Secure Erase via Custom Initramfs and EFI Boot Configuration

Overview

To perform a secure erase on Shift5 edge compute devices upon next reboot — without mounting the device's main storage — this design uses a **custom initramfs** paired with a temporary **EFI boot entry**.

The solution ensures that the secure erase command is executed **before the OS boots** and the storage device is mounted.

1. **Create a Custom Initramfs**

A. init Script (Runs as PID 1 in initramfs)

```
#!/bin/sh
# Minimal secure erase script for initramfs

set -e

echo "=== Shift5 Secure Erase Boot ==="
echo "Starting secure erase..."
sleep 3  # Optional safety delay

# Perform secure erase
cryptsetup erase -q /dev/sda3
blkdiscard -z /dev/sda

echo "Secure erase completed successfully."
echo "Powering off..."
sleep 2
poweroff -f
```

This script must be placed at the root of the initramfs as /init, marked executable, and must not rely on the system root being mounted.

B. Initramfs Directory Structure

```
secure-erase-initramfs/
├── bin/
│   ├── sh -> busybox
│   ├── busybox
│   ├── cryptsetup
│   └── blkdiscard
├── dev/               # Created at runtime
├── etc/
├── proc/
├── sys/
├── tmp/
├── init               # Your init script (chmod +x)
└── lib/               # Required libraries (from ldd)
```

Use BusyBox for minimal utilities. Make sure binaries like cryptsetup and blkdiscard are statically linked or include required lib/ dependencies.

C. Build the Initramfs

From inside the **secure-erase-initramfs/** directory:

```
find . | cpio -H newc -o | gzip > ../secure-erase-initramfs.img
```

This produces a **secure-erase-initramfs.img** usable during EFI boot.

---

2. **Modify EFI Boot Logic**

A. Install Your Custom EFI Files

Assuming the EFI System Partition is mounted at /boot/efi:

```
mkdir -p /boot/efi/EFI/secureerase
cp secure-erase-initramfs.img /boot/efi/EFI/secureerase/initramfs.img
cp /boot/vmlinuz-linux /boot/efi/EFI/secureerase/vmlinuz
```

Use a kernel compiled with EFI stub support.

B. Create One-Time EFI Boot Entry

Using efibootmgr:

```
efibootmgr \
  --create \
  --disk /dev/sda --part 1 \
  --label "Secure Erase" \
  --loader /EFI/secureerase/vmlinuz \
  --unicode 'root=/dev/ram0 initrd=\EFI\secureerase\initramfs.img console=ttyS0' \
  --bootnext XXXX
```

- This sets a **one-time boot** to the secure erase payload
- Replace XXXX with the actual Boot ID if needed
- Kernel must support booting without mounting /dev/sda

---

3. **On Reboot**

- The system boots into the secure-erase initramfs
- The init script runs
- Disk is securely wiped (/dev/sda3 and /dev/sda)
- System powers off immediately after

---

4. **Post-Erase Behavior**

After wipe completion:

- System can be manually re-imaged or PXE-booted into a recovery environment
- If the bootloader (shift5.efi) was temporarily replaced instead of using efibootmgr, restore it manually or from the initramfs script

---

Summary

This solution:

- Leverages a **custom initramfs** to securely erase the disk before mounting
- Uses **standard Linux tools** (cryptsetup, blkdiscard)
- Integrates cleanly with existing **EFI boot mechanisms**
- Avoids OS corruption by powering off immediately after erase