Evolution of Training Methods and Skill Sets in Modern Intelligence Operations

CS4991 Capstone Report, 2024

Aaryan Dhore
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
knc8xp@virginia.edu

ABSTRACT
My project involves creating a database schema that adheres to the Single Source of Truth principle for a government contracting company. We achieved data consistency across independent cores, each with its database through a Continuous Integration/ Continuous Deployment (CI/CD) pipeline. With the designed and implemented schema, developers can add, edit, or delete applications, with changes seamlessly propagated to all cores. A collaborative feature enables individual cores to modify application descriptions, fostering communication and improvement exchange. Results show successful SSOT resolution, efficient data management and enhanced maintainability. Future work involves refining the pipeline and doing more rigorous testing.

1. INTRODUCTION

Espionage is a practice as ancient as human civilization itself. Throughout history, methods for gathering private information covertly have been employed, with records dating back to the Roman Empire's use of ciphers and double agents for military intelligence (Encyclopedia.com, n.d.) Traditionally, espionage was a purely human endeavor. However, the emergence of the internet and digital technology has ushered in a new era.

In modern espionage there are many different types of tradecrafts—skills or techniques used to carry out missions. Technical tradecraft refers to the methods to disguise technologies used (Intelligence Academy, n.d.). For example, if an agent is trying to communicate with targets in Mexico, but the agent's computer is in Eastern Standard Time and there is no browser history, the authenticity of the communication degrades in the eyes of the target. Technical tradecraft has gained prominence due to the growing reliance on online communication for missions, especially for evidence gathering or luring targets into traps.

Technical tradecraft has become more prevalent over the past few decades due to rapid advancements in internet and communication tools making it easier for agencies to conduct remote surveillance, cyber espionage, and communication. These types of missions are more cost effective and efficient as agents can cover a broad range of targets over extended periods of time without worrying about the logistical or safety challenges of physically deploying agents (Dehlinger, 2020). Given the increasing reliance on technical tradecraft, how has that affected the training and skill sets required for intelligence operatives, and how do intelligence agencies adapt to these changing demands?

To address this, my previous internship project centered on maintaining technical tradecraft to make computer/network connections appear as genuine as possible to the target. Unlike other technologies, our solution allowed operatives to customize their setup to suit specific missions, selecting applications, browser history, keyboards, and more. Additionally, our solution could generate virtual machines (VMs) with desired configurations instantly. This software-based approach saved agencies money and enhanced security, eliminating the need for frequent purchases of disposable "burner" laptops. In contrast, other technologies lacked this customizability, providing blank slates that required time-consuming setup tailored to the target/mission.

2. RELATED WORKS

Criminals often use the Tor browser as a tool for their activities and communications. The Tor browser works by routing internet traffic through a network of nodes. Each node encrypts and relays data, which hides the user's IP and makes it challenging for anyone monitoring the network to trace the origin or destination of the data. This layered encryption ensures that individuals engaging in illicit activities, such as trafficking illegal goods, coordinating cyberattacks, or communicating sensitive information, can operate with reduced fear of detection (Jadoon, et al., 2019). Since many criminals conduct their affairs on the Tor browser, law enforcement may have their own agents go undercover within the network to try and extract information. Since the anonymity goes both directions, the criminals would have a difficult time tracing the communication to law enforcement.

However, the Tor browser is not impenetrable; many attacks can be used to remove the anonymity of a connection. One such attack is traffic analysis, where adversaries monitor network traffic patterns to correlate communication endpoints, potentially revealing the origin of data. Additionally, timing attacks exploit the latency introduced by Tor's routing, allowing adversaries to deduce connections between users and hidden services. Furthermore, end-to-end correlation attacks aim to link entry and exit points of data packets, compromising anonymity (Karunanayake, et al., 2021). Due to vulnerabilities in the Tor browser, many criminals try to layer security by using Tor on VMs or burner devices. Virtual machines allow for isolated environments where Tor can be run, reducing the risk of compromising the host system and minimizing the footprint of their activities (Fassl, et al., 2023). Similarly, burner devices, such as cheap smartphones or laptops, provide a temporary platform for accessing Tor, which can be easily discarded to avoid detection.

3. PROJECT DESIGN

The project aimed to create a customizable solution for operatives to establish authentic computer connections with targets. It offered personalized setups and rapid virtual machine generation, saving costs and boosting security over disposable laptops. Users could customize their virtual machines (VMs) with specific software, history, settings, and operating systems. My main task was to improve data consistency in the existing application.

The application was deployed on multiple independent cores, with each core containing a JSON file listing configurable applications, software, and settings for VMs. Data consistency issues arose due to overlaps and repetitive information across the cores. Simple changes, such as adding software configurations, were cumbersome and time-consuming. My task involved migrating data from JSON to a well-structured database for easier CRUD operations.

```
‘software’: {
   ‘communication’ : {
        ‘Discord’: {
           //attributes of Discord
        }
   },
   ‘file sharing’ : {
        ‘Discord’: {
           //attributes of Discord
        }
   }
}
```

Figure 1: Sample JSON Data Format

The data schema was designed with separate tables for each configurable option, simplifying data management. However, there are issues when it comes to more complex relationships. For example, in the above Figure we see that Discord is a software that can be used for communication and file sharing. Therefore, it was present within both categories. This means that a specific program can be in multiple categories, and each category can have multiple programs. This ‘many-to-many’ relationship was solved by using a bridge table. A bridge table is a seperate table that handles the relationships between the categories and programs. It’s only purpose is to keep track of links between them and is a common solution to many-to-many relationships.

After this schema was finalized and approved, I created a python script to automatically create and populate the tables. The script worked by parsing the JSON and iteratively populating each table.

The project's second portion was focused on maintaining data consistency across the cores. When developers added a new item to the database, it had to be propagated to all cores to maintain uniformity. Initially, a master-slave configuration was considered, but it was deemed too risky due to security concerns. Establishing direct links between the central database and the cores posed potential vulnerabilities. Instead, we implemented a CI/CD pipeline to manage updates. When changes occurred, the pipeline was triggered and executed updates across cores using secure shell (SSH). This approach avoided direct connections and allowed targeted, secure updates without compromising security.

The CI/CD pipeline was structured into three stages: analyzing changes, establishing SSH connections, and executing updates. The analysis stage used a Python script to identify changes in the database and write them to a text file. To access the cores, we set up passwordless SSH using public-private key pairs for secure and seamless connections. Once connected to a core, another Python script read the text file and made the specified changes directly to the core's database. This method ensured efficient and secure propagation of updates across all cores.

4. RESULTS

For these projects there were not many quantifiable metrics to judge success. Regardless, the duplicate data was removed. Of 2K records in the original JSON, over 150 were duplicate records. As the data is normalized in the database, the UI was alterred so that so that users could change descriptions within the web application itself. Previously, they would be forced to go into the backend and manually change the JSON file. Furthermore, the code to show the data in the front end was made a lot simpler and less confusing.

The execution time of the pipeline was instantaneous, as it got triggered by a commit from the JSON file. Each stage occurred within 100ms except the SSH portion. That makes sense, however, as network protocols

slow down the SSH process. During the internship, the pipeline was only deployed to make changes on a development/test server. However, afterwards we were informed that the pipeline was deployed to all the active cores.

CONCLUSION

This project significantly enhanced the ability to maintain data consistency and efficiently manage complex data across multiple independent cores. The implementation of a structured database schema, which adheres to the Single Source of Truth (SSOT) principle, addressed the previous challenges associated with JSON data overlaps and redundancies. This improvement not only streamlined the addition, editing, and deletion of applications, but it also facilitated the seamless propagation of changes across all cores, ensuring data uniformity and accuracy.

The CI/CD pipeline played a key role in achieving these outcomes by enabling secure and targeted updates without direct links between the central database and the cores. This method mitigated security risks while fostering better data management. The ability of individual cores to modify application descriptions within the web application encourages collaboration and information exchange, which can lead to ongoing improvements.

The results demonstrate the project's success in resolving SSOT issues, enhancing data maintainability, and creating a more efficient system. The experience gained from this project can guide future efforts in refining the pipeline and conducting more rigorous testing to ensure sustained data consistency and optimal system performance. This project provides meaningful advancements in the field of technical tradecraft and offers valuable insights for similar initiatives within the intelligence and government contracting sectors.

5. FUTURE WORK

In the end the pipeline, while worked relatively fast and reliably, was not designed optimally. Running multiple python scripts locally and within the foreign core is a slow process. Furthermore, if I had more time I would have also added a feature to the pipeline to ensure that if a user made an edit, that the developers could see that change.

Furthermore, I would conduct more comprehensive and rigourous testing on the pipeline and the database schema. I would want to build an automatic testing suite, unit test integration, and end-to-end testing.

Lastly, I would want to make setting up the pipeline easier. As of right now, if a new core gets deployed then a developer would manually have to set up passwordless SSH and include the key within the pipeline. However, this can be tedious, so I would design a bash script to automatically run the commands and complete the set up pipeline to work with the new core as well.

6. ACKNOWLEDGMENTS

No acknowledgements.

REFERENCES

Dehlinger, K. W. (2020). Espionage in the Digital Age: How Technology is Impacting the Recruitment and Handling of Spies. https://doi.org/10.26153/tsw/11267

Encyclopedia.com. (n.d.). Espionage and Intelligence, Early Historical Foundations https://www.encyclopedia.com/politics/encyclopedias-almanacs-transcripts-and-maps/espionage-and-intelligence-early-historical-foundations

Fassl, M., Ponticello, A., Dabrowski, A., & Krombholz, K. (2023). Investigating Security Folklore: A Case Study on the Tor over VPN Phenomenon. Proceedings of the ACM on Human-computer Interaction, 7(CSCW2), 1–26. https://doi.org/10.1145/3610193

Intelligence Academy: Getting professional tradecraft training. (n.d.). https://academy.ieis.eu/mod/page/view.php?id=168#:~:text=Generally%2C%20it%20simply%20refers%20to,used%20by%20human%20intelligence%20operators.

Karunanayake, I., Ahmed, N., Malaney, R., Islam, R. and Jha, S. (2021). "De-Anonymisation Attacks on Tor: A Survey," in IEEE Communications Surveys & Tutorials, vol. 23, no. 4, pp. 2324-2350, Fourthquarter 2021, doi: 10.1109/COMST.2021.3093615

Jadoon, A., Iqbal, W., Amjad, M., Afzal, H., & Bangash, Y. (2019). Forensic analysis of TOR Browser: A case study for privacy and anonymity on the web. Forensic Science International, 299, 59–73. https://doi.org/10.1016/j.forsciint.2019.03.030

(Assignment 4—full FIRST DRAFT of report—due April 12.)

INSTRUCTOR NOTE: More detailed instructions for each assignment and element of the Capstone report can be found in the *CS4991 Writing Guide,* posted within Canvas.

(Assignment 5—full FINAL REPORT—due May 1 for PASS/FAIL grading.)