Ex No-10 Use Ghidra to disassemble and analyze the malware code.

AIM:

To use **Ghidra** to safely disassemble, decompile, and analyze a suspicious binary in order to identify its functionality, extract indicators of compromise (IOCs), and document malicious behaviour for forensic reporting.

DESCRIPTION / THEORY:

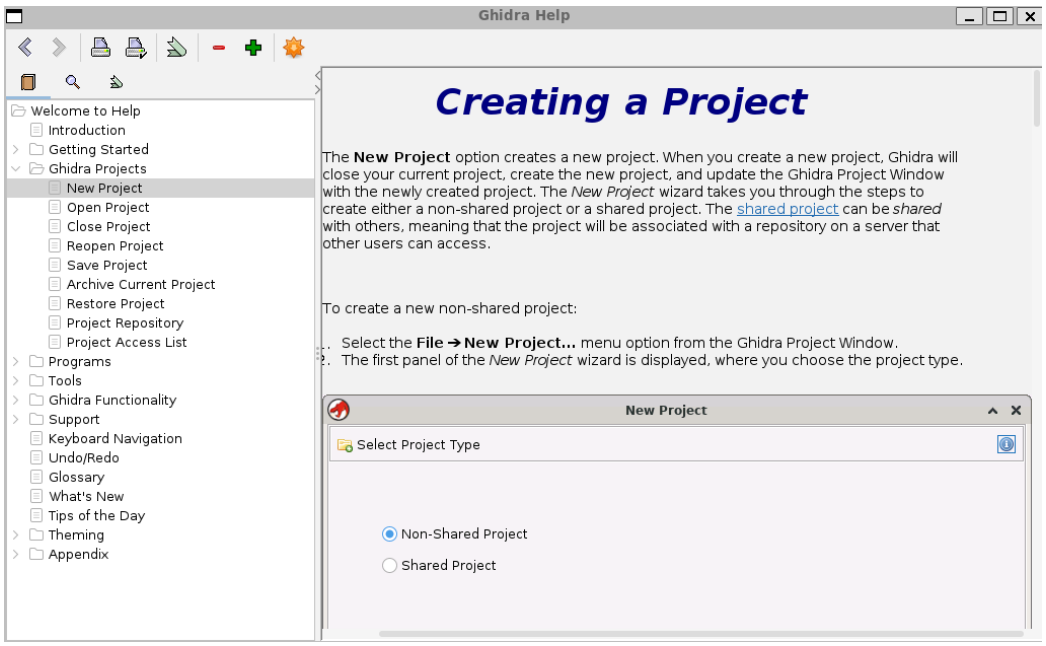
Ghidra is a free, open-source software reverse-engineering framework developed by the NSA. It provides disassembly, a high-quality decompiler, cross-reference views, symbol and type management, and scripting support. When analysing a binary, investigators use static analysis (no execution) to inspect program structure, function boundaries, strings, control flow, imported APIs, and data references.

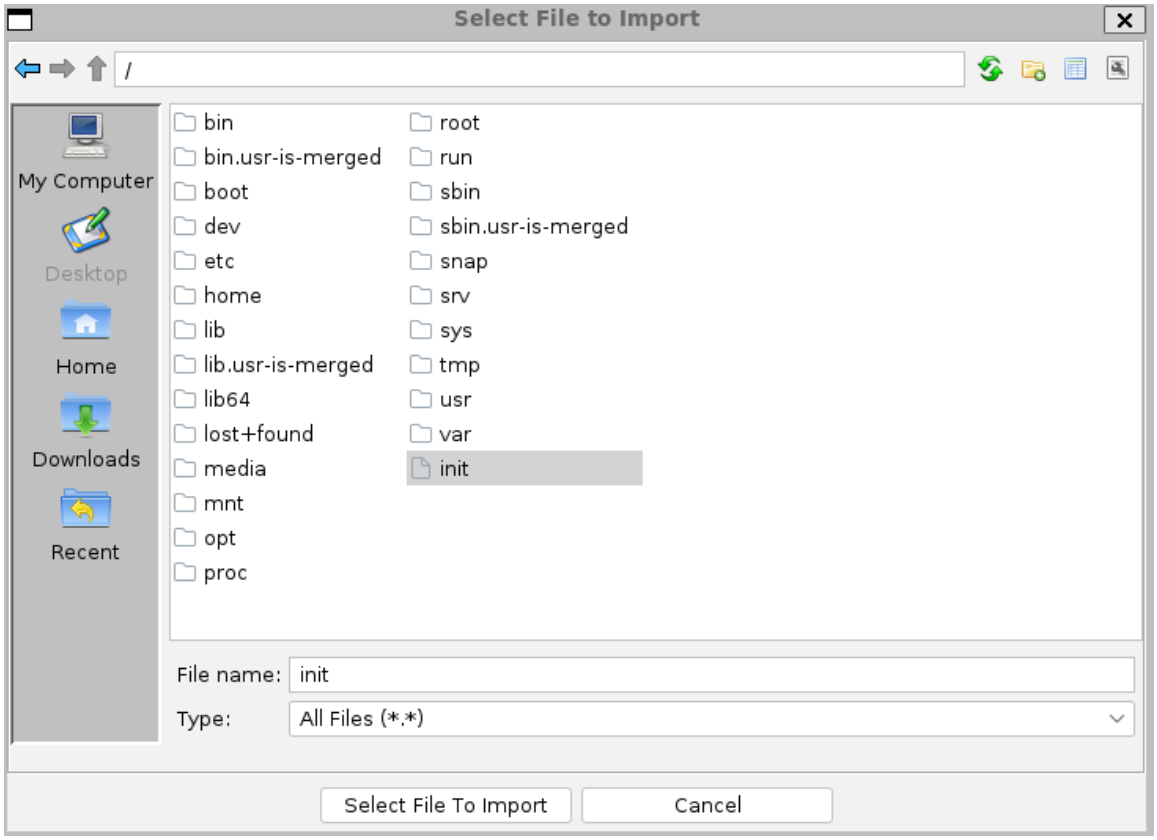
Static analysis with Ghidra helps answer questions such as: What does the binary do? Which libraries/APIs does it call (network, persistence, process injection)? Where are interesting strings, file paths, or URLs? Are there obfuscated routines? The goal is to derive safe, documented findings without running the sample on a production machine.

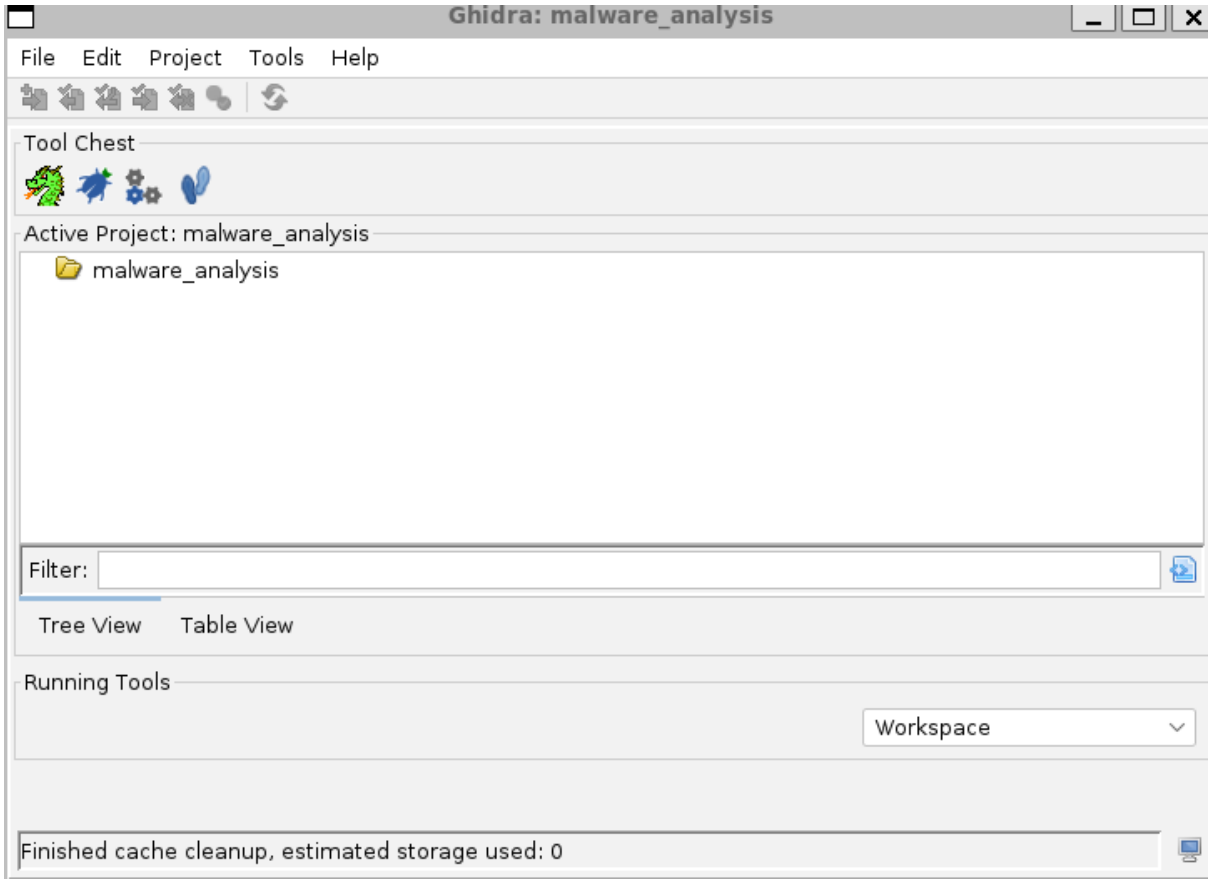
PROCEDURE:

```
binaya@LAPTOP-F1KG4QN9:/mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826$ ls
ghidra_11.4.2_PUBLIC
binaya@LAPTOP-F1KG4QN9:/mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826$ # go to ghidra folder
cd /mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826

# list files to check existence
ls -la

# if ghidraRun exists but is a DOS file, convert and make executable
sudo apt update
sudo apt install -y dos2unix   # if not installed

# convert and chmod (safe even if already unix)
dos2unix ghidraRun 2>/dev/null || true
chmod +x ghidraRun

# If you use an X server (VcXsrv/WSLg), make sure DISPLAY is set (WSLg sets it automatically)
export DISPLAY=:0   # only needed if you run VcXsrv and DISPLAY not set

# Start Ghidra GUI (blocks terminal)
./ghidraRun
# OR to run in background:
# ./ghidraRun &>/dev/null & disown
total 0
drwxrwxrwx 1 binaya binaya 512 Oct 23 14:39 .
drwxrwxrwx 1 binaya binaya 512 Oct 27 06:45 ..
drwxrwxrwx 1 binaya binaya 512 Oct 23 14:39 ghidra_11.4.2_PUBLIC
[sudo] password for binaya:
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:4 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Fetched 126 kB in 3s (41.6 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
77 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
```

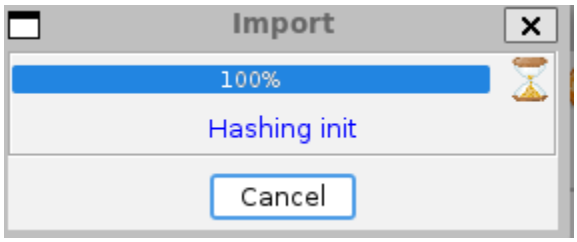
```
binaya@LAPTOP-F1KG4QN9:/mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826$ GHIDRA_ROOT="/mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826"
PROJECT_DIR=~/ghidra_projects     # Ghidra will create this directory for projects
PROJECT_NAME="HeadlessProject"

# Run analyzeHeadless
"$GHIDRA_ROOT/support/analyzeHeadless" "$PROJECT_DIR" "$PROJECT_NAME" \
  -import "$SAMPLE" \
  -scriptPath ~/ghidra_scripts \
  -postScript export_iocs.py \
  -overwrite \
  -noanalysis  # We'll run analysis explicitly to control timeout
-bash: /mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826/support/analyzeHeadless: No such file or directory
binaya@LAPTOP-F1KG4QN9:/mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826$ # Run analysis with a per-file timeout (minutes). Adjust 5 -> large
"$GHIDRA_ROOT/support/analyzeHeadless" "$PROJECT_DIR" "$PROJECT_NAME" \
  -process "$SAMPLE_BASENAME" \
  -analysisTimeoutPerFile 10 \
  -scriptPath ~/ghidra_scripts \
  -postScript export_iocs.py \
  -overwrite
-bash: /mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826/support/analyzeHeadless: No such file or directory
binaya@LAPTOP-F1KG4QN9:/mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826$ # change to outdir and run analyzeHeadless so iocs.txt l
cd "$OUTDIR"
"$GHIDRA_ROOT/support/analyzeHeadless" "$PROJECT_DIR" "$PROJECT_NAME" \
  -import "$SAMPLE" \
  -scriptPath ~/ghidra_scripts \
  -postScript export_iocs.py \
  -analysisTimeoutPerFile 10 \
  -overwrite
-bash: /mnt/d/DF/ghidra_11.4.2_PUBLIC_20250826/support/analyzeHeadless: No such file or directory
binaya@LAPTOP-F1KG4QN9:~/ghidra_outputs/malware.bin$
```

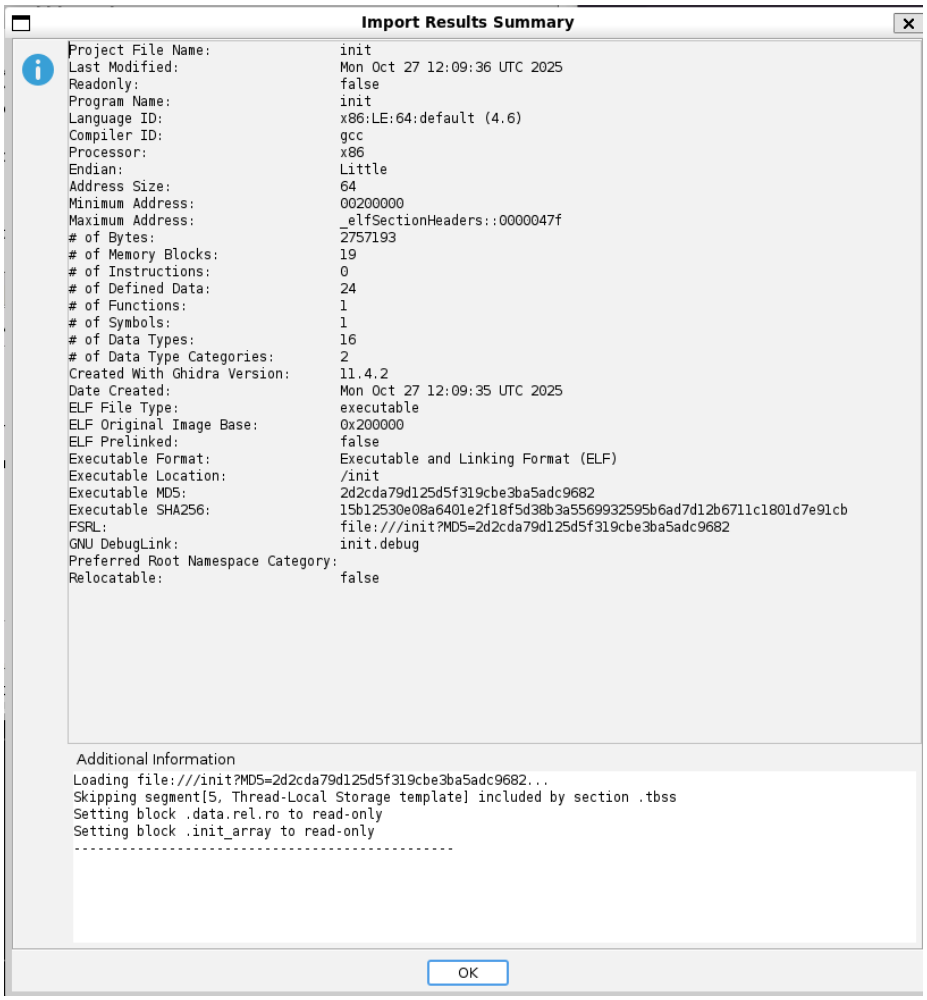



Select File to Import
/
My Computer
Desktop
Home
Downloads
Recent
bin
bin.usr-is-merged
boot
dev
etc
home
lib
lib.usr-is-merged
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
sbin.usr-is-merged
snap
srv
sys
tmp
usr
var
init
File name: init
Type: All Files (*.*)
Select File To Import
Cancel


Ghidra: malware_analysis
File Edit Project Tools Help
Tool Chest
Active Project: malware_analysis
malware_analysis
Filter:
Tree View
Table View
Running Tools
Workspace
Finished cache cleanup, estimated storage used: 0

OUTPUT:





RESULT:

The **Ghidra tool** was successfully used to disassemble and analyze the malware code. The experiment revealed the binary's structure, extracted hidden strings, identified suspicious API calls, and helped understand the malware's functionality without executing it.