# Windows Service & Process Monitoring Agent
# (Project Documentation)

## 1. Project Overview / Description

This project focuses on building a Windows Service & Process Monitoring Agent capable of detecting malicious, unauthorized, or suspicious process behavior.

Windows services and processes are common targets for malware, persistence mechanisms, privilege escalation, and stealth operations.

The monitoring agent analyzes service configurations, process hierarchies, startup entries, and runtime activity to identify anomalies.

This framework provides:

- Suspicious parent–child relationship detection

- Startup service auditing

- Unauthorized process monitoring

## 2. Practical Motivation

Windows systems are frequently compromised through:

- Malicious services registered under system privileges

- Abnormal parent–child process relationships used in malware execution

- Startup manipulation for persistence

- Rogue processes mimicking legitimate names

Monitoring these behaviors helps detect:

- Malware activity

- Intrusion attempts

- Privilege escalation

- Persistence mechanisms

This project gives practical SOC, IR, and threat detection experience.

## 3. Project Objectives

1. Monitor active Windows processes and analyze parent–child behavior.

2. Audit startup services for suspicious, newly registered, or modified services.

3. Detect unauthorized or unknown processes running on the system.

4. Generate alerts based on rule-based or behavior-based triggers.

5. Produce a detailed report summarizing anomalies and threats.

## 4. Practical Scope of the Project

A. Parent–Child Relationship Monitor:

- Track process lineage using Process IDs and Parent Process IDs.

- Detect anomalous relationships (e.g., cmd.exe spawned from winword.exe).

- Identify process injection signs or suspicious process chains.

B. Startup Service Audit:

- Enumerate all Windows services at startup.

- Identify services with unusual file paths or newly added services.

- Detect service permission misconfigurations allowing escalation.

C. Unauthorized Process Detection:

- Maintain a whitelist/blacklist of allowed processes.

- Detect unknown, unsigned, or high-risk processes.

- Identify processes running from temporary or user-writeable directories.

D. Reporting & Alert System:

- Log event details (timestamp, process name, PID, path).

- Flag high-severity events.

- Generate consolidated detection reports.

## 5. Tools & Technologies Used

Programming Languages:

- Python (recommended)

- PowerShell (optional)

Modules/Tools:

- psutil (process enumeration)

- wmi (Windows Management Interface)

- win32service / win32process (optional)

- PowerShell Get-Service, Get-Process

Documentation Tools:

- Word / Google Docs

- Draw.io for diagrams

## 6. Practical Techniques Implemented

Detection Techniques:

- Behavior monitoring of parent–child process trees

- Service configuration auditing

- Whitelist/blacklist-driven detection

- Identifying persistence or escalation techniques

Blue Team Techniques:

- Real-time process monitoring

- Detection of suspicious runtime activity

- Identifying malicious service entries

- Strengthening system security baselines

## 7. Workflow / Architecture (Practical Explanation)

STEP 1: Enumerate Active Processes

- Capture PID, PPID, names, file paths, signatures.

STEP 2: Build Parent–Child Process Tree

- Analyze chains for anomalies.

STEP 3: Audit Startup Services

- Extract service configurations, paths, permissions.

STEP 4: Unauthorized Process Detection

- Compare processes against whitelist/blacklist.

STEP 5: Generate Alerts

- Based on suspicious behavior rules.

STEP 6: Reporting

- Provide structured logs and reports summarizing detection results.

## 8. Flowchart (Text Version)

START

↓

Enumerate Processes & Services

↓

Analyze Process Tree (Parent–Child)

↓

Audit Startup Services

↓

Detect Unauthorized / Anomalous Processes

↓

Generate Alerts

↓

Export Final Detection Report

↓

END

## 9. Expected Practical Output

The monitoring agent should output:

- Parent–child relationship anomalies

- Suspicious startup services

- Unauthorized processes

- Timestamped monitoring logs

- Final detection summary report

Examples:

- Suspicious child process detected: winword.exe → powershell.exe

- New startup service detected: UnknownServiceXYZ

- Unauthorized process running from temp folder

## 10. Learning Outcomes

This project teaches:

- Windows process architecture and service internals

- How malware abuses services and processes

- Real-time monitoring techniques

- Rule-based detection logic

- Defensive security engineering

## 11. Project Deliverables

1. Project documentation (Word/PDF)

2. Monitoring agent toolkit

3. Logs/screenshots of monitoring results

4. Flowcharts & architecture diagrams

5. Detection rules used

6. Final presentation (PPT)