Leveraging Bayes Theorem for Enhanced Automated Red-Team Operations using MITRE Caldera

Reyes Lee Yui Hou[1], Goh Shang Yu[2], Lim Seh Leng[3]

[1]Anglo-Chinese School (Independent), 121 Dover Rd, Singapore 139650
[2]Victoria School, 2 Siglap Link, Singapore 448880
[3]Defence Science and Technology Agency, 1 Depot Rd, Singapore 109679

Abstract

This report explores the application of Bayes Theorem to enhance automated red-team operations, focusing on improving the efficiency and adaptability of adversary simulations using the MITRE Caldera platform. Red-teaming is a critical method for identifying vulnerabilities in computer systems by emulating real-world threats. Caldera's planner decides which attacks to execute and in which order. This research investigates whether a probabilistic, algorithmic planner based on Bayes Theorem can outperform fixed-sequence approaches in red-team emulation.

We conducted controlled experiments using virtual machines, comparing the planners in terms of success rate and stealth, and analysing other output data from the platform. The results show a modest improvement in both areas, with the bayes planner achieving an 88.3% success rate, compared to 84.2% for the atomic planner, and slightly better stealth. Secondary qualitative data suggests that the improvement in performance would have been greater should the planner have been provided with more data and tuned appropriately.

We have found that algorithmic planners have the potential to optimize attack simulations by focusing on high-success actions, reducing unnecessary failures, and improving overall operational efficiency. Our research reveals the importance of adaptive techniques in cybersecurity testing and the potential for further development of intelligent adversary emulation planners.

1 Introduction

1.1 About the Research Topic

Red-teaming refers to a cybersecurity exercise that simulates an adversarial cyberattack to find vulnerabilities and assess the incident response capabilities of a computer system. The red-team attempts to exploit vulnerabilities and test the defensive strategies of the system, aiming to improve its security measures. They use a black-box methodology (of no prior knowledge of the target) to accurately reflect the mindset and tactics of attackers.

Threats can be emulated by mimicking the historical **tactics, techniques, and procedures (TTPs)** of real-world adversaries. The data collected from the simulated attacks can highlight the weaknesses and vulnerabilities discovered, along with recommendations for improvement. This feedback is essential for enhancing the system's security posture.

TTPs are used to model the strategies and behaviour of adversaries. Threat actors use these methods to achieve their goals, such as bypassing authentication and firewalls, stealing data, or disrupting operations.

1. **Tactics**: Overall objectives that an adversary seeks to accomplish during an attack. This could be gaining unauthorized access, escalating privileges, maintaining persistent access to the network, exfiltrating sensitive data, or disrupting the system.
2. **Techniques**: Specific methods used by adversaries to achieve their tactical objectives. For example, to gain initial access, a technique could be exploiting a login vulnerability, or to conduct privilege escalation, a technique might involve abusing access control mechanisms.
3. **Procedures**: Specific, detailed actions that an attacker takes to implement a technique. They are represent the exact implementation of a technique. For example, to exfiltrate data, the procedure could involve a custom script uploading encrypted data using Domain Name System (DNS) tunnelling.

The most widely recognised and used framework in cyber red-teaming is the **MITRE ATT&CK Framework**, which is popular for organising and sharing TTPs. It is a knowledge base that documents the TTPs used by threat actors in various stages of an attack.

MITRE Caldera is a powerful tool for automated adversary emulation, using TTPs in accordance with the MITRE ATT&CK Framework. Users can build a specific threat profile and launch it against target systems, testing defenses and training the blue-team on detection/incident response. The software is modular and extensible, with plugins for customizing attack sequences.

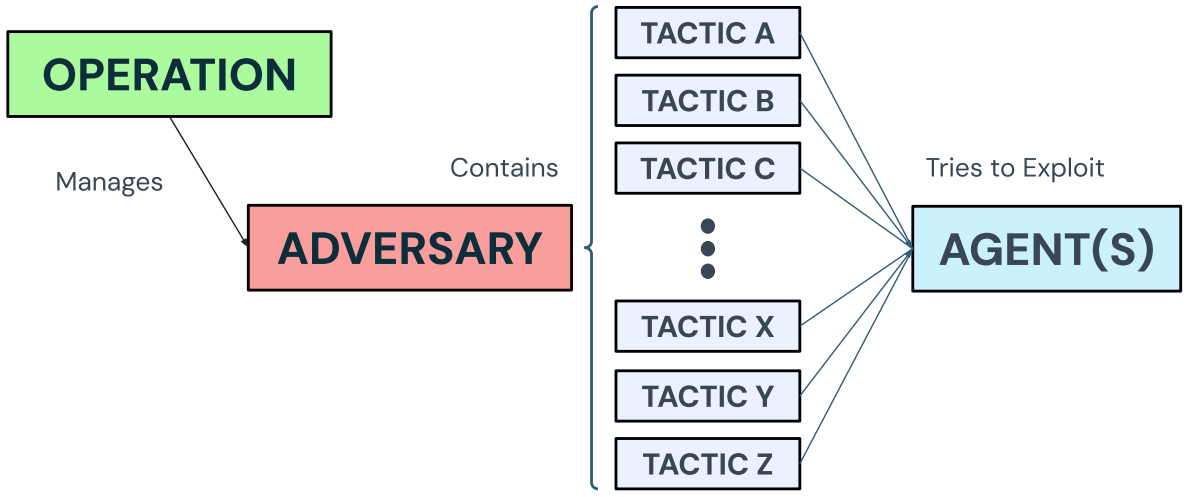

Figure 1: How MITRE Caldera Works (Colours in image are insignificant)

MITRE Caldera uses specialised terminology that will be used throughout this report:

1. **Operation**: Specifically ordered series of abilities from the adversary being run on agent groups.
2. **Adversary**: Predefined group of abilities that models the profile of real-world threat actors to run attack chains.

3. **Ability/Link**: Specific tactic/technique implementation that an adversary can perform on target agents during an attack simulation. It is termed a link when the ability can be run during an operation because its execution requirements have been fulfilled.
4. **Agent**: Software deployed on a target system to execute commands and carry out adversarial actions. Acts as a bridge between the attack platform and the target. Agents can be grouped together to be tested at the same time.

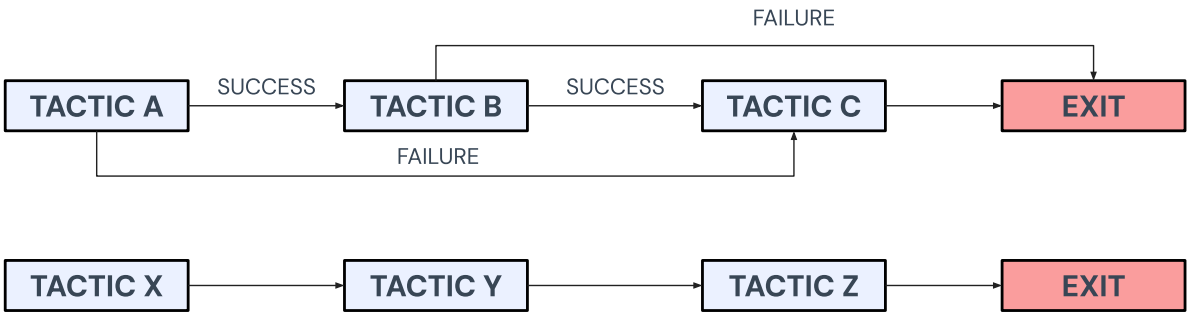

Figure 2: How a Planner may Work

The **planner** chosen at the start of the **operation** contains logic which determines the order in which abilities will be executed. Some common planners are detailed below.

1. **Atomic Planner**: Executes abilities in a strict, predefined order (the adversary's atomic ordering). Links are executed one by one, for each agent. Waits for each ability to finish before moving on to the next one. Mirrors the idea of an adversary moving carefully, performing actions step by step in sequence.
2. **Batch Planner**: Executes all the available abilities from the adversary profile all at once on every agent. This approach has significant drawbacks including inefficient computing resource utilisation, lack of stealth and decision-making.
3. **Buckets Planner**: Executes abilities in the adversary profile grouped by ATT&CK tactic. Abilities in the same bucket are executed all at once, and one bucket must finish before another can begin.

The **Naive Bayes Planner** is a more complex planner that implements Bayes Theorem in an attempt to increase operational efficiency and adaptability. It utilizes past operational history to execute operations while prioritizing likelihood of link success. Hence, it requires the operation to obtain sufficient local data. It will run abilities in order of most to least effective links, while dropping links with insufficient likelihood of success, and could continuously improve decision-making as more operational data becomes available.

Bayes Theorem states the probability P of a hypothesis H given an evidence E is

$$P(H|E) = \frac{P(E|H) \times P(H)}{P(E|H) \times P(H) + P(E|\neg H) \times P(\neg H)} \tag{1}$$

Equation 1 is used in this specific planner to update the expected probability of success for each potential action using each new piece of operational data.

Its algorithm can be described in the below pseudocode:

```
while (available links in the operation for live agents):

  if (link has sufficient operational data and probability of success):
    execute link with highest calculated probability of success

  elif (remaining links with insufficient past data):
    execute next link based on atomic ordering

  finally:
    drop any links with sufficient past data but insufficient probability
of success
```

1.2 Research Aims

1. To investigate the significance of the sequence and manner in which an automated red-team attack is carried out, on its effectiveness.
1. To determine if the application of Bayes Theorem in an automated red-team attack can make the attack more effective.
2. To demonstrate that algorithmic or conditional planners can help penetration testers with their work in automated red-team attacks.

1.3 Research Questions

1. Does the sequence and manner in which an automated red-team attack is carried out affect its effectiveness?
2. To what extent does Bayes Theorem enhance the effectiveness of adversary emulation in cybersecurity testing?
3. How can algorithmic and conditional planners reduce human workload and make automated red-team emulation more effective?

1.4 Rationale for Research

Automated red-team attacks are highly important for penetration testers and organisations looking to secure their computer systems, as it allows them to test their systems for common vulnerabilities and policy errors, saving time and effort.

There are only a few industry-standard red-team emulation tools, such as: Cobalt Strike by Fortra, Caldera by MITRE, Metasploit Framework by Rapid7 and Atomic Red Team by Red Canary. All of these primarily rely on user-driven or batch-style execution of techniques, where a sequence of actions are executed in a predefined order.

These methods, while effective for straightforward scenarios, often lack the ability to dynamically adapt to changing conditions in the target environment. Algorithmic planners, such as those employing probabilistic models, are almost never utilized in these tools. Instead, the standard approach is either sequential execution, where each step is executed in a fixed order, or batch execution, where multiple techniques are run indiscriminately across all systems in scope.

The lack of strategic reasoning limits the effectiveness of emulation, and underscores the need for research into algorithmic planners.

This research will quantify how well an algorithmic planner performs compared to traditional methods. It may help penetration testers and organisations develop more realistic attack simulations, since more efficient adversaries can be emulated. Overall, it may help organizations improve their cybersecurity defenses and create a better security posture.

2 Methodology

2.1 Variables

The **independent variable** is the **planner** used to run the operation.

Table 1: Dependent Variables

Dependent Variable	**Method of Measurement**
Success Rate / %	The percentage of abilities that achieved successful execution without erroring out, timing out, or being denied by the target. It is calculated using Equation 2. $$\frac{\text{number of successful abilities}}{\text{total abilities executed}} \quad (2)$$
Stealth	The total number of detected actions by the anti-virus/ detection software across all target computers. This value is obtained by manually checking the Windows Defender logs after the entire operation is completed, and verifying that the timestamps of the warnings is similar to when that particular attack was executed.

Table 2: Constant Variables

Constant Variable	**Reason**	**Method to Keep Constant**
Target System	Different computer systems have to be exploited and attacked differently. One attack may have completely different mechanics on another operating system.	Windows 10 Home 2022 is chosen as the operating system for the target computer. It is the most common operating system for daily usage and many attacks are designed for it. It is set up in a virtual machine with standardised parameters.
Agent Used	Different agents (reverse shells) may have varying performance characteristics, such as resource utilization or payload compatibility.	The default agent sandcat (golang/HTTP) is used, and the agent is **not** given administrative permissions.

Constant Variable	Reason	Method to Keep Constant
Adversary Profile	The tactics, techniques, and procedures (TTPs) used determine the complexity and scope of the attacks, which must remain consistent for comparison.	A predefined sequence of MITRE ATT&CK techniques is used, covering all tactics.
Defensive Tools	Changing defense mechanisms can lead to biased comparisons, as different tools may detect or block attacks differently.	Windows Defender with default settings is used as the only defensive tool.

2.2 Procedure

1. Oracle VirtualBox is used to setup three virtual machines running Windows 10. They are configured as home-use computers.
   - Apps are installed, browsers are setup with history and passwords.
   - Sample files of different types are planted in various directory locations.
   - All software is checked to be running the same version.
2. Windows Defender is set to whitelist the agent. This is important as Windows Defender would otherwise flag the agent as malicious by default.
3. A snapshot of each virtual machine is taken after setup is completed to ensure the same starting parameters.
4. An agent is launched on every virtual machine.
5. From the Caldera web console, an operation using the custom adversary profile is launched on every agent, with the atomic planner managing the operation.
6. The total number of abilities executed, the number of successful abilities and the number of antivirus warnings logged is recorded and tabulated. The **planner's logs**, which detail the planner's decision making, are also collected from Caldera.
7. Each virtual machine is reset to the original snapshot.
8. Step 5-7 is repeated for a total of three iterations.
9. Step 5-8 is repeated with the bayes planner.

3. Results and Analysis

3.1 Results

Table 3: Quantitative Results

Dependent Variable	Atomic Planner	Bayes Planner
Average Success Rate / %	$\frac{33+35+33}{3\times40} = 84.2\%$	$\frac{34+36+36}{3\times40} = 88.3\%$
Stealth / Total No. of Warnings	$6 + 8 + 5 = 19$	$7 + 6 + 5 = 18$

3.2 Analysis

The bayes planner has a slightly higher percentage of success at 88.3% compared to that of the atomic planner at 84.2%. This increase in success rate can be attributed to the bayes planner making decisions to avoid executing certain abilities that had too low a chance of success, as determined by past operational data.

For example, if a particular ability has failed to execute on every target system so far, it is likely that the next attempt on another system will fail again. Hence, the bayes planner can make the decision to skip this ability, saving time and resources, allowing it to focus on the attacks that work.

In contrast, since the atomic planner is programmed to execute every ability, it will still execute an attack even if it has a significant known history of failure. This wastes time and resources when running an automated attack; Penetration testers could be forced to sift through many irrelevant failures in the operation report.

There is also a slight improvement in stealth for the bayes planner, with 18 warnings generated compared to 19 for the atomic planner. This marginal improvement can be attributed to the bayes planner's ability to adapt its behavior based on past detection patterns. For instance, if certain abilities have frequently triggered antivirus system alerts in prior attempts, the bayes planner skips those abilities, improving its stealth profile.

3.3 Discussion

The results collected can be further explained and visualised by secondary debugging data collected during the testing.

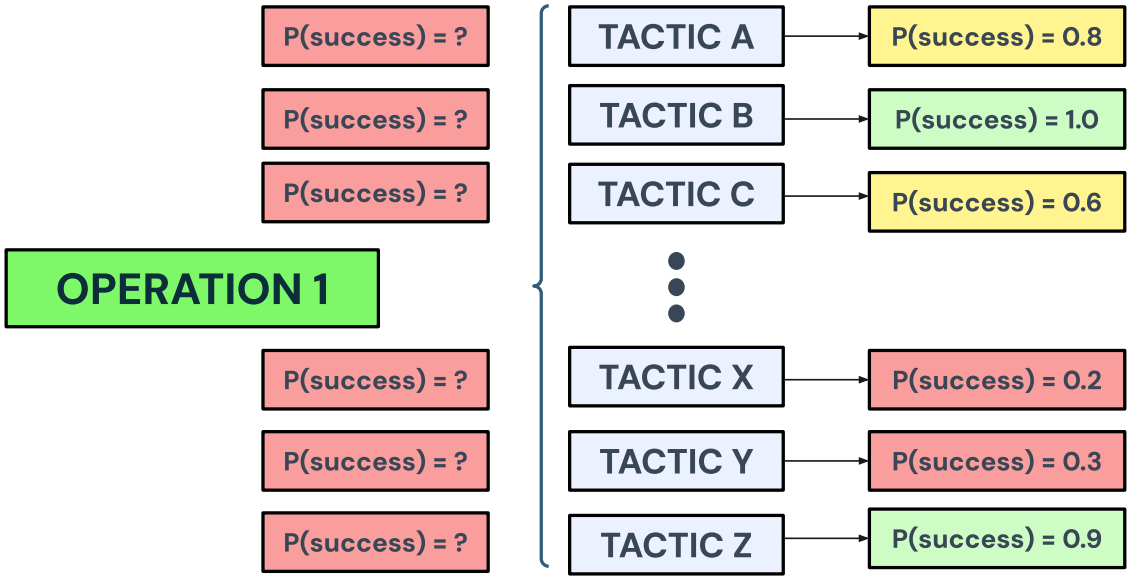

Figure 3: Simplified Visualisation of Bayes Planner Initial Usage

In Figure 3, the bayes planner defaults to an atomic ordering when no past operational data is available, going from Tactic A to Tactic Z. As the operation completes, the success rate is measured and recorded. In other words, the planner mathematically learns which abilities have the highest prospects of success.

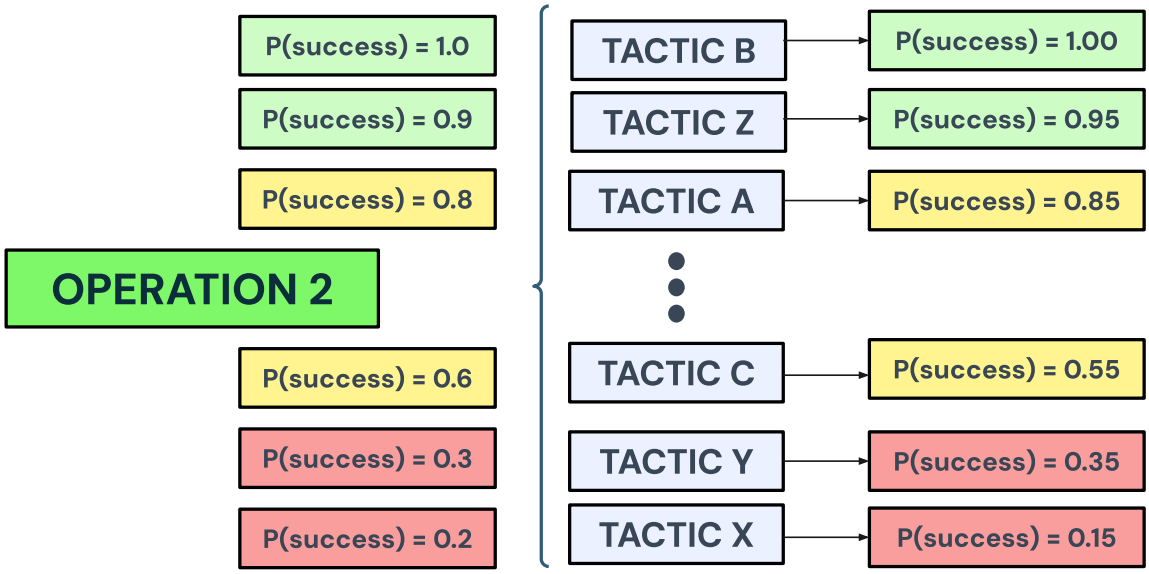

Figure 4: Simplified Visualisation of Bayes Planner Usage on a Second Operation

Figure 4 then shows how the planner prioritises tactics that are most likely to succeed, based on the historical success rate collected in the previous operations. Instead of sending Tactics A-Z, it begins with Tactic B because it has the highest success rate, and proceeds in descending order of predicted effectiveness.

Depending on the minimum probability of success threshold set by the executing user, Tactic Y and X may not be executed. For example, if that threshold is set to 0.50, $P_y(\text{success})$ and $P_x(\text{success})$ will be insufficient and they will be skipped.

The increasing precision of $P(\text{success})$ between Figure 3 and Figure 4 also highlights the planner's improved decision-making capabilities as it gathers more operational data. This is somewhat like a positive feedback loop, a system that continually refines itself as more data becomes available. It reflects a refined estimation process, enabling more accurate predictions about the success of future tactics based on prior performance.

Hence, the conditions required to maximise operational efficiency and success rate are a **large amount of historical data** and an **optimised minimum success threshold setting**.

3.4 Evaluation

A strength of this investigation is the use of quantitative parameters such as success rate and stealth to analyse the differences between the planners. Together with qualitative debugging data collected from Caldera, it ensures an objective basis for comparison.

Another strength of this work is the use of snapshots of the virtual machines before running each test. This approach ensures that every test run starts from a consistent system state, eliminating variability caused by leftover processes, altered system/network configurations, or residual malware from previous runs.

A weakness of this investigation is that the scale of the operation is too small — the numerical gaps between the planners in success rate and stealth are relatively small. Drawing a definitive conclusion about the superiority of one planner over the other is challenging, as the limited number of abilities and target systems did not allow significant performance variation. An improvement would be to expand the experiment to a larger scale — by using more abilities and targeting a broader range of systems, the bayes planner might produce much more noticeable advantages.

Another weakness is the single target environment. Testing was conducted on a specific Windows OS setup for ease of setup, potentially reducing the applicability of results to other environments. The improvement would be to conduct tests across various operating systems (e.g. MacOS, Linux), network configurations (e.g. Ethernet, WiFi), file system setups, running varied software and services, to ensure broader applicability.

For a deeper understanding of the bayes planner, further work should attempt to quantify how each generation of bayesian learning by the statistical model affects the performance of the planner. Further work can also compare the bayes planner with other statistical models that involve risk-reward calculations. Finally, multi-stage adversaries and conditional planners can be tested as they may produce more varied results.

3.5 Conclusion

The experimental findings conclude that the sequence and manner in which an automated red-team attack is executed is highly important. This addresses the first research question; execution strategy impacts effectiveness, even if the size of that impact depends on the scale of the operation.

The bayes planner's ability to adjust its execution based on historical data demonstrates the potential for applying Bayes Theorem to enhance adversary emulation. It can achieve a higher success rate and better stealth, meaning that adaptive planners can optimize attack performance, addressing the second research question.

Optimal operational efficiency of the bayes planner is achieved when a large amount of historical operational data is available and the parameters of the planner are optimally tuned.

Finally, the bayes planner's data-driven decision-making can streamline operations by helping testers identify attacks with the highest probability of success, conserving resources, reducing unnecessary alerts, hence improving blue-team preparedness through more efficient adversary emulation.

3.6 Real-World Applications

A practical application of this research could be when testing large numbers of Windows systems for Living Off the Land Binaries and Scripts (LOLBAS/LOLBins). LOLBAS are trusted binaries or scripts native to the system, that attackers can abuse to circumvent security measures and carry out malicious actions. Given the extensive list of potential LOLBin vulnerabilities, it is extremely inefficient and resource-intensive for a penetration tester to attempt each LOLBin on every target system.

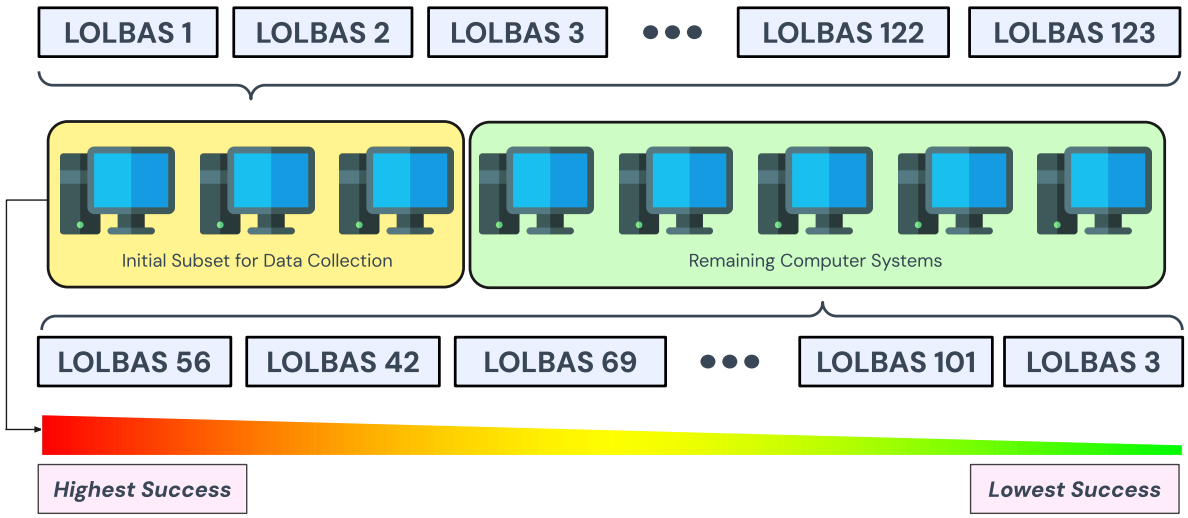

Figure 5: How the Bayes Planner can Streamline LOLBAS Testing

Using the bayes planner, the penetration tester could first test a small subset of systems to gather some initial data on which LOLBins succeed or fail. If certain LOLBins consistently trigger antivirus alerts, fail to execute due to system restrictions, or error out, the planner will simply deprioritise or skip these binaries in future operations. Conversely, LOLBins with a high success rate can be targeted more aggressively across the remaining systems.

This approach reduces time wasted on ineffective binaries and allows the tester to focus on high-impact techniques, improving testing coverage in a shorter timespan. The bayes planner condenses the penetration testing process, enabling quicker recognition and remediation of exploitable LOLBins.

In conclusion, the bayes planner can **increase the productivity of cybersecurity testers** by making it much easier to **pinpoint the best starting point of attacks**.

4. Acknowledgements and Bibliography

4.1 Acknowledgements

Special thanks to Ms Lim Seh Leng for her mentorship and guidance throughout the project.

4.2 Bibliography

1. MITRE Caldera Source Code (2024). https://github.com/mitre/caldera
2. MITRE Stockpile Source Code (2024). https://github.com/mitre/stockpile/
3. MITRE Caldera Homepage (2024). https://caldera.mitre.org/
4. MITRE Caldera Documentation (2024). https://caldera.readthedocs.io/
5. MITRE ATT&CK Framework Documentation (2024). https://attack.mitre.org/
6. MITRE Caldera Bayes Planner (2023). https://medium.com/@mitrecaldera/mitre-caldera-naive-bayes-planner-1a581c2140c3
7. LOLBAS Project (2024). https://lolbas-project.github.io/
8. Windows 10 Download (2022). https://www.microsoft.com/en-us/software-download/windows10ISO

5. Appendix

5.1 List of Software Versions

1. MITRE Caldera: v5.0.0 “Magma”
2. Oracle VirtualBox: 2023 Version 7.0.10 r158379 (Qt5.15.2)
3. Windows 10 x64: Version 22H2 (OS Build 19045.3803)
4. PowerShell: 5.1.19041.3803
5. Windows Defender Antimalware Client: 4.18.24090.11
6. Windows Defender Engine: 1.1.24090.11
7. Windows Defender Antivirus: 1.421.725.0
8. Windows Defender Antispyware: 1.421.725.0