9/1/2023

Simple Anti-Malware Tool

Kit325

Louis Reay-Smith #552768
Mohammad Rahi Al Towhid #551098
Ihala Ihalakorala #627871

Project Goal

The main project goal is to develop a malware detection tool using a gray list approach that primarily focuses on detecting and removing crypto jacking software, designed to run on Windows 10 and/or 11 computers.

The tool will work by scanning the running processes and comparing them to a database of known malware, as well as recognizing anomalies by comparing the CPU, RAM and GPU to a baseline monitoring dataset of CPU, RAM, and GPU usage. A user-friendly UI is a nice-to-have feature to be implemented if time permits. The tool will also prompt the user if they want to shut down the detected abnormal processes and will do so after a user confirmation.

Technical Goals

The main technical goal is to create a program that scans the running processes and takes in their names, also logging the usage of the CPU, RAM and GPU of the system.
The processes taken should be compared to an established database of known crypto jacking software processes and flag the ones that match from the database. In an ideal scenario this database would be hosted on an online server, however with this software being designed as a proof of concept it was decided to host the malicious process database locally.
The usage of the CPU, RAM, and GPU must be logged and compared with a pre-configured snapshot of baseline values and identify any abnormal usage values.
The software will aim to isolate the identified abnormal processes, and send an alert to the user regarding those processes while also displaying them in a separate listed view.
Functionality must be developed to get the user input regarding the separated processes.
Based on the input from the user, functions have to be developed on the actions about the processes, whitelisting them, or blacklisting them and shutting them down.
If it is possible, develop a user interface for the tools to be accessed easily by the user, if not, arrange an understandable intuitive command line for the user to interact with.

Challenges

A major potential challenge to overcome is to address the issues that our tool might encounter with Windows security mechanisms that may identify our tool as a threat. As Windows 10/11 has higher security features, this could come as a major potential challenge.
The processes could also be mislabelled. This may be legitimate processes being labelled as malware or malware being able pass through the tool without detection. The user might have legitimate mining software running on their computer which could be mislabeled and shut down by our tool.
Another challenge is our tool consuming a large amount of resources, thus breaking the baseline comparison feature. This could also disrupt the user's work by slowing down the computer. The tool should be optimized as much as possible in the limited allocated time to overcome that challenge.
The last potential challenge is that the User Interface or Terminal interface may be less user friendly and confusing for the user to navigate, making the user reluctant or unable to use the tool. Consideration must be given to the user experience when designing the front end of the tool.

Requirement Matrix

1. Software will start on system boot
2. Software will create a list of running processes
3. Software recognises non-approved processes
4. Software is able to present list to user
5. Software will ignore whitelisted processes
6. Software will be able to determine if a process is suspicious based on the process's resource usage.
7. Software will prompt the user of suspicious process on the users computer
8. Software will be able to blacklist certain processes via a user prompt
9. Software will be able to whitelist certain processes via a user prompt
10. Software will automatically kill malicious/blacklisted processes
11. Software is able to kill a targeted process
12. Software will be able to trace a processes source and related information and present it to the user
13. Software will be able to modify the black/white list of processes specific to each user

NTH’s

Software will isolate and remove the source of a suspicious process

Software will block the source of an online suspicious process

Software will receive updates about malicious process via an online server

Software will alert the user to the source of suspicious processes

Structured scenarios for each use case

UC1 Software_Initialization

Requirement: On software start-up, the software will start monitoring and create a list of background processes.

Overview: Upon computer boot up, UI will present a message to the user regarding the process running in the background. Any unusual software or process will be identified in the system monitor. This will be determined via excessive or unusual usage of the processor, GPU and RAM.

Pre-conditions:

1. A database on a system running perfectly without any software or malware installed to compare with.

2. Proper system boot-up

Scenario:

Action	Software Reaction
1. User starts application	1. Software creates a list of all currently running processes 2. Software compiles all currently processes into a list 3. Software marks non-approved processes for removal.
2. User selects a “view current processes” option	1. Software retrieves the list of current processes. 2. Software displays a list to the user.

Scenario notes: User independently starts up the malware detection tool. User must complete action 1 for this use case.

Exceptions: Anti malware tool starting automatically if unusual memory consumption is noticed right after system boot.

Requirements Met: 1,2,3,4

UC2 Software_catagorizing_processes

Requirement: Software will categorize the running processes as “whitelisted” , “ greylisted” and “blacklisted” .

Overview: Upon software start-up, it will catagorize all the running processes in three categories. Whitelisted processes will have a database of approved software which is running with admin approval. Blacklisted category will consist of applications that are being blocked from running in the system by the admin. Greylisted database will trace the softwares that are running but does not match with whitelisted database or blacklisted database. Greylisted processes might be a potential risk to the system.

Pre-conditions:

1. A “whitelist” or approved software database.
2. A “blacklist” or blocked software database.
3. All software not under “whitelist” or “blacklist” will automatically fall under greylist.

Scenario:

Action	Software Reaction
1. Software completes a successful bootup	1. Software goes through the list of current processes, ignoring whitelisted processes. 2. Each process will have it resource usage tested to determine if it is with expected usage. 3. Processes with unexpected usage will be flagged as suspicious.
2. Software finds suspicious process	1. Software will send a prompt regarding the process to the user along with other explanatory data. 2. Software will give the user the option to blacklist or whitelist.
3. User whitelists a process	1. System removes the process from any other existing list if it is marked to be on any and unchecks the related boolean. 2. System marks the Processes WhiteList boolean as true. 3. System adds the process to a whitelist process list.
4. User Blacklists a process	1. System removes the process from any other existing list if it is marked to be on any and unchecks the related boolean. 2. System marks the Processes BlackList boolean as true. 3. System adds the process to a Blacklist process list.

Scenario notes: User must have a database on whitelisted and blacklisted programs. Action 3 and 4 completes automatically and does not require users’ attention.

Exceptions: High memory consuming whitelisted program, for example Google Chrome can show different results and/or be listed under greylisted category.

Requirements Met: 5,6,7,8,9

UC3 Software_Interacting_with_Processes

Requirement: The software will be able to suspend or kill processes that are black listed or deemed potentially malicious or that are listed on a blacklist.

Overview: The software can trace any processes related information and present it to the user. The user can modify their black/white list of processes if they deem necessary. This modification might include killing or suspending the processes.

Pre-conditions:

1. System tracing the processes and listed into it's category
2. User must not kill or suspend a process that is not listed. This means that process is not running or suspended previously.

Scenario:

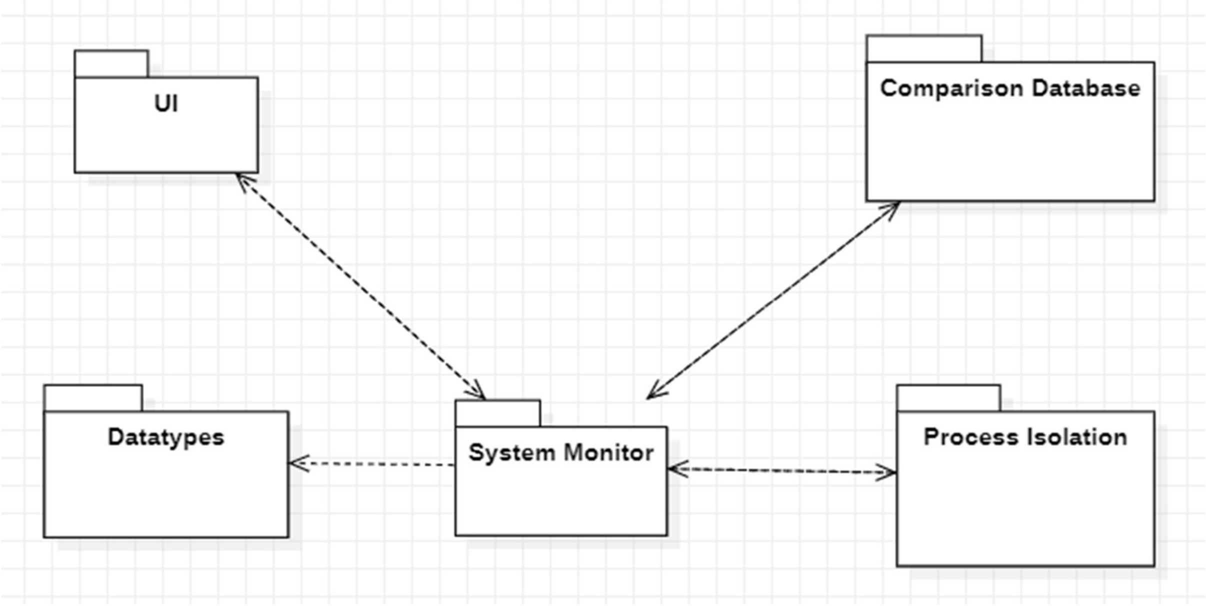
Action	Software Reaction
1. System finds a Process that is blacklisted	1. System attempts to kill the process 2. System checks if the process has been stopped.
2. System fails to suspend a process	1. System will attempt to determine the source of the process 2. System will alert the user to the source of a blacklist process
3. User selects a process from a Process List	1. System will attempt to determine the source of the process 2. System will return the information to the user.
4. User selects a process from a Process List and selects to remove it from a black/white List it	1. System removes the process from from the selected list along with its associated boolean check 2. System notifies user process has been removed from the list

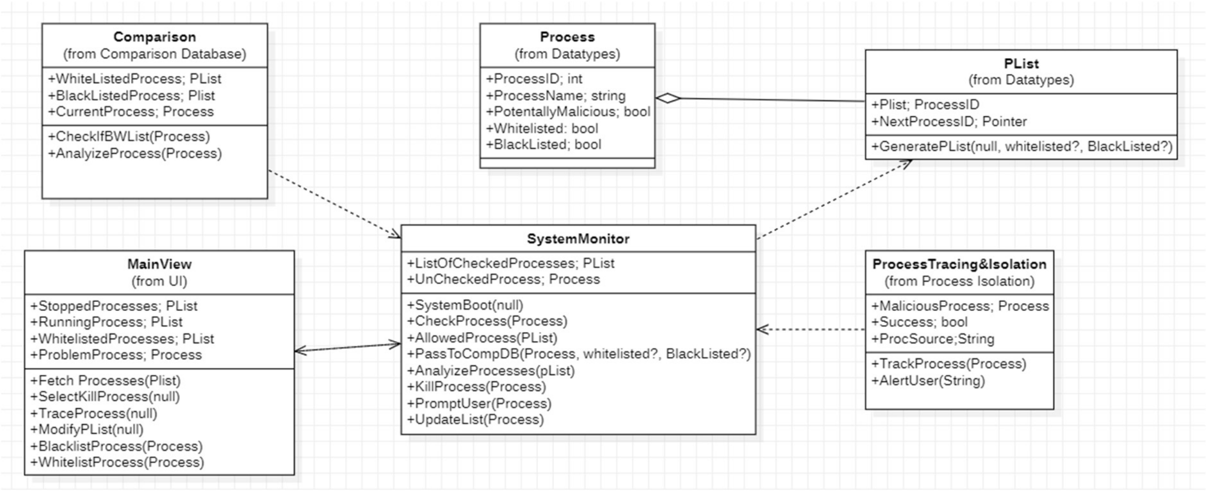
Scenario notes: User has the authority to change a program. They can modify the program from a blacklist.

Exceptions: Greylisted software can be killed without modifying.

Requirements Met: 10,11,12,13

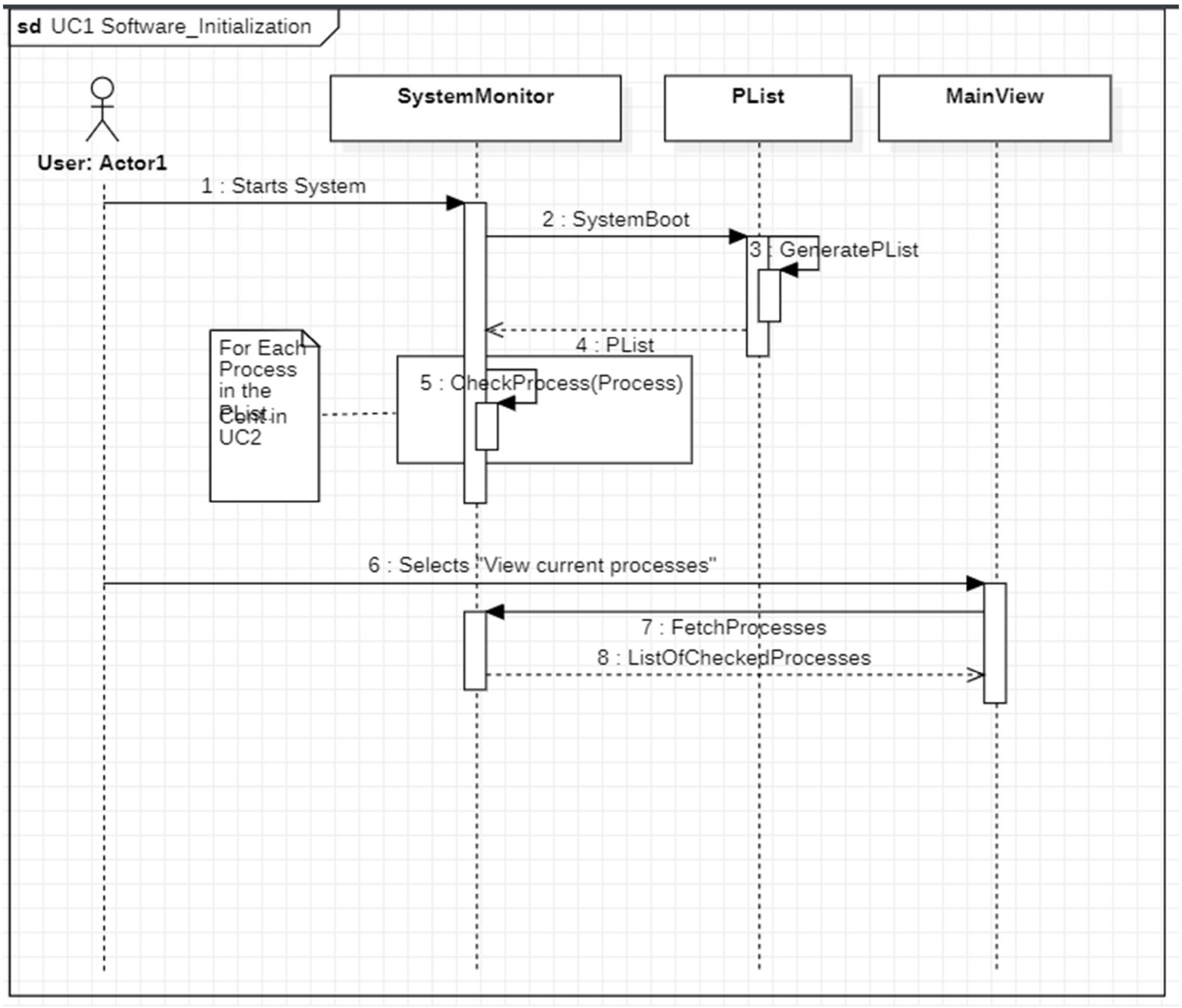
High Level Use Case



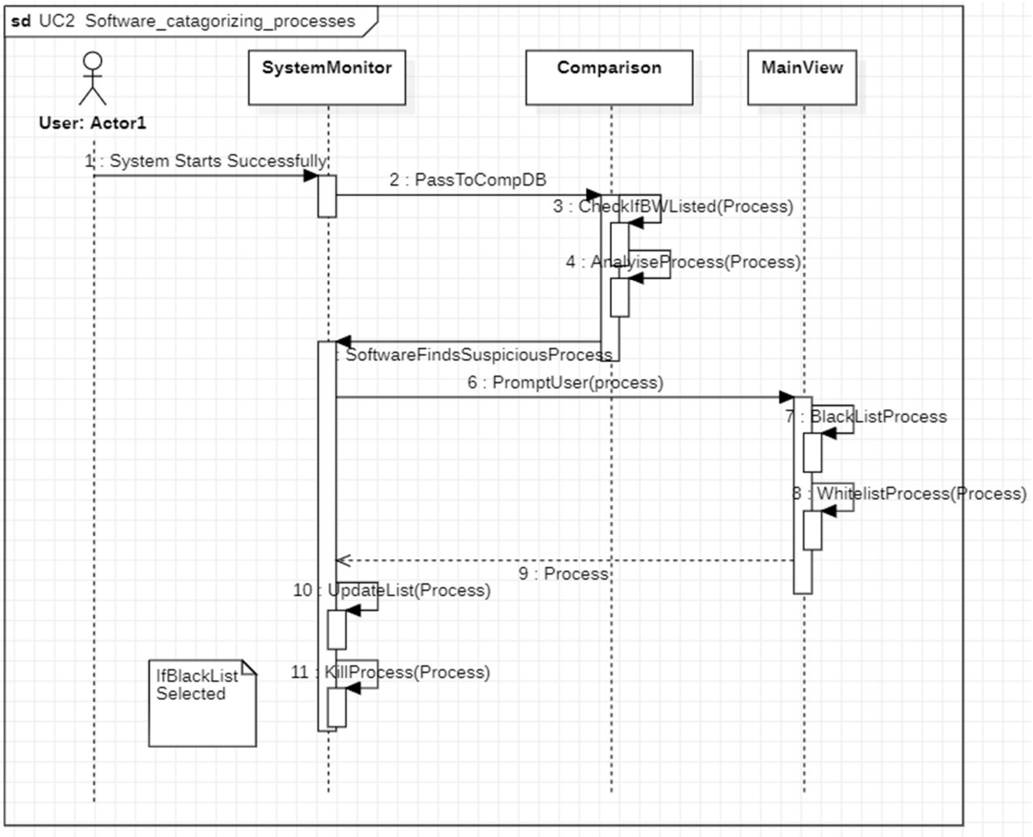


Interaction diagrams for each use case

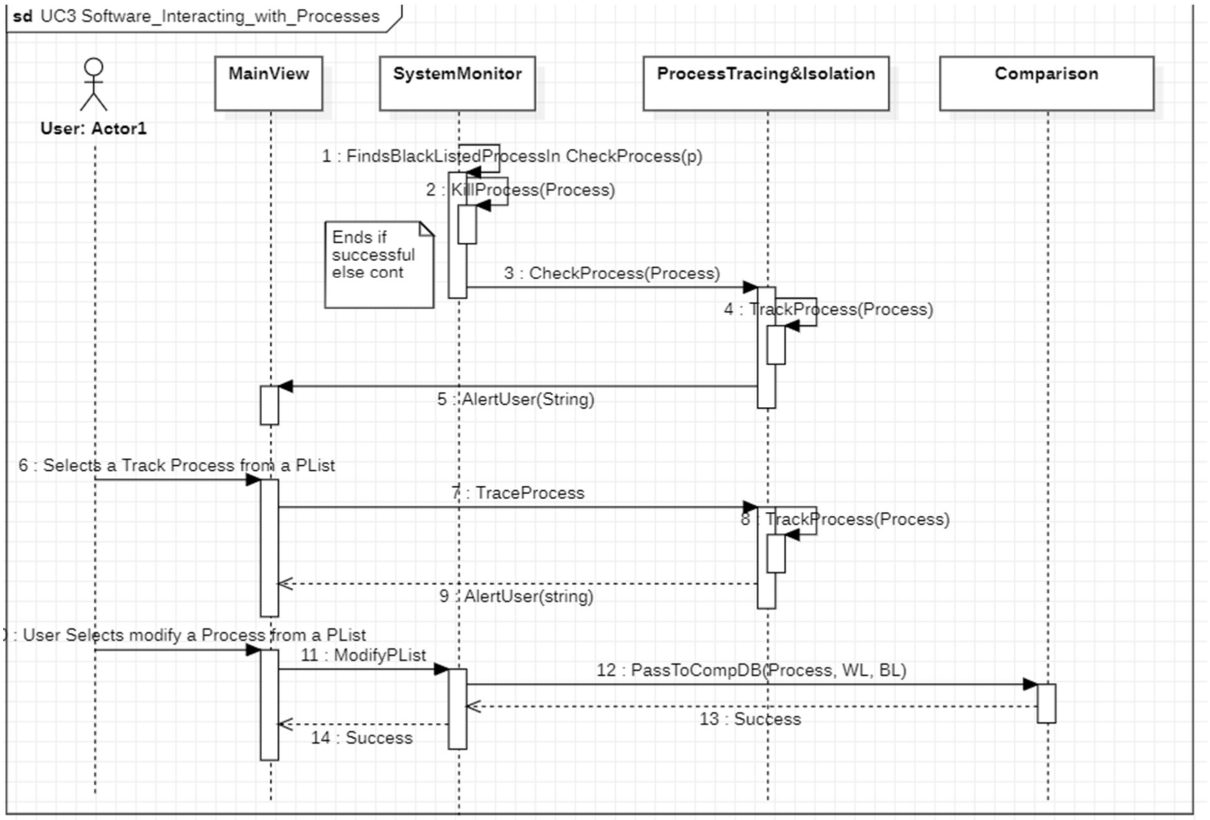
Use Case 1


sd UC1 Software_Initialization
User: Actor1
SystemMonitor
PList
MainView
1 : Starts System
2 : SystemBoot
3 : GeneratePList
4 : PList
For Each Process in the PList
Cent in UC2
5 : CheckProcess(Process)
6 : Selects "View current processes"
7 : FetchProcesses
8 : ListOfCheckedProcesses

Use Case 2


sd UC2 Software_catagorizing_processes
SystemMonitor
Comparison
MainView
User: Actor1
1 : System Starts Successfully
2 : PassToCompDB
3 : CheckIfBWListed(Process)
4 : AnalyiseProcess(Process)
: SoftwareFindsSuspiciousProcess
6 : PromptUser(process)
7 : BlackListProcess
8 : WhitelistProcess(Process)
9 : Process
10 : UpdateList(Process)
11 : KillProcess(Process)
IfBlackList Selected

Use Case 3


sd UC3 Software_Interacting_with_Processes
MainView
SystemMonitor
ProcessTracing&Isolation
Comparison
User: Actor1
1 : FindsBlackListedProcessIn CheckProcess(p)
2 : KillProcess(Process)
Ends if successful else cont
3 : CheckProcess(Process)
4 : TrackProcess(Process)
5 : AlertUser(String)
6 : Selects a Track Process from a PList
7 : TraceProcess
8 : TrackProcess(Process)
9 : AlertUser(string)
) : User Selects modify a Process from a PList
11 : ModifyPList
12 : PassToCompDB(Process, WL, BL)
13 : Success
14 : Success

List of test cases

Test#1

Description	System Check-Up on boot
Requirements Met	UC_1
Criteria	The system is in a clean state. Linked activity monitor showing normal usage of the CPU, GPU and memory.
Method and Outcome	White Box Testing Normal CPU and Memory reading under 10% usage. GPU Usage under 5% - Expected

Test#2

Description	Software Startup
Requirements Met	UC_2, UC_3, UC_4
Criteria	Malware detetcting software starts up. Categorizing processes.
Method and Outcome	White Box Testing Software starts to categorize all the processes under three categories. The categories are “Whitelist” , “Blacklist” and “Unlisted processes” . - Expected

Test#3

Description	Categorizing White and Black listed Program
Requirements Met	UC_6, UC_8, UC_9, UC_13
Criteria	User installs a software with admin approval. That software being verified as “whitelisted” by the anti malware tool. Blocked program, unverified software running in the background and
Method and Outcome	Black Box Testing Anti malware tool categorsizing admin approved software, and waiting for permission to kill or suspend blacklisted softwares from the user. - Expected

Test#4

Description	Categorizing Unlisted or Grey listed Program
Requirements Met	UC_7
Criteria	Anti malware tool being able to categories a program that might be a potential risk to the system. This processes might be a white listed

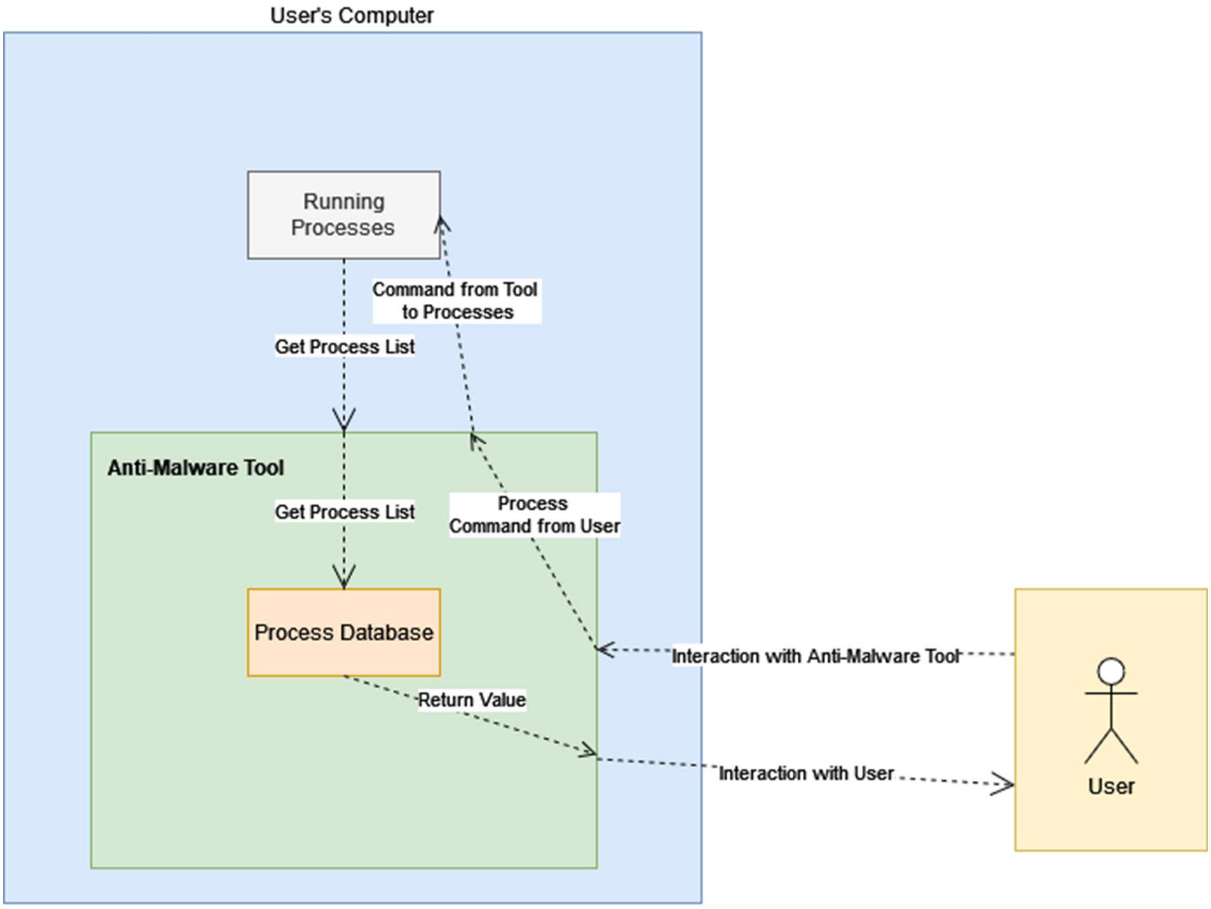
Description	Categorizing Unlisted or Grey listed Program
	program, for example Google Chrome using a lot of memory might fall under grey list.
Method and Outcome	Black Box Testing Anti malware identfies the potential risk to the system. Process that was admin approved but taking up more usage than expected. For example: Google Chrome being installed as an admin, showing unexpected usage of processor, GPU and memory. - Expected

Test#1

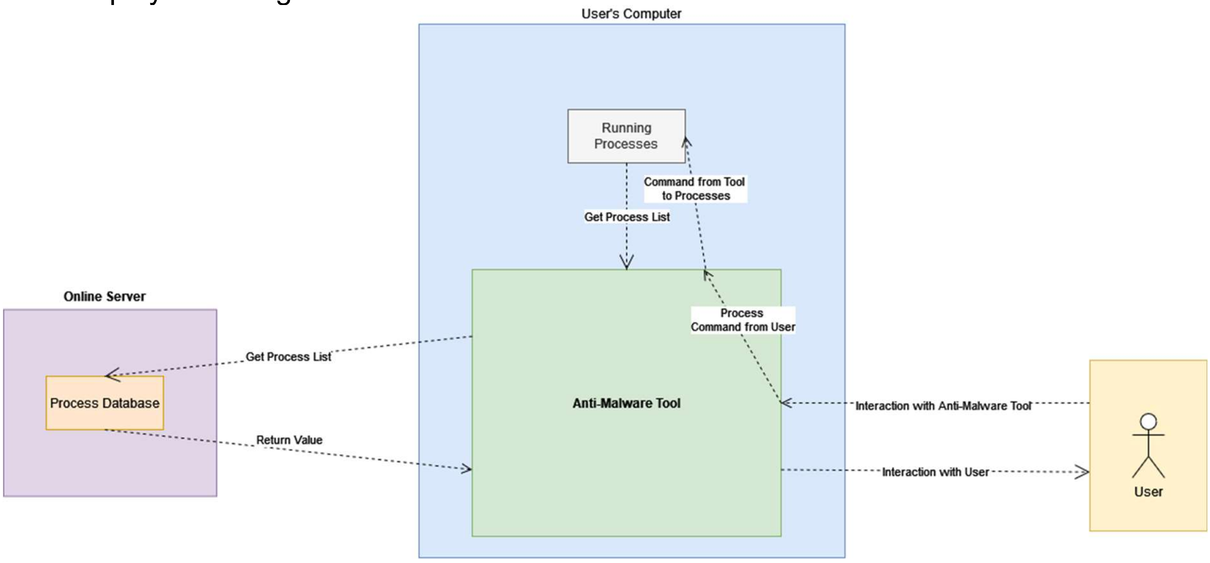
Description	Killing or Suspending a program under user command
Requirements Met	UC_11
Criteria	Anti malware software being able to kill or suspend a software that the user might think is a potential risk to the system.
Method and Outcome	Black Box Testing User gets a promt notification on black listed processes to be suspended. The anti malware software kills a process that is a potential threat to the system. User can also kill white listed programs and unlisted programs as well. But white listed programs does not notify the user about a threat. - Expected

Deployment diagram

Currently planned Deployment Diagram.



NTH Deployment diagram where server is hosted online.



User's Computer
Running Processes
Command from Tool to Processes
Get Process List
Online Server
Process Command from User
Get Process List
Process Database
Anti-Malware Tool
Interaction with Anti-Malware Tool
Return Value
Interaction with User
User