Homework 6: Simulating the British Airways Breach

Craig Troop

The George Washington University

SEAS 8405: Cybersecurity Architectures

Dr. Ravi Mallarapu

May 07, 2025

Table of Contents

1. Introduction 3
2. Steps Taken.. 3
3. Vulnerability.. 5
4. Mitigations 5
5. Insights 5
6. References.. 6

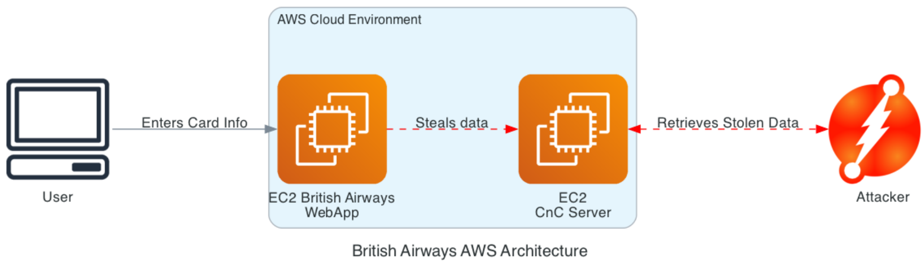

Figure 1 – Attack Demonstration System Architecture

1. Introduction

This report explores the British Airways (BA) data breach involving an attacker ex-filtrating sensitive customer data from what should have been protected S3 buckets. The implemented attack demonstration is a simplified version of this attack using embedded JavaScript and a CNC server.

2. Steps Taken

The first step to demonstrate this attack is to instantiate a Flask web application acting as the CNC server to capture stolen data. This application is hosted on an AWS EC2 t2.micro instance built from the standard Ubuntu AMI. The application is deployed with a Web Server Gateway Interface (WSGI) wrapper and Gunicorn on port 8001. Nginx listens on port 80 and forwards traffic destined for the /exfiltrate endpoint to the Flask application.

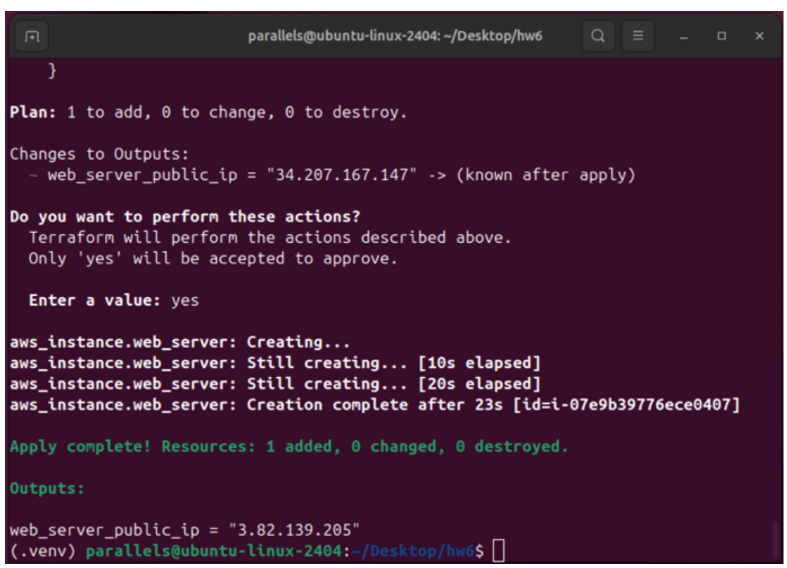

Figure 2 - Terraform script execution

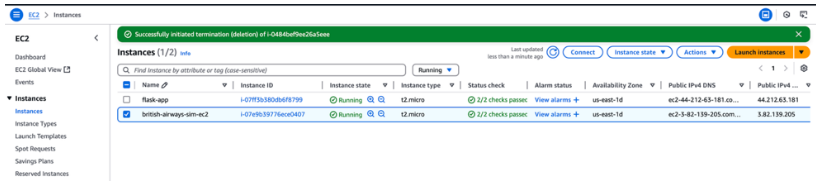

Figure 3 - EC2 Instances Running

Once the CNC server is listening, the next step is to ensure the terraform script contains the correct IP address for the CNC server, and then deploy it to AWS as seen in Figures 2 and 3 above. This deployment requires a terraform user with certificate for authentication and the appropriate least-privilege permissions (AdministratorAccess). The script then deploys a second EC2 t2.micro instance hosting the credit card submission form with the malicious payload as seen in Figure 4 below.

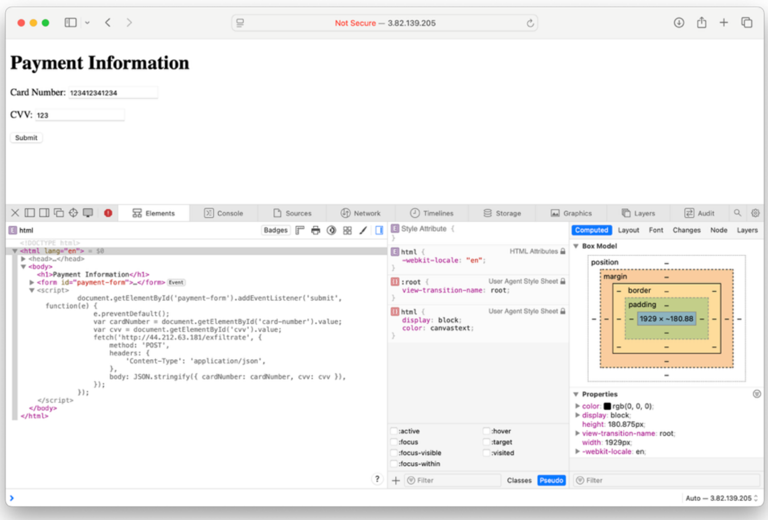

Figure 4 - Malicious Payload on Card Entry Page

When both servers are running, a user can browse to the website and enter their card information. The card information is then sent over to the CNC server and logged for the attacker to retrieve as shown in Figures 5 and 6 below.

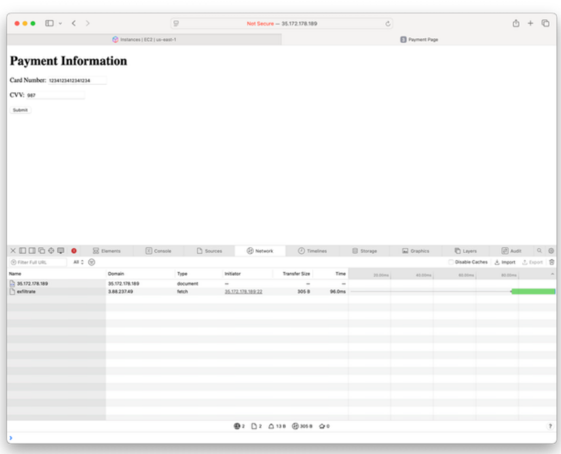

Figure 5 - Javascript execution

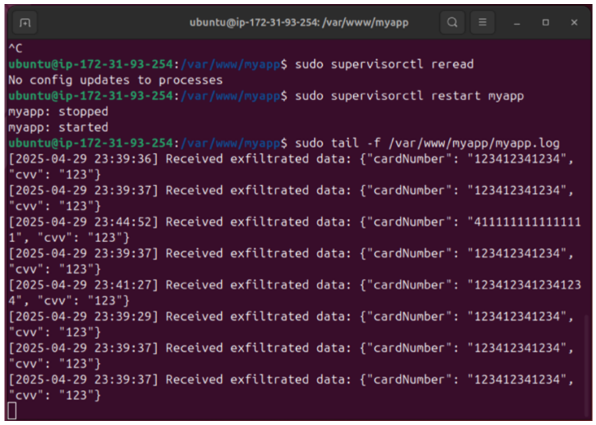

Figure 6 - CNC Server logging card information

3. Vulnerability

The vulnerability demonstrated with the malicious JavaScript payload is an example of a T1189 Drive-by Compromise where an attacker is able to inject malicious code into a web application (Angiolelli et al., 2025). We deployed this payload with a terraform script, however an attacker would most likely inject the malicious payload into the source code directly. In the BA attack, malicious actors gained access to the page source for the BA website and injected a malicious script in this way. Further complicating the vulnerability is the prevalent use of JavaScript in modern web interfaces. BA did not have any additional mitigations in place to prevent the loss of data in this way.

4. Mitigations

Using a Defense in Depth architecture, the perimeter layer control would be Web Proxy Filtering. This control employs an appliance to block known malicious URLs through secure web gateways, a web proxy, or DNS filtering. This control assumes the attacker's CNC server has a known malicious IP address.

A browser/application layer mitigation technique is the use of a Content Security Policy (CSP) (Mallarapu, 2025). A CSP requires the server to sign any scripts with a nonce that is validation when the scripts are loaded. If the script is not signed, the web server will block it. This restricts script sources and can prevent third-party code from executing on a company's website without their permission.

These mitigations should be augmented with a monitoring and response layer control like SIEM. A SIEM solution can rapidly identify the exfiltration of data from the web server and alert the cybersecurity response team to anomalous activity. If combined with a SOAR solution, these alerts could trigger immediate response from the network devices to prevent the data exfiltration and block the attacker's IP address.

5. Insights

I did not have previous experience exposing a Flask application to the internet, so working through the WSGI, Gunicorn, and nginx configurations took some research. Creating the IAM role and configuring the certificates for authentication was straightforward. I discovered nginx was actually caching the post data and when I fixed a misconfiguration, several credit cards came through all at once. I have previously explored injection attacks on Hack-the-Box, so having the malicious payload already present was much simpler than some attack vectors I've seen. Although if a malicious actor was able to modify the source, as we did with the terraform script, the danger is the same.

When researching the BA attack, it was interesting to see the trivial exploits involved that enabled pivots to more sensitive infrastructure. Plaintext administrator passwords stored where the attacker could find them is an incredible oversight. The lack of robust authentication mechanisms for third-party providers was also a contributing factor (Aljaidi, 2023). I think the biggest lesson is that no single event led to this compromise, but rather a combination of several smaller oversights and misconfigurations that compounded to enable and significantly increase the impact of the breach.

6. References

Aljaidi, M. (2023). A Comprehensive Technical Analysis of URL Redirect Attacks: A Case Study of British Airways Data Breach. In *2023 24th International Arab Conference on Information Technology (ACIT)* (pp. 1–5). https://doi.org/10.1109/acit58888.2023.10453784

Angiolelli, F., Sakowicz, J., & Agrawal, S. (2025, April 15). *Drive-by compromise, technique T1189 - Enterprise*. MITRE ATT&CK®. https://attack.mitre.org/techniques/T1189/

Mallarapu, R. (2025). *Week-6: Cloud and Web Security Architecture* [Electronic]. The George Washington University.