#HITB2023AMS https://conference.hitb.org/

HITB 2023 AMS

Feeding Gophers to Ghidra

Max ‘Libra’ Kersten | Malware Analyst | Trellix

Table of contents

- About me
- Java
- Understanding Ghidra
- Languages and difficulties
- Golang
- Live demo
- Q&A

About me

- Max 'Libra' Kersten (@Libranalysis, @libra@infosec.exchange)
- Malware analyst and reverse engineer
- Working for Trellix' Advanced Research Center
  - Published DotDumper
- I write blogs about reverse engineering
  - Including my free Binary Analysis Course
- My tools are open-sourced on GitHub
  - Such as AndroidProjectCreator and the Mobile Malware Mimicking Framework

About Gophers

- The official Golang mascot
- No Gophers were harmed during the research

Created by Renee French, more info on the Golang blog

About Ghidra

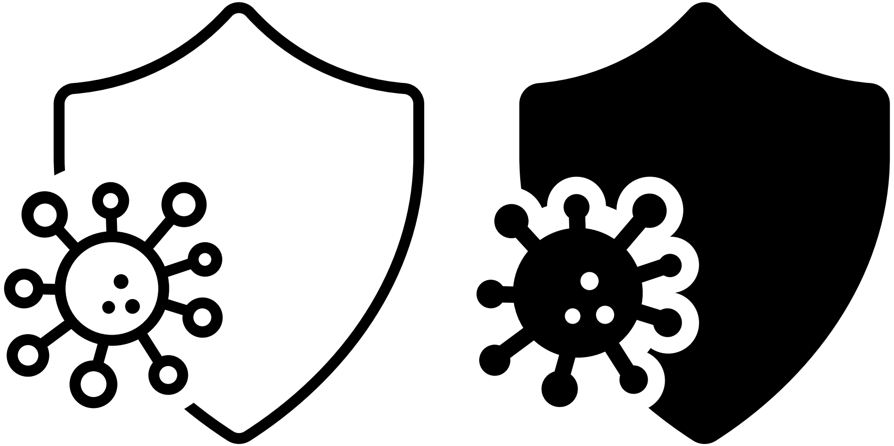
- Software reverse-engineering framework
- Created by the NSA
- Open-sourced in 2019

Java

- Ghidra's native tongue
  - Jython supports Python 2.7, and is included by default
  - Python 3 bridges exist as external plug-ins
- Not the most favoured language
  - Though it is by me
- Java and Jython related code can be debugged in Eclipse
  - With the Ghidra-dev plug-in

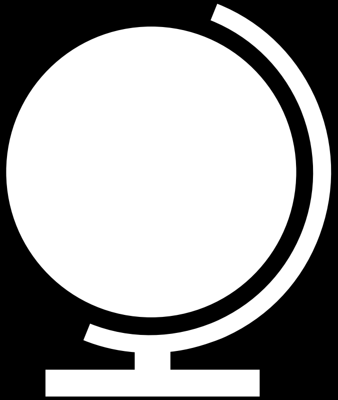
Understanding Ghidra

- Live demo



Languages and difficulties

- Approaches to binary languages
  - Differ based on the architecture
  - Contain re-used and unique concepts
  - No magic catch-all solution
- Cross-platform capabilities
  - Are a double-edged sword
  - Easy-of-use for developers
  - Multi-architecture analysis required
  - Re-used concepts save time



Golang

- The Golang runtime is embedded
  - Statically compiled
- Compiler allows the creation of stripped binaries
  - Though they actually aren't
- Cross platform compatibility

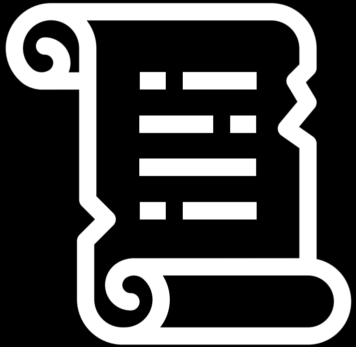
Golang analysis

- Difficult to see what is (not) runtime related
- Disassembly and decompiled code are cluttered
- Often approached as a C-like binary
  - Can be done successful
  - Begs the question: is the invested time worth it?

- Interested in
  - Strings
  - Functions
  - Types

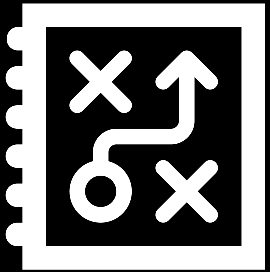
Golang analysis scripts

- Based on the public work of Dorka Palotay
  - She works for CUJO
- This is not the first project to use (similar) concepts



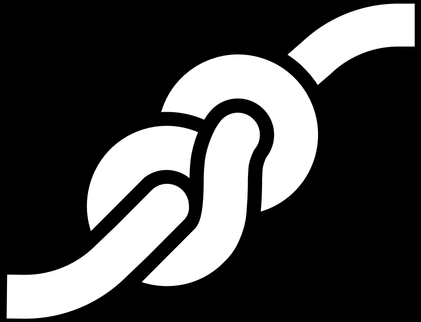
Golang analysis approach

- Static string recovery
- Dynamic string recovery
- Function name discovery and recovery
- Type recovery



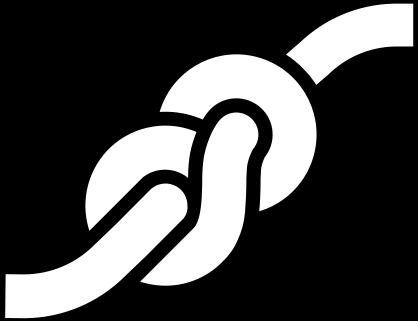
Static string recovery

- Iterates over `.rodata` and `.data`
- Performs sanity checks
- Creates the string (pointer)



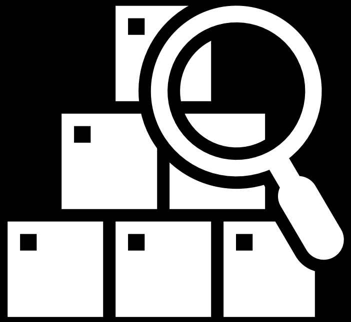
Dynamic string recovery

- Pattern based instruction matching
  - On a per architecture basis
- Parses instructions
  - Gets the operand values
- Creates the string (pointer)



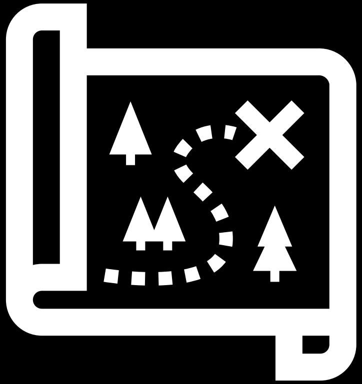
Function name recovery

- Searches for the `pclntab`
- Parses said tab
- Creates or renames functions



Type recovery

- Searches for different sections and structures
- Parse said structures
- Rename corresponding types



Results – SwiftSlicer

Live demo

Q&A

For questions, you can also reach out to me via @Libranalysis, @libra@infosec.exchange, or Max Kersten on LinkedIn