

```
import winreg
import time
import os
import logging
from datetime import datetime

# =====
# PROJECT 3: WINDOWS REGISTRY MONITOR (HIDS)
# AUTHOR: Raj Bharti
# PURPOSE: Detect persistence mechanisms by monitoring 'Run' keys.
# SAFETY: This script provides READ-ONLY access to the Registry.
# =====

# --- CONFIGURATION ---
# The specific path in the Registry we want to watch (Startup Programs)
REG_PATH = r"Software\Microsoft\Windows\CurrentVersion\Run"
LOG_FILE = "registry_integrity.log"

# Setup Logging
logging.basicConfig(filename=LOG_FILE, level=logging.INFO,
                    format='%(asctime)s - %(message)s')

def get_registry_values():
    """
    Connects to the Windows Registry and takes a snapshot of the current
    startup programs.
    Returns: A dictionary {ProgramName: ProgramPath}
    """
    registry_snapshot = {}
    try:
        # Connect to HKEY_CURRENT_USER (HKCU) - Safe for testing
        key = winreg.OpenKey(winreg.HKEY_CURRENT_USER, REG_PATH, 0,
                             winreg.KEY_READ)

        # Loop through all values in this key
        index = 0
        while True:
            try:
                # v_name = Program Name, v_data = Path to Exe
                v_name, v_data, _ = winreg.EnumValue(key, index)
                registry_snapshot[v_name] = v_data
                index += 1
            except OSError:
                break # No more values

        winreg.CloseKey(key)
        return registry_snapshot
    except Exception as e:
        print(f"[!] Error accessing Registry: {e}")
```

```

    return {}

def monitor_registry():
    print("====")
    print("  WINDOWS REGISTRY MONITOR - PERSISTENCE HUNTER  ")
    print("====")
    print(f"[*] Monitoring Key: HKCU\\{REG_PATH}")

    # 1. ESTABLISH BASELINE
    print("[*] Taking initial Baseline snapshot...")
    baseline = get_registry_values()
    print(f"[*] Baseline established. Found {len(baseline)} entries.")
    for name, path in baseline.items():
        print(f"    - {name}: {path}")

    print("\n[*] MONITORING ACTIVE. Keep this window open.")
    print("[*] Press Ctrl+C to stop.")

    # 2. CONTINUOUS MONITORING LOOP
    try:
        while True:
            time.sleep(3) # Check every 3 seconds
            current_snap = get_registry_values()

            # CHECK FOR ADDITIONS (New Malware Persistence?)
            for name in current_snap:
                if name not in baseline:
                    msg = f"ALERT! NEW REGISTRY VALUE DETECTED: {name} ->
{current_snap[name]}"
                    print(f"\n[!] {msg}")
                    logging.warning(msg)
                    # Update baseline so we don't alert forever on the same change
                    baseline[name] = current_snap[name]

            # CHECK FOR DELETIONS (Cleanup?)
            for name in list(baseline):
                if name not in current_snap:
                    msg = f"NOTICE: Registry Value Removed: {name}"
                    print(f"\n[-] {msg}")
                    logging.info(msg)
                    del baseline[name]

    except KeyboardInterrupt:
        print("\n[*] Monitoring Stopped.")

if __name__ == "__main__":
    monitor_registry()

```