jailbreaking and Rooting

- Jailbreaking is the process of removing the limitations imposed by Apple on devices running the iOS operating system.
- Jailbreak allows the phone's owner to gain full access to the root of the iOS and access all the features.
- Rooting is the process of removing the limitations on a mobile or tablet running the Android operating system.
- Jailbreaking/Rooting open security holes that may not been readily apparent or undermine the device's built-in security measures.
- Jailbroken and Rooted phones are much more susceptible to
  - Viruses and malware
  - Bypass app vetting processes imposed by Google and Apple
  - Can lead to malicious or virus infected app downloads

Android Application Security

Android Fundamentals



- Device hardware
- HAL (Hardware Abstraction Layer)
- Android OS – Linux with device drivers
- Android App Runtime
  - Mostly in Java
  - Also uses native libraries
- Pre-installed apps: email, phone, calendar, contacts, web browser etc
- User installed apps

Google Security Services

innovate achieve lead

Service Name	Service Description
Google Play	Collection of services that allow users to discover, install, and purchase apps from their Android device or the web
Android Updates	Delivers new capabilities and security updates to selected Android devices
App Services	Frameworks that allow Android apps to use cloud capabilities such as data backup
Verify Apps	Warn or automatically block the installation of harmful apps, continually scan apps on the device, warn about or remove harmful apps
SafetyNet	A privacy preserving intrusion detection system to assist Google tracking, mitigate known security threats, and identify new security threats
SafetyNet Attestation	Third-party API to determine whether the device is CTS compatible
Android Device Manager	To locate a lost or stolen device

Android Security

- Dalvik Virtual Machine (VM), a software component that runs each application in its own instance of the Dalvik VM
- Once an application is developed in Java, it is transformed to dex (Dalvik Executable) files
  - Android SDK ‘dx’ toolset converts into dex files compatible with the Dalvik VM
- Android security is SQLite (a SQL database engine) based
- Applications store persistent data in the device in SQLite databases without proper security measures (No encryption) to protect its confidentiality
- Once an Android device has been compromised, it is possible to access confidential information stored in those databases

Kernel Security

- Linux kernel is the base for a Android computing environment.
- Linux kernel provides Android with several key security features including:
  - A user-based permissions model
  - Process isolation
  - Extensible mechanism for secure IPC
  - Ability to remove unnecessary and potentially insecure parts of the kernel
- Application Sandbox
  - Android's application security is enforced by the application sandbox
  - Isolates apps from each other
  - Protects apps and the system from malicious apps.

Kernel Security

- System Partition and Safe Mode
  - Contains Android's kernel and operating system libraries like application runtime, application framework, and applications.
  - This partition is set to read-only.
  - In Safe Mode booting of device third-party applications are not launched automatically however device owner can launch these manually.
- Filesystem Permissions
  - Follows UNIX-style filesystem permissions to ensure that one user cannot alter or read another user's files.
  - Each application runs as its own user.
  - Unless the developer explicitly shares files with other applications, files of one application cannot be read or altered by another application.
- Security-Enhanced Linux
  - Uses Security-Enhanced Linux (SELinux) to apply access control policies and establish mandatory access control (mac) on processes.

Kernel Security

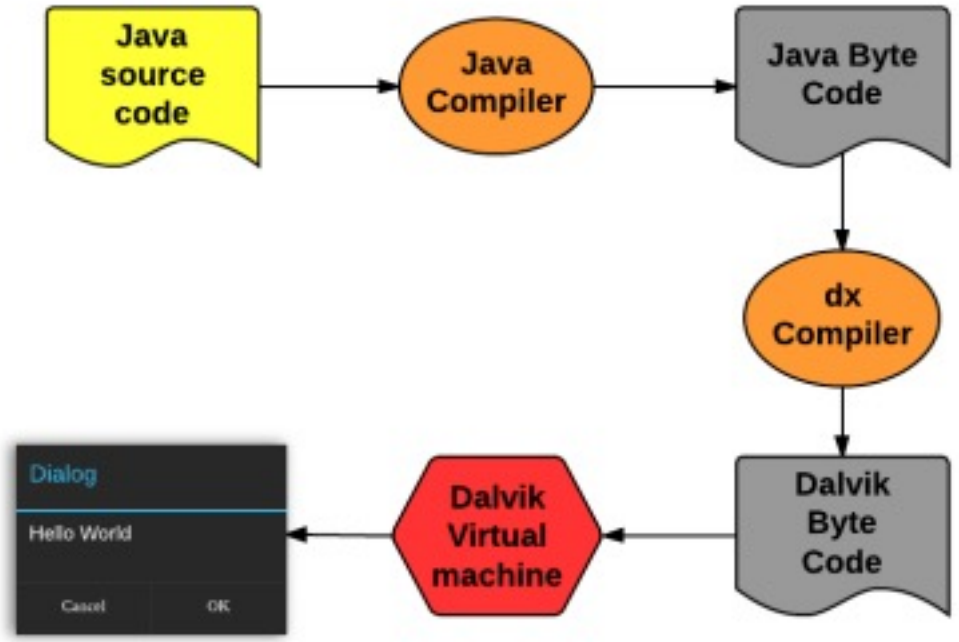
- Verified boot
  - Android 6.0 and later supports verified boot and device-mapper-verity.
  - Verified boot guarantees the integrity of the device software starting from a hardware root of trust up to the system partition.
  - During boot, each stage cryptographically verifies the integrity and authenticity of the next stage before executing it.
  - Android 7.0 and later supports strictly enforced verified boot, which means compromised devices cannot boot.
- Cryptography
  - Android provides a set of cryptographic APIs for use by applications.
  - APIs include implementations of standard and commonly used cryptographic algorithms such as AES, RSA, DSA, and SHA.
  - Specific APIs are provided for higher level protocols like SSL and HTTPS.
  - Android 4.0 provides the KeyChain class to allow applications to use the system credential storage for private keys and certificate chains.

User Security Features

- Filesystem Encryption
  - Android 3.0 and later provides full filesystem encryption at kernel level.
  - Android 5.0 and later supports full-disk encryption.
    - Full-disk encryption uses a single key—protected with the user's device password—to protect the whole of a device's user data partition.
  - Android 7.0 and later supports file-based encryption.
    - File-based encryption allows different files to be encrypted with different keys.
- Password Protection
  - Android can be configured to verify a user-supplied password prior to providing access to a device.
  - Use of a password and/or password complexity rules can be required by a device administrator.
- Device Administration
  - Android 2.2 and later provide the Android Device Administration API
  - Administrators can also remotely wipe lost or stolen handsets.
  - APIs are available to third-party providers of Device Management solutions.

Elements of App

- AndroidManifest.xml
- Activities
- Services
- Broadcast Receiver
- Protected APIs:
  - Camera functions
  - Location data (GPS)
  - Bluetooth functions
  - Telephony functions
  - SMS/MMS functions
  - Network/data connections



App Structure

- Primarily written in Java, Kotlin (transpiled to Java), and C++.
- Distributed in Android Package (.apk) format which is similar to a ZIP file containing all the assets and bytecode for an app.
- A typical unzipped APK structure looks like this:
  - **AndroidManifest.xml:** Basic application details like name, version, external accessible activities & services, minimum device version etc
  - **META-INF:** Metadata information, developer certificate, checklists etc
  - **classes.dex:** Compiled byte code of application
  - **resources.arsc:** Metadata about resources and XML
  - **res/:** Compressed binary XMLs
  - **lib/:** External C/C++ libraries if any

Hacking an App

- The way that most Android malware works is to:
  - take a legitimate application
  - disassemble the dex code, and decode the manifest
  - include the malicious code
  - assemble the dex, encode the manifest, and sign the final apk file.

Hacking an App

- One popular tool to do this is apktool
  - Install apktool (code.google.com/p/android- apktool/downloads/list)
  - Download the apk that is going to be modified (ex: old version of Netflix)
  - Disassemble the apk (apktool d Netflix.apk out)
  - Perform the modifications in the .smali files and in the manifest located in the folder generated with the same name as the disassembled application
  - Execute the **build** command to rebuild the package again (apktool b)
    - Repacked apk is stored in the out/dist folder.
    - Before signing the apk, generate a private key with a corresponding digital certificate.
  - Download the SignApk.jar tool.
    - Unzip it in the dist folder
    - Execute following command: java –jar signapk.jar certificate.pem key.pk8 Netflix.apk Netflix_signed.apk
    - Verify the process by: jarsigner –verify –verbose –certs Netflix_signed.apk

App Analysis: Static

- apktool:
  - Extracts APK and resources from app binary
  - Also extracts smali (human readable java byte code) code from dex file

```
import java.lang.System;

public class HelloWorld {
    public static void
main(String[] args) {
        System.out.println("Hello
World!");
    }
}
```

Java

smali

```
.class public LHelloWorld;
.super Ljava/lang/Object;
.source "HelloWorld.java"


# direct methods
.method public constructor <init>()V
    .registers 1

    .prologue
    .line 3
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V

    return-void
.end method

.method public static main([Ljava/lang/String;)V
    .registers 3
    .param p0, "args"    # [Ljava/lang/String;

    .prologue
    .line 5
    get-object v0, Ljava/lang/System;->out:Ljava/io/PrintStream;

    const-string v1, "Hello World!"

    invoke-virtual {v0, v1}, Ljava/io/PrintStream;->println(Ljava/lang/String;)V

    .line 6
    return-void
.end method
```

App Analysis: Static

- dex2jar or jadx
  - decompile from dex to jar format
  - APKs are minified for easier distribution and obfuscation reasons.
  - jadx provides features that make it easier to work with deobfuscated jars and lets enforce minimum field name lengths during decompilation.
  - modified names are based around the original names
  - Other decompilers are procyon, Fernflower, CFR
- Decompilation process:
  - Use apktool to extract APK and decompress resource files
  - Use jadx to decompile the APK to .java source files
  - Open the decompiled source code folder in Visual Studio Code for easy search and navigation
- Use aapt to extract

App Analysis: Static

- aapt: Part of Android SDK, can dump AndroidManifest.xml tree from an APK without needing to decompile or extract anything

```
$ aapt dump xmltree com.myapp-1.0.0.apk AndroidManifest.xml
N: android=http://schemas.android.com/apk/res/android
  E: manifest (line=2)
    A: android:versionCode(0x0101021b)=(type 0x10)0x409
    A: android:versionName(0x0101021c)="1.0.0" (Raw: "1.0.0")
    A: android:installLocation(0x010102b7)=(type 0x10)0x0
    A: package="com.myapp" (Raw: "com.myapp")
    E: uses-sdk (line=8)
      A: android:minSdkVersion(0x0101020c)=(type 0x10)0x15
      A: android:targetSdkVersion(0x01010270)=(type 0x10)0x1b
    E: uses-feature (line=12)
```

```
$ aapt dump badging com.myapp-1.0.0.apk
package: name='com.myapp' versionCode='123' versionName='1.0.0'
platformBuildVersionName=''
install-location:'auto'
sdkVersion:'21'
targetSdkVersion:'27'
uses-permission: name='com.venmo.permission.C2D_MESSAGE'
uses-permission: name='com.google.android.c2dm.permission.RECEIVE'
uses-permission: name='android.permission.INTERNET'
uses-permission: name='android.permission.WRITE_EXTERNAL_STORAGE'
uses-permission: name='android.permission.READ_EXTERNAL_STORAGE'
uses-permission: name='android.permission.READ_CONTACTS'
...
```

App Analysis: Passive

- Involves proxying the device, bypassing SSL pinning, and observing device logs
- **Logcat**
  - Built in tool in the Android SDK to monitor device logs.
  - Often apps print out useful debug information i.e. secret keys, user information etc into logcat.
  - This information can be very useful to understand working of an application.
  - To use logcat, simply run “adb logcat” with a device connected, and you should see system logs.
  - Ref logcat: https://developer.android.com/studio/command-line/logcat

App Analysis: Passive

- **Drozer**
  - Toolkit designed to help analyze Android applications
  - Provides a lot of useful information such as checking for bad permissions, monitoring IPC calls, and more.
- **SSL Pinning**
  - SSL Certificate pinning is where an app has a known list of valid SSL certificates for a domain (or a set of domains).
  - While making HTTPS connections from the device, it ensures that the certificates from the server match what they are set to in the application.
  - If the cert from the server doesn't match the list of pre-approved certificates, the device drops the connection and throws an SSL error.
- To bypass SSL pinning, one can use tools (a catch-all Frida script, something pre-built like JustTrustMe for Xposed) or a custom solution.

App Analysis: Dynamic

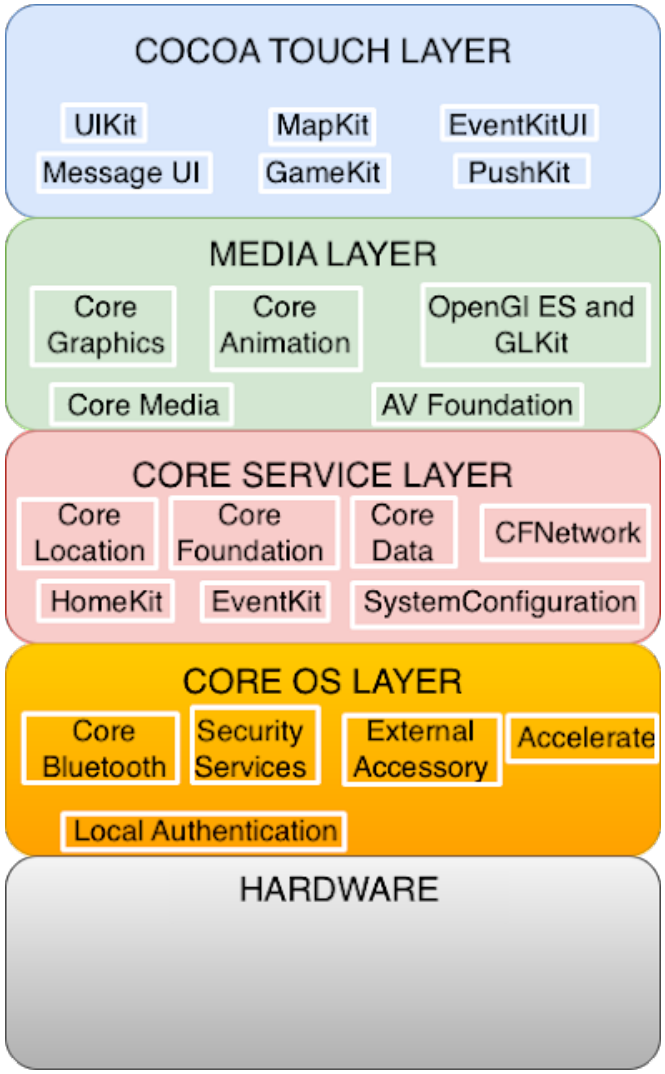
- Way of interacting with and figuring out security vulnerabilities within applications by writing dynamic hooks to talk with them.
- Frida tool can modify, hook and dynamically interact with applications
- **Frida**
  - Some useful resources for writing Frida scripts are:
  - https://frida.re/docs/android/
  - https://www.frida.re/docs/javascript-api/

iOS Application Security

iOS Platform

- iOS development begin during mid 80s at NeXT Inc, who developed high end workstations.
- NeXT's operating system NeXTSTEP was based on Carnegie Mellon University's Mach kernel and BSD Unix.
- In 1996 Apple purchased NeXT and NeXTSTEP was chosen to replace ageing Mac OS (Classic).
- In a pre-release version (Rhapsody), NeXTSTEP was modified to adopt Mac styling – pre-cursor for UI of Mac OSX.
- Released as Mac OSX in Mar 2001.
- In 2007 iPhone iOS released
  - Derived from NeXTSTEP/Mac OS X family
  - Kernel is Mach/BSD based
  - Uses Objective-C and class libraries of Apple

iOS Architecture



1. **Core OS Layer:**
   - Core OS layer holds the low level features that most other technologies are built upon.
     - Core Bluetooth Framework.
     - Accelerate Framework.
     - External Accessory Framework.
     - Security Services framework.
     - Local Authentication framework.

iOS Architecture: Core Services Layer

- Important Frameworks available in the core services layers are:
  - **Address book framework** – Gives programmatic access to a contacts database of user.
  - **Cloud Kit framework** – Gives a medium for moving data between your app and iCloud.
  - **Core data Framework –** Technology for managing the data model of a Model View Controller app.
  - **Core Foundation framework –** Interfaces that gives fundamental data management and service features for iOS apps.
  - **Core Location framework –** Gives location and heading information to apps.
  - **Core Motion Framework –** Access all motion based data available on a device. Using this core motion framework Accelerometer based information can be accessed.
  - **Foundation Framework –** Objective C covering too many of the features found in the Core Foundation framework
  - **Healthkit framework –** New framework for handling health-related information of user
  - **Homekit framework –** New framework for talking with and controlling connected devices in a user's home.
  - **Social framework –** Simple interface for accessing the user's social media accounts.
  - **StoreKit framework –** Gives support for the buying of content and services from inside your iOS apps, a feature known asIn-App Purchase.

iOS Architecture: Media Layer

- Graphics, Audio and Video technology is enabled using the Media Layer.
- **Graphics Framework**:
  - **UIKit Graphics –** It describes high level support for designing images and also used for animating the content of your views.
  - **Core Graphics framework –** It is the native drawing engine for iOS apps and gives support for custom 2D vector and image based rendering.
  - **Core Animation –** It is an initial technology that optimizes the animation experience of your apps.
  - **Core Images –** gives advanced support for controling video and motionless images in a nondestructive way
  - **OpenGl ES and GLKit –** manages advanced 2D and 3D rendering by hardware accelerated interfaces
  - **Metal –** It permits very high performance for your sophisticated graphics rendering and computation works. It offers very low overhead access to the A7 GPU.

iOS Architecture: Media Layer

- Graphics, Audio and Video technology is enabled using the Media Layer.
- **Audio Framework**
  - **Media Player Framework –** It is a high level framework which gives simple use to a user's iTunes library and support for playing playlists.
  - **AV Foundation –** It is an Objective C interface for handling the recording and playback of audio and video.
  - **OpenAL –** is an industry standard technology for providing audio.
- **Video Framework**
  - **AV Kit –** framework gives a collection of easy to use interfaces for presenting video.
  - **AV Foundation –** gives advanced video playback and recording capability.
  - **Core Media –** framework describes the low level interfaces and data types for operating media.

iOS Architecture: Cocoa Touch Layer

- **EventKit framework –** gives view controllers for showing the standard system interfaces for seeing and altering calendar related events
- **GameKit Framework –** implements support for Game Center which allows users share their game related information online
- **iAd Framework –** allows to deliver banner-based advertisements from app.
- **MapKit Framework –** gives a scrollable map that can include into user interface of app.
- **PushKit Framework –** provides registration support for VoIP apps.
- **Twitter Framework –** supports a UI for generating tweets and support for creating URLs to access the Twitter service.
- **UIKit Framework –** gives vital infrastructure for applying graphical, event-driven apps in iOS. Some of the Important functions of UI Kit framework:
  - Multitasking support.
  - Basic app management and infrastructure.
  - User interface management
  - Support for Touch and Motion event.
  - Cut, copy and paste support and many more.

Jailbreaking iOS

- Taking control of device during booting process
  - Obtain firmware image (IPSW) that corresponds to the iOS version and device model that needs to be jailbroken
  - Obtain jailbreak software (redsn0w, greenpois0n, limera1n)
  - Connect the device to the computer hosting the jailbreak software via the standard USB cable
  - Launch the jailbreak application and select the previously downloaded IPSW on jailbreak s/w console
  - Jailbreak software typically customizes the IPSW
  - Switch the device into Device Firmware Update (DFU) mode. To do this, the device should be powered off.
  - Once the switch into DFU mode occurs, the jailbreak software automatically begins the jailbreak process. Wait until the process completes.

Jailbreaking iOS

- Remote Jailbreaking (jailbreakme.com)
  - Loads a specially crafted PDF into mobile safari browser.
  - The PDF takes control of browser, operating system and provides user full control of devices
  - Another option is to load home page of jailbreakme.com in browser and press INSTALL button (jailbreakme3.0)

Thank You