GDB for incident responders - an overview

Taz Wake | Halkyn Consulting Ltd | Originally published 2023-11-29

The GNU Debugger (GDB) is a sophisticated tool in Linux for debugging applications and analysing complex software issues, including malware analysis. This article provides an overview of GDB's functionality, some examples of how to use it, and its application in incident response, particularly in analysing Linux malware.

GDB in action

GDB is a powerful debugger in the Linux environment, offering a detailed examination of program execution and state. Its capabilities are essential for software developers and security professionals alike.

Compiler Options for Effective Debugging

For optimal use of GDB, it's important to compile applications with specific flags: -g3 for detailed debug information and -O0 to disable optimisation. These settings ensure that GDB can access comprehensive information about the application during debugging.

GDB Startup Scripts

Upon startup, GDB executes commands from various script files: system default initialization in /etc/gdbinit, user's global initialisation in $HOME/.gdbinit, and application-specific customizations in ./.gdbinit.

Utilising GDB's Help System

GDB offers a built-in help system accessible through the help and apropos commands, providing usage information for any command or option.

Starting and Using GDB

The basic way to start GDB is by sending the application's name as an argument. GDB then loads the program and its debug information, ready for debugging tasks.

Passing Arguments and Attaching to Processes

GDB allows passing command-line arguments to the program via the --args option. For analysing already running processes, the --pid option is used to attach GDB to the process.

Analysing Crashes and Automating Commands

The --core option in GDB is useful for loading and analysing core dumps post-crash. GDB also provides --ex, --iex, --x, and --batch options for expedited command execution, particularly useful in automated or repetitive debugging tasks.

Breakpoints and GDB

Breakpoints are a fundamental aspect of debugging with GDB, allowing the user to pause program execution at specific points. This feature is incredibly useful for examining the state of a programme at critical moments or before a suspected error occurs.

Setting Breakpoints

To set a breakpoint in GDB, use the command (abbreviated as), followed by a specific location in the code. This location can be specified in various ways:

1. **By Function Name**:
2. **By Line Number**:
3. **By Address**:

For example, to set a breakpoint at line 50 of a file named , you would enter .

Conditional Breakpoints

GDB also allows setting conditional breakpoints, which pause the programme only when a specified condition is true. This is done by appending a condition to the command: .

For instance, to break on line 100 only if the variable is greater than 10, use: .

Managing Breakpoints

Once breakpoints are set, GDB provides commands to manage them:

- **List Breakpoints**:
- **Disable Breakpoints**:
- **Enable Breakpoints**:
- **Delete Breakpoints**:

Tips for Effective Breakpoint Usage

- Use breakpoints to isolate problematic sections of code.
- Conditional breakpoints are particularly useful for observing behaviour under specific circumstances without stopping the programme at every iteration.
- Remember to manage breakpoints effectively, especially in complex debugging sessions, to maintain clarity and focus.

Incorporating breakpoints effectively in GDB debugging sessions enhances the ability to diagnose and resolve issues efficiently, making them an indispensable tool for developers and analysts working in Linux environments.

Incident Response: Key Checks for Linux Malware Analysis

Incident responders using GDB for Linux malware analysis should focus on:

1. **Examining Running Processes**: Attaching to suspicious processes to inspect their state and behaviour.
2. **Crash Analysis**: Analysing core dumps to understand the cause of crashes, which may indicate malicious activity.
3. **Disassembly and Debugging**: Stepping through the code at an assembly level to uncover the functionality and intent of the malware.

Recent Linux Malware Analysis

Recent reports highlight various malware types targeting Linux systems, including ransomware, cryptocurrency miners, web shells, and rootkits. Commonly exploited vulnerabilities include unpatched software, misconfigurations, insecure code, and phishing attacks. Notable examples include KillDisk ransomware, XMRig cryptocurrency miner, and various rootkits. These malware types leverage different vulnerabilities and attack vectors, requiring comprehensive analysis techniques.

GDB in Action: Analysing Recent Linux Malware

In the context of these threats, GDB can be instrumental in dissecting malware binaries. By setting breakpoints, stepping through instructions, and inspecting memory and register states, analysts can gain insights into the malware's operations and strategies. This approach is particularly effective against complex threats like those exploiting protocol anomalies or writing rootkits.

Conclusion

GDB's depth and versatility make it a vital tool in the Linux ecosystem for both development and security purposes. Its ability to dissect and analyse software at a granular level is crucial for understanding and mitigating the effects of Linux malware, an ever-growing concern in the cybersecurity landscape. Understanding and proficiently using GDB is, therefore, an essential skill for developers and security professionals dealing with Linux systems.

Note - this article looks to give a high-level overview of GDB rather than in-depth details. If you want to know more about analysing running processes have a look at: https://sans.org/for610 Reverse Engineering Malware and https://sans.org/sec660 Advanced Penetration Testing, Exploit Writing, and Ethical Hacking

About Halkyn Consulting

Halkyn Consulting Ltd is a UK-based security and risk management consultancy providing 24/7 incident response, digital forensics, threat hunting, virtual security leadership (vCISO) and security compliance support to organisations worldwide. We deliver clear, practical, vendor-independent advice based on real-world threats — no hype, no upsell.

More free security resources are available at www.halkynconsulting.co.uk/security-resources.

Web: www.halkynconsulting.co.uk | **Email:** info@halkynconsulting.co.uk | **Tel:** +44 1244 940858