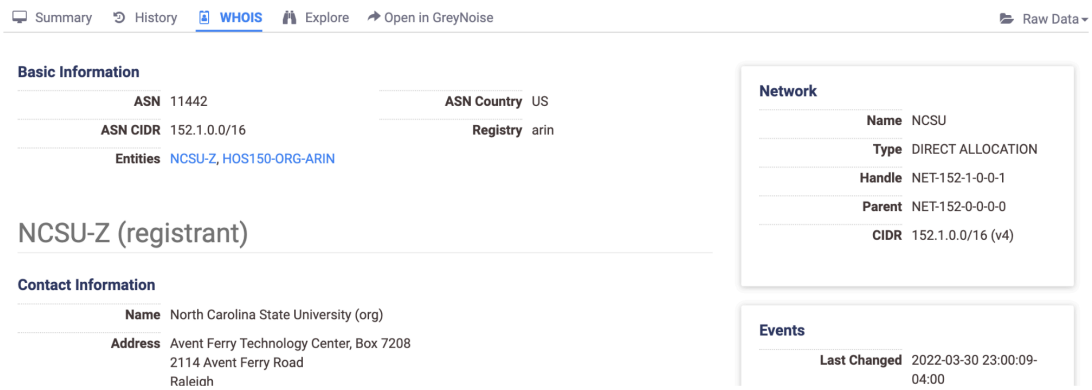
CNS Mini Project 2

Arohan Ajit 200538209 aajit@ncsu.edu

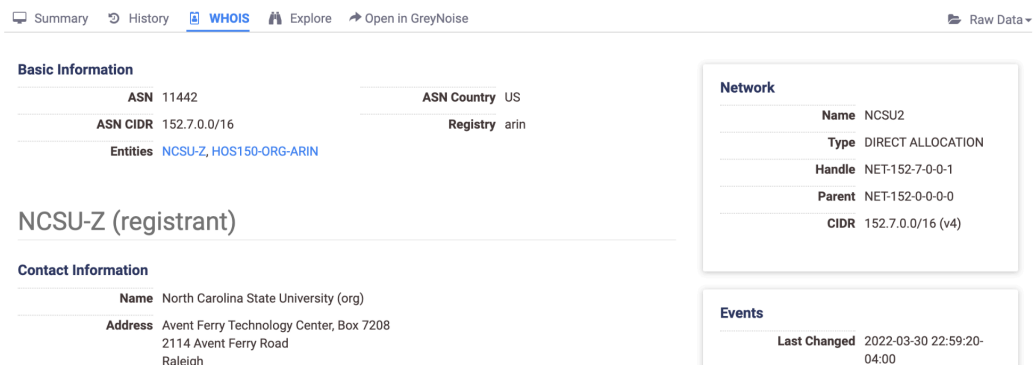
Part 1 – Identifying Unique IPv4 Blocks

Steps Taken

- Install censys api on python
- Search for ncsu.edu domain and save it to json file using `censys search ncsu.edu --pages -1 -o <filename>.json`
- Among various IPs found, look at the WHOIS details for each IP using command `censys view <ip>`
- Details found under keys - ASN number -> ["autonomus_system]["asn"], ASN name -> ["autonomus_system]["name"], CIDR Block -> ["autonomus_system]["bgp_prefix"], Network Name -> ["network"]["name"]
- Network 1
    - **IP:** 152.1.8.157
    - **ASN:** 11442
    - **CIDR:** 152.1.0.0/16
    - **ASN Name:** NCSU
    - **Network Name:** North Carolina State University
    - 

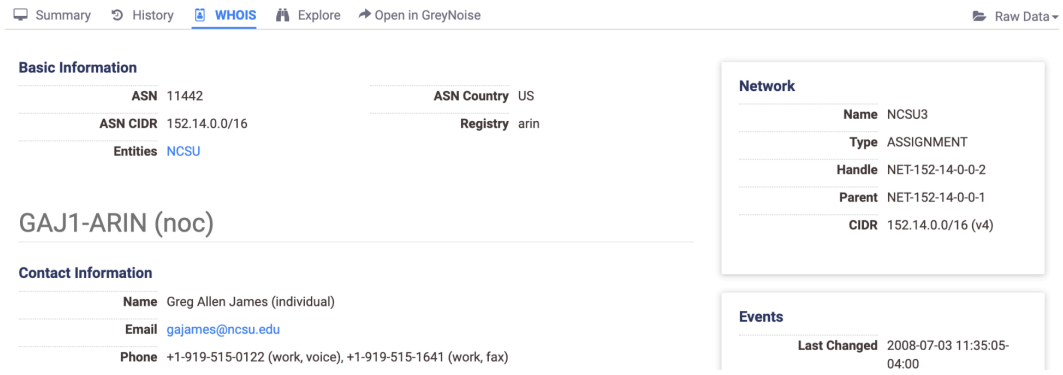
- Network 2
    - **IP:** 152.7.114.146
    - **ASN:** 11442
    - **CIDR:** 152.7.0.0/16
    - **ASN Name:** NCSU
    - **Network Name:** North Carolina State University

152.7.114.146



- Network 3
  - **IP:** 152.14.93.47
  - **ASN:** 11442
  - **CIDR:** 152.14.0.0/16
  - **ASN Name:** NCSU
  - **Network Name:** North Carolina State University

152.14.93.47



Part 2: Network Summary

Code to plot bar graphs for each category:

```
from censys.search import SearchClient
import matplotlib.pyplot as plt


def plot_bar_charts(data):
```

```
    """Plots bar charts for each data point in a row."""

    fig, axes = plt.subplots(1, len(data), figsize=(15, 6))

    for i, data_point in enumerate(data):
        labels = data_point['labels']
        values = data_point['values']
        title = data_point['title']
        ax = axes[i]

        ax.bar(labels, values)  # Create bar chart
        ax.set_xlabel('Operating Systems')  # Set x-axis label
        ax.set_ylabel('Count')  # Set y-axis label
        ax.set_title(title)  # Set chart title

        # Optional: Rotate long x-axis labels for better readability
        ax.set_xticklabels(labels,rotation=90, ha='right')

    plt.tight_layout()
    plt.show()



c = SearchClient()


cidr = ['152.1.0.0/16','152.7.0.0/16','152.14.0.0/16']

# The aggregate method constructs a report using a query, an aggregation field, and the
# number of buckets to bin.
data = []
for i in cidr:
    temp = {'labels':[],'values':[],'title':i}
    report = c.v2.hosts.aggregate(
        "autonomous_system.bgp_prefix: {}".format(i),
        "operating_system.product",
        num_buckets=10,
    )

    for j in report['buckets']:
        temp['labels'].append(j['key'])
        temp['values'].append(j['count'])

    data.append(temp)
```

plot_bar_charts(data)

Commands:

- operating_system.product
- services.service_name
- Services.software.product

Data Collection Process:

The data collection process utilized the Censys Search API to query and aggregate information about IPv4 hosts within the 'ncsu.edu' domain, specifically focusing on operating system distribution. Queries targeted specific network blocks (152.1.0.0/16, 152.7.0.0/16, 152.14.0.0/16) and used the autonomous_system.bgp_prefix field to ensure results stayed within the NCSU domain boundaries. The aggregate method grouped the results by the commands, providing counts for the top data found. This generated a structured dataset, with labels representing operating systems and values indicating their frequency within each subnet.

Within the aggregate method, the query includes filters like "autonomous_system.bgp_prefix: {}".format(i), where 'i' iterates through the CIDR ranges. This filter helps narrow down the search to hosts within the specified IP ranges. Additionally, the aggregation is performed based on the operating system's product field, allowing the code to gather data on the distribution of operating systems across the identified hosts.

The resulting data is structured into a list of dictionaries, with each dictionary containing information about a specific CIDR range. Each dictionary includes labels (operating system names), corresponding values (counts of hosts using each OS), and a title (CIDR range). Finally, the plot_bar_charts function processes this data to create bar charts, visually representing the distribution of operating systems across the specified IP ranges.

Justification:

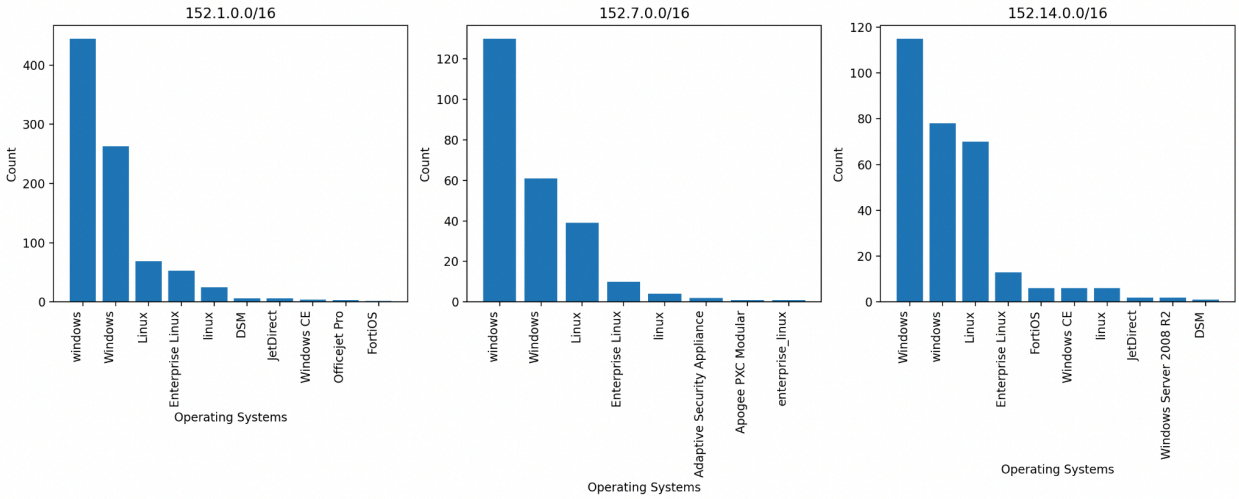
Collecting, visualizing, and analyzing host data in a network environment like 'ncsu.edu' is critical for effective cybersecurity. This method reveals important information about the network, such as the distribution of operating systems, web protocols in use, and the numerous web servers used.

Understanding the prevalence of various operating systems (e.g., Windows, Linux, macOS) enables security teams to prioritize patching and vulnerability management activities. By identifying shared operating systems, teams can concentrate on addressing the risks associated with their known flaws.

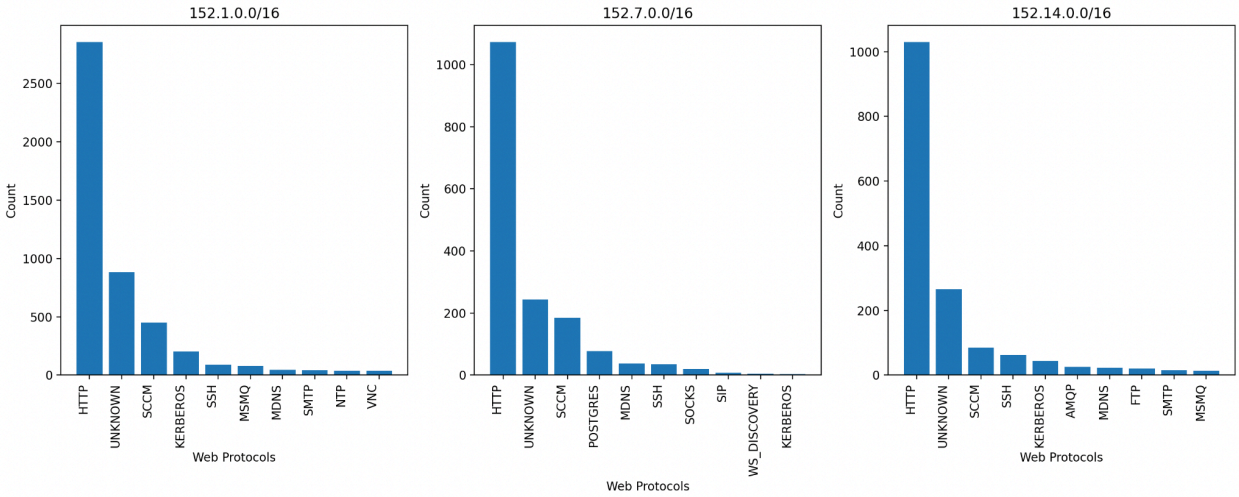
Analyzing web protocols (HTTP, HTTPS, FTP, etc.) reveals the types of services and applications that run on the network. To decrease potential points of exploitation, cybersecurity teams can deploy appropriate security solutions such as web application firewalls, secure coding techniques, and protocol-specific hardening.

Knowing the distribution of web servers (Apache, Nginx, IIS, etc.) provides insight into the technologies that support the network's web infrastructure. This allows protection techniques to be tailored to each type of web server's known vulnerabilities and attack vectors.
Visualizing host data can reveal trends or surprising distributions that require more examination. Identifying different operating systems, outdated systems, unsafe protocols, and unpatched web servers allows security teams to prioritize hardening work and reduce the attack surface.

Hosts by Operating System

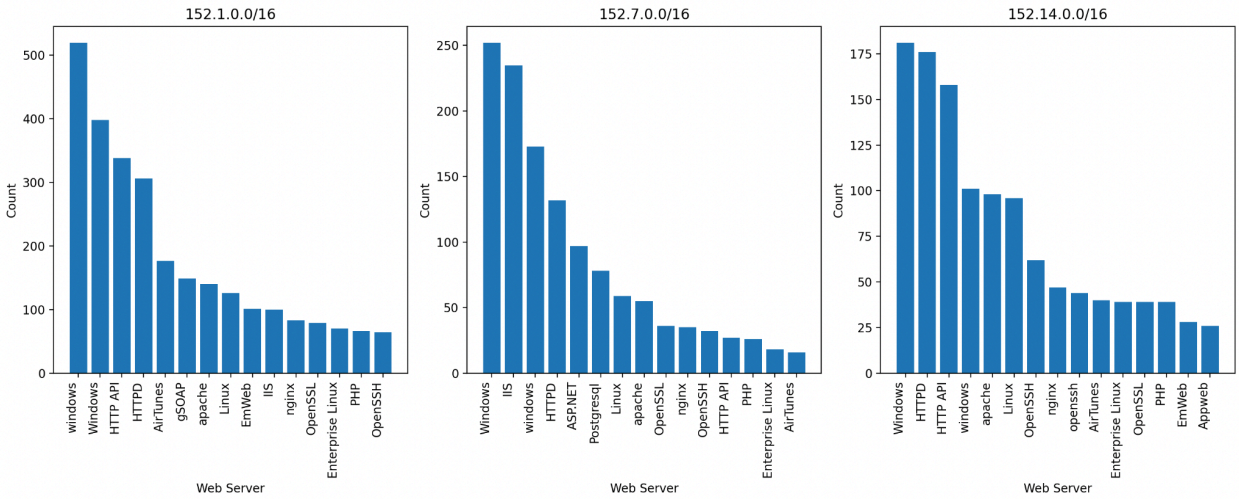


Analysis:
Windows-based systems maintain a strong presence across all three subnets within the NCSU domain. However, there's a healthy diversity of Linux systems, including standard Linux, Enterprise Linux, and even Fedora Core. This mix of operating systems is fairly common in large organizations and educational institutions. The dominance of Windows highlights the need for robust patch management and endpoint protection on those systems, as they are often targeted by malware. The presence of older Windows versions (e.g., Windows CE, Windows Server 2008 R2) warrants special attention, as these might be out of official support, leaving them exposed to known vulnerabilities. Linux systems, while generally more secure, require their own security measures.



Analysis:

HTTP, expectedly, stands out as the dominant web protocol, with a notable presence across all subnets. The high usage of UNKNOWN likely masks more specific HTTP or HTTPS implementations. It's interesting to see a significant amount of SCCM, suggesting widespread use of Microsoft System Center Configuration Manager for systems management within the NCSU network. The diverse web protocols necessitate focused defense strategies. HTTP traffic should be monitored for anomalies and, ideally, shifted to HTTPS wherever possible. Less common protocols like FTP, Telnet, and VNC should receive extra scrutiny as they often transmit data in plain text and possess known vulnerabilities. The presence of database protocols like MySQL and Postgresql necessitates strong access controls and database hardening.



Analysis: The results show a split landscape. Windows/IIS holds a significant share but Apache on Linux hosts is also widely prevalent. Nginx's growing popularity is evident as well. The data also reveals diverse embedded web servers (AirTunes, EmWeb) likely used by network appliances and IoT devices. Each web server software has its own set of known vulnerabilities. Security teams must track relevant advisories and updates for both Windows/IIS and Apache/Nginx. Misconfigurations are a common threat; hence secure configuration practices are paramount. The discovery of embedded web servers underscores the need for IoT/device discovery and risk assessment, as these are often overlooked in vulnerability management.

Part 3: Interesting Security Findings

Interesting Security-Related Findings

- Popularity of Apache Web Servers: Censys data revealed a significant Apache footprint within the NCSU domain. Since all three vulnerable IP addresses run Apache, this becomes a significant focus area. Apache, while robust, has a history of vulnerabilities. Its popularity makes it a common target.
- Multiple WordPress Installations: The presence of numerous WordPress hostnames (especially test/demo environments) is noteworthy. WordPress is a popular CMS, but plugins and outdated versions are frequent vulnerability sources. CVEs like CVE-2022-31813, CVE-2022-23943, CVE-2022-22720 often target WordPress plugins or core vulnerabilities.
- Potentially Unpatched Systems: The mix of CVEs, some dating back to 2021, suggests possible delays in patching within the environment. This highlights a potential weakness in NCSU's vulnerability management process.The mix of Windows, Linux variants, and some legacy systems (Windows CE, Windows Server 2008 R2) creates a complex patching landscape.

Recommendations:

- Prioritized Patching: NCSU should immediately address vulnerable systems, especially WordPress installations. Focus on CVEs with known exploits or high severity.
- WordPress Security Audit: NCSU should review their WordPress hardening practices:
- Plugin Management: Scrutinize plugins used, removing unnecessary ones.
- Updates: Enforce regular updates to the core, themes, and plugins.
- Vulnerability Scanning Review: Ensure NCSU has regular vulnerability scans covering web servers and WordPress.
- Patch Management Process: Help NCSU review patch application timing, especially for critical or internet-facing systems.

Discovery Process

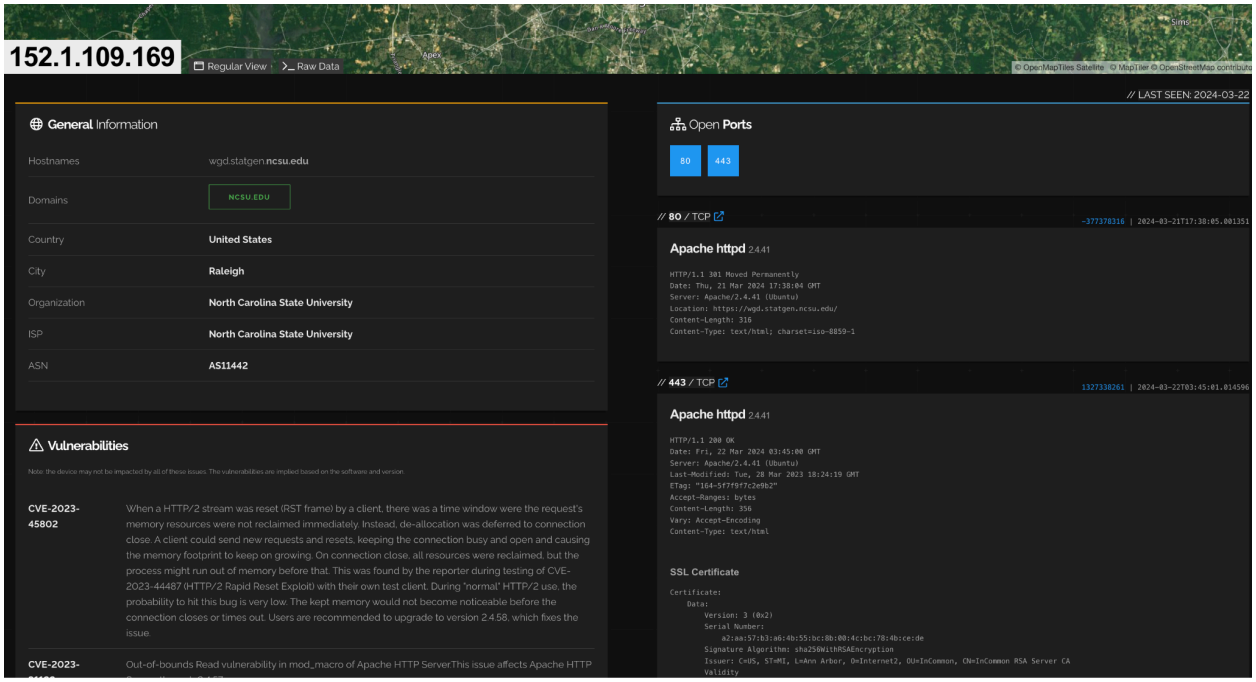
- Censys Initial Data: Used Censys to query the 'ncsu.edu' domain, focusing on operating system and service banners to identify Apache web servers.
- Targeted Shodan Queries: (With instructor permission)

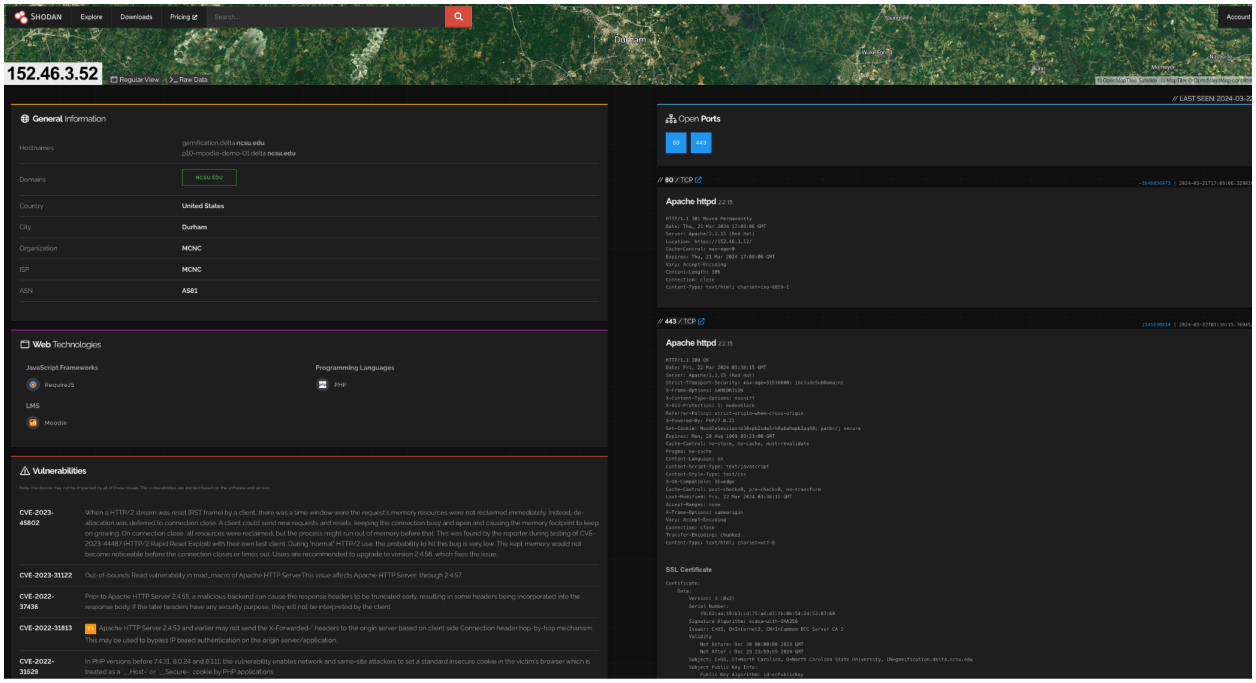
    - IP Filtering: Used Shodan API with the IP address from Censys to get more detailed host information.
    - Service Banners: Looked for specific Apache versions mentioned in CVEs.
    - WordPress Fingerprinting: Search keywords within hostnames (like those containing 'wordpress-test') to support vulnerability correlation at Shodan.

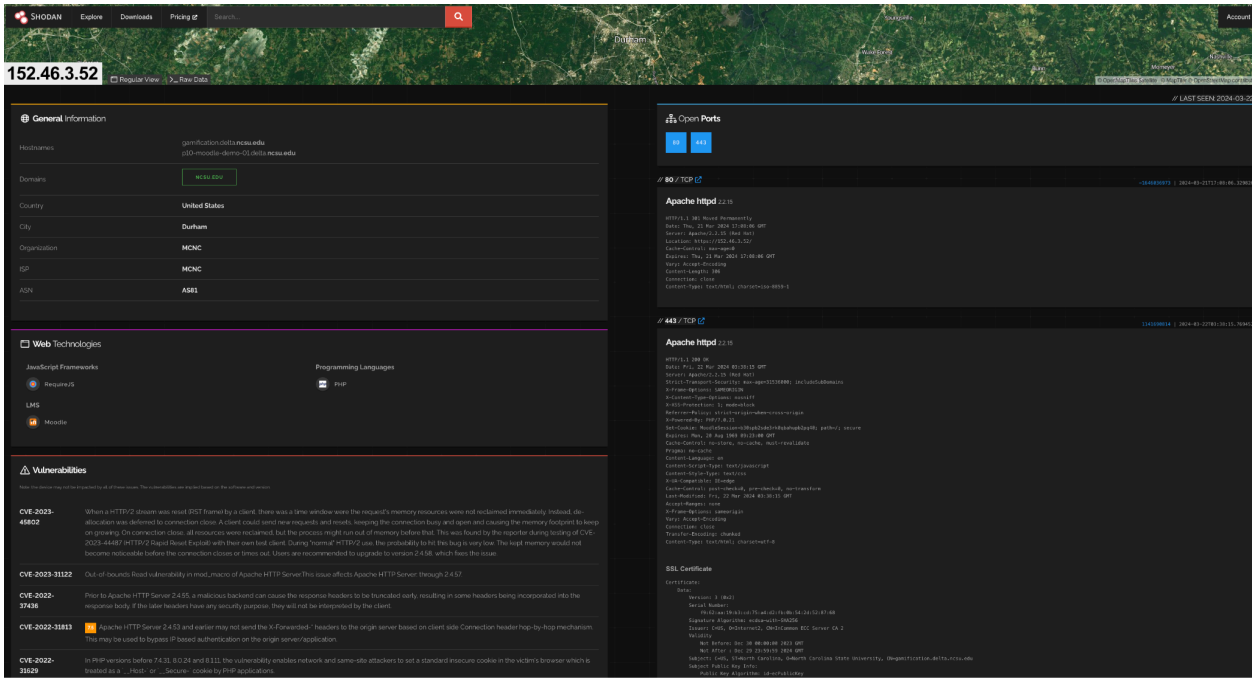
CVE Analysis

Vulnerable IP Found: 152.1.109.169, 152.46.3.52, 152.46.29.191

- CVE-2022-31813 (Apache HTTP Server)
    - Type: Path Traversal and Remote Code Execution (potential for full system compromise)
    - Severity: Critical
    - Impact: A successful attacker could gain complete control of the affected system. This is highly dangerous for publicly accessible servers.
- CVE-2022-22720 (Apache HTTP Server)
    - Type: Denial of Service, Information Disclosure
    - Severity: High
    - Impact: Attackers could crash the web server, making websites unavailable and potentially leaking sensitive information in the process.
- CVE-2021-44790 (Apache HTTP Server)
    - Type: Buffer Overflow (potential code execution)
    - Severity: High
    - Impact: If exploited, it could allow attackers to potentially run malicious code on the server, but it's slightly less severe than a full remote compromise like CVE-2022-31813.
- CVE-2022-23943 (Apache Tomcat)
    - Type: Denial of Service, Potential Remote Code Execution
    - Severity: Critical
    - Impact: Attackers could make the system unresponsive and might be able to remotely execute code depending on the server's configuration. Note that this likely only applies if Apache Tomcat is used in conjunction with Apache HTTP Server.
- CVE-2022-28615 (Apache Druid)
    - Type: Remote Code Execution
    - Severity: Critical
    - Impact: Extremely dangerous if the Apache Druid software is in use on the 152.1.109.169 system. Attackers could fully take over the server.

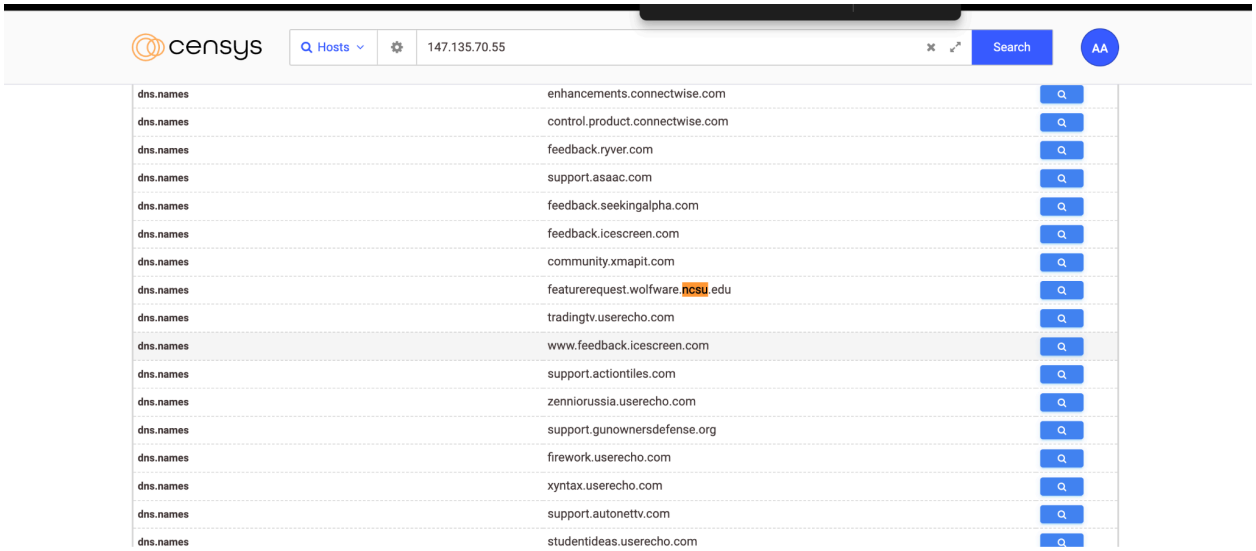

152.1.109.169
Regular View
Raw Data
// LAST SEEN: 2024-03-22
General Information
Hostnames
wgd.statgen.ncsu.edu
Domains
NCSU.EDU
Country
United States
City
Raleigh
Organization
North Carolina State University
ISP
North Carolina State University
ASN
AS11442
Open Ports
80
443
// 80 / TCP
Apache httpd 2.4.41
HTTP/1.1 301 Moved Permanently
Date: Thu, 21 Mar 2024 17:38:04 GMT
Server: Apache/2.4.41 (Ubuntu)
Location: https://wgd.statgen.ncsu.edu/
Content-Length: 316
Content-Type: text/html; charset=iso-8859-1
// 443 / TCP
Apache httpd 2.4.41
HTTP/1.1 200 OK
Date: Fri, 22 Mar 2024 03:45:00 GMT
Server: Apache/2.4.41 (Ubuntu)
Last-Modified: Tue, 28 Mar 2023 18:24:19 GMT
ETag: "164-5f7f9f7c2e9b2"
Accept-Ranges: bytes
Content-Length: 356
Vary: Accept-Encoding
Content-Type: text/html
SSL Certificate
Vulnerabilities
Note: the device may not be impacted by all of these issues. The vulnerabilities are implied based on the software and version.
CVE-2023-45802
When a HTTP/2 stream was reset (RST frame) by a client, there was a time window were the request's memory resources were not reclaimed immediately. Instead, de-allocation was deferred to connection close. A client could send new requests and resets, keeping the connection busy and open and causing the memory footprint to keep on growing. On connection close, all resources were reclaimed, but the process might run out of memory before that. This was found by the reporter during testing of CVE-2023-44487 (HTTP/2 Rapid Reset Exploit) with their own test client. During "normal" HTTP/2 use, the probability to hit this bug is very low. The kept memory would not become noticeable before the connection closes or times out. Users are recommended to upgrade to version 2.4.58, which fixes the issue.
CVE-2023-
Out-of-bounds Read vulnerability in mod_macro of Apache HTTP ServerThis issue affects Apache HTTP


SHODAN
Explore
Downloads
Pricing
Search...
Account
152.46.3.52
Regular View
Raw Data
// LAST SEEN 2024-03-22
General Information
Hostnames
gamification.delta.ncsu.edu
p10-moodle-demo-01.delta.ncsu.edu
Domains
NCSU.EDU
Country
United States
City
Durham
Organization
MCNC
ISP
MCNC
ASN
AS81
Web Technologies
JavaScript Frameworks
RequireJS
Programming Languages
PHP
LMS
Moodle
Open Ports
80
443
// 80 / TCP
Apache httpd 2.2.15
// 443 / TCP
Apache httpd 2.2.15
SSL Certificate
Vulnerabilities
CVE-2023-45802
When a HTTP/2 stream was reset (RST frame) by a client, there was a time window were the request's memory resources were not reclaimed immediately. Instead, de-allocation was deferred to connection close. A client could send new requests and resets, keeping the connection busy and open and causing the memory footprint to keep on growing. On connection close, all resources were reclaimed, but the process might run out of memory before that. This was found by the reporter during testing of CVE-2023-44487 (HTTP/2 Rapid Reset Exploit) with their own test client. During "normal" HTTP/2 use, the probability to hit this bug is very low. The kept memory would not become noticeable before the connection closes or times out. Users are recommended to upgrade to version 2.4.58, which fixes the issue.
CVE-2023-31122
Out-of-bounds Read vulnerability in mod_macro of Apache HTTP ServerThis issue affects Apache HTTP Server through 2.4.57.
CVE-2022-37436
Prior to Apache HTTP Server 2.4.55, a malicious backend can cause the response headers to be truncated early, resulting in some headers being incorporated into the response body. If the later headers have any security purpose, they will not be interpreted by the client.
CVE-2022-31813
Apache HTTP Server 2.4.53 and earlier may not send the X-Forwarded-* headers to the origin server based on client side Connection header hop-by-hop mechanism. This may be used to bypass IP based authentication on the origin server/application.
CVE-2022-31629
In PHP versions before 7.4.31, 8.0.24 and 8.1.11, the vulnerability enables network and same-site attackers to set a standard insecure cookie in the victim's browser which is treated as a __Host- or __Secure- cookie by PHP applications.



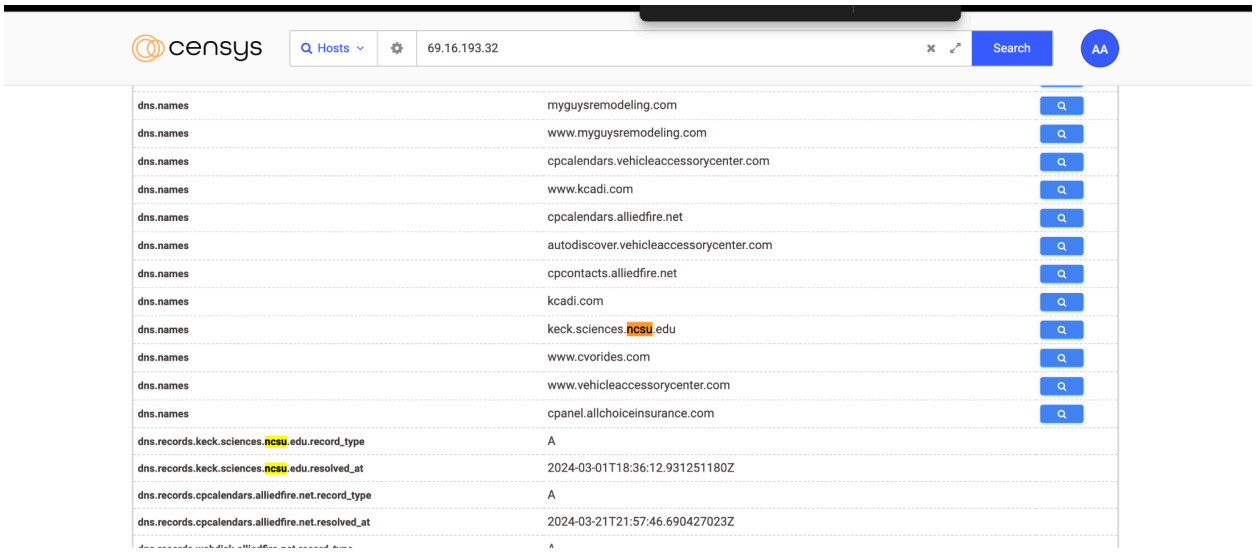
Part 4: Identifying External Shadow IT

IP Address: 147.135.70.55
DNS Name: featurerequest.wolfware.ncsu.edu
Record Type: CNAME

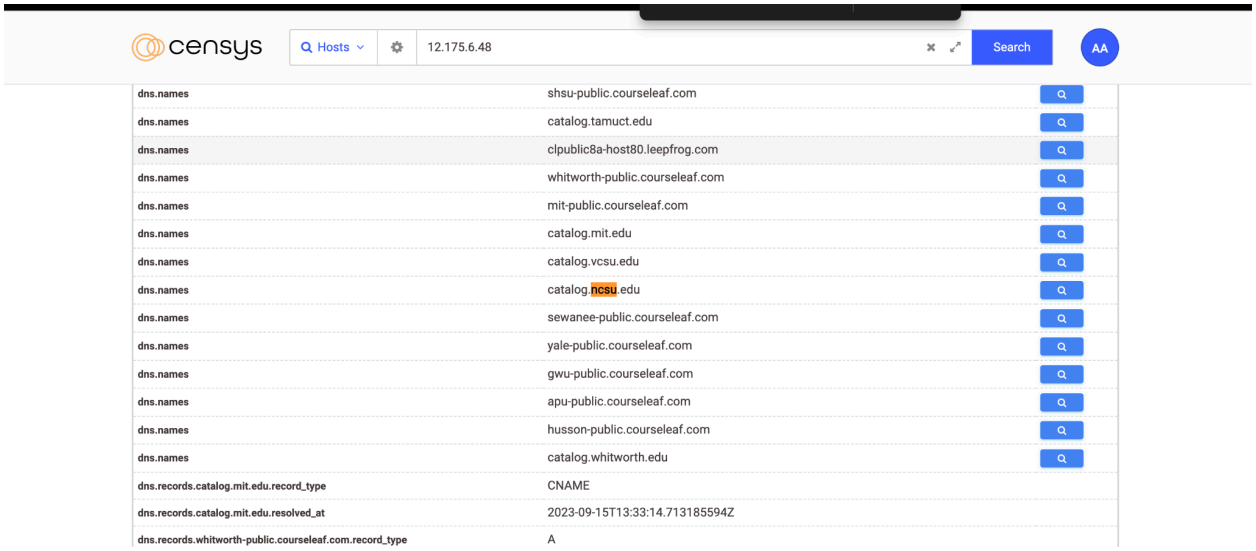


IP Address 2: 69.16.193.32

DNS Name: keck.sciences.ncsu.edu
Record Type: A



IP Address: 12.175.6.48
DNS Name: catalog.ncsu.edu
Record Type: CNAME



Analysis:

- featurerequest.wolfware.ncsu.edu (147.135.70.55):
    - Points to an IP outside NCSU's ASN.
    - CNAME record suggests that this subdomain likely utilizes a cloud-based service for the wolfware application.
- keck.sciences.ncsu.edu (69.16.193.32):
    - Also outside NCSU's ASN.

    - A record (direct IP mapping) indicates it might be a standalone server or a service directly hosted on a cloud provider's IP.
- catalog.ncsu.edu (12.175.6.48):
    - Another external IP.
    - CNAME record, once again suggesting a cloud-based service is likely behind this subdomain.

Process of obtaining results

1. Initial Data Gathering (Censys)

Focused on the 'ncsu.edu' domain and queried Censys.io to identify IP ranges associated with NCSU's Autonomous System Number (11442). This helped establish a baseline of what's "inside" NCSU's network. Exported the results as a JSON file (your ncsu1.json).

2. Local Data Filtering

Employed the provided Python script (read_json_file) to process your ncsu1.json file. The core logic of the script was to isolate IP addresses that did NOT belong to the NCSU ASN (11442). The script then checked the remaining IPs for the presence of 'dns' data, specifically 'reverse_dns' records. Output highlighted IPs and their associated reverse DNS names resolving to the 'ncsu.edu' domain or its subdomains.

3. Analysis and Insights

The IPs discovered (147.135.70.55, 69.16.193.32, 12.175.6.48) not being part of the NCSU ASN signifies the likely use of external providers for certain services under the ncsu.edu domain.

CNAME vs. A Records: The presence of CNAME records suggests cloud-based services, while the A record points to a possibly standalone server or a service directly hosted on a cloud provider's IP.

Part 5: Impact of IPv6

IPv4, with 32-bit addresses, provides approximately 4.3 billion distinct addresses. IPv6, with 128-bit addresses, offers 2^128 potential addresses. This scale makes exhaustive scanning almost impossible.

Traditional scanning approaches, such as brute-force scanning of successive IP addresses, become extremely resource-intensive and time-consuming due to the sheer magnitude of IPv6 address ranges. This difficulty can result in inadequate or inaccurate host databases because scanning all IPv6 addresses is impractical. Brute-force scanning the whole IPv6 address space is impossible due to the large quantity of addresses. With IPv6, enormous amounts of address space will go unused. Scanning programs may overlook a big number of active hosts simply because they are not searching in the correct places. Even targeted IPv6 scans create a large amount of data, which might be difficult to store and analyze.

IPv6 frequently uses random or pseudo-random address generating algorithms to provide proper address coverage. Instead of scanning the full IPv6 address space, this strategy seeks to optimize scanning efforts by focusing on areas likely to include active hosts.

Instead of exhaustive scans, tools must rely on:
Prior Information: Seeds based on known DNS records, BGP routing data, etc.
Sampling Techniques: Scanning representative portions of the IPv6 space.
Focused Targeting: Scanning specific subnets or address ranges based on intelligence.

The research article "IPv6 Address Scanning in the Internet" investigates these issues and offers insights into IPv6 scanning approaches, address generating strategies, and implications for Internet-wide scanning activities.
The paper highlights these key points:

- Building lists of potential IPv6 addresses using external sources (DNS, leaked addresses)
- Sharing scan results and target lists across the security community
- Developing techniques to identify likely patterns of IPv6 address allocation for more targeted scans.

Changes required by Censys and Shodan

- These platforms likely collaborate with network operators and researchers to gain information about active IPv6 ranges.
- Their scanning methods must use sophisticated heuristics and optimizations to focus on areas of the IPv6 address space with higher probabilities of active hosts.

IPv6 represents a significant barrier to standard thorough internet scanning. To remain effective in the IPv6 era, tools must evolve regularly, incorporate external data, and employ clever targeting strategies.