NVCAT: A Network Visibility Configuration Analysis Tool

Gabriel Siewert
Georgia Institute of Technology
Atlanta, GA

***Abstract*—Government auditing organizations such as the Government Accountability Office must analyze the network device configurations of other government agencies. This includes checking that traffic monitoring tools are enabled and that Access Control Lists are sufficiently restrictive with permissions. Until now, this analysis had to be performed manually. The Network Vulnerability Configuration Analysis Tool (NVCAT) is an application which automates this process, ensuring that vulnerabilities or insecure defaults present in a network device's configuration are more quickly detected and accurately reported. NVCAT is an open-source project with an intuitive front-end interface, offering a variety of analysis capabilities to any user interested in improving a network's visibility and security.**

I. INTRODUCTION

In workflows requiring an organization's network security to be assessed or audited, a manual review of network settings is inefficient [1]. Utilizing a tool that is consistent in its analysis minimizes the possibility of error when vulnerabilities are reported. In particular, the Government Accountability Office (GAO) routinely performs audits of other government agencies' information systems. This involves analyzing the running configurations of routers and switches, and the Access Control Lists (ACLs) of firewalls. There are various settings that auditors will flag as insecure, either for limiting the visibility of network traffic or for being excessively permissive. Being able to more efficiently audit or assess a system's network security will lead to faster identification of poor security practices, and consequently save auditors a significant amount of time and effort. This forms the basis of the need for a product like the Network Visibility Configuration Analysis Tool (NVCAT).

NVCAT is an application which assesses a network's visibility through analysis of a configuration or ACL file. It is meant to be an open-source, free-to-use, intuitive front-end application which generates a visibility report based on user specifications. The reports generated by NVCAT consolidate data that would otherwise require significant effort to synthesize. The most useful feature of NVCAT is its batch analysis capability. This allows a user to generate many reports at once, removing the need for examining individual files. In this case, an aggregate summary report will be created, detailing the extent to which each network device from a particular organization is monitoring traffic, forbidding connections, or otherwise implementing best network security practices.

While NVCAT is not the first network configuration analysis tool, it is the first tool of its kind that was developed specifically to assist government auditing efforts. In addition to being open-source and free-to-use, NVCAT brings together several disparate features that no other tool offers in combination: NetFlow and SPAN analysis, firewall ACL analysis, HTTP analysis, local and remote data retrieval, a notification system, and report encryption. Although NVCAT has been developed in collaboration with the auditing agency GAO, it is expected to be useful as a general network security assessment tool.

This paper will be structured in the following manner. First, a necessary background will be established. Next, products similar to NVCAT will be discussed, and major differences will be noted. The NVCAT project will then be described in detail, including its conception, implementation process, and challenges that arose throughout. Following this, the evaluation methodology, results, and future assessment strategy will be reviewed. Finally, conclusions will be drawn and recommendations for future work will be suggested.

II. BACKGROUND

This section will provide a background on NetFlow, SPAN, and firewall Access Control Lists. These are the primary traffic monitoring and control tools that NVCAT checks for. Each of these aspects of network visibility/security will be described in detail, in addition to their relevance with regard to NVCAT and GAO auditing efforts.

A. NetFlow

NetFlow is a network protocol developed by Cisco [2]. NetFlow collects IP network traffic and provides visibility into the network for administrators. This also provides an audit trail of network traffic. More specifically, NetFlow provides the following information (when enabled correctly):

- Application and network usage
- Network productivity and utilization of network resources
- The impact of changes to the network
- Network anomaly and security vulnerabilities
- Long term compliance issues

The above information is known as an IP flow, or simply a flow [2]. A flow is the structured information NetFlow provides after collecting traffic. Before discussing NetFlow any further, some other basic definitions must be established. An interface (also known as a network interface controller) is a network device which connects a computer to a network. An interface can be conceptualized as sitting between a computer and a network. Traffic from the computer to the network must pass through the interface, and vice versa. Ingress network traffic is incoming traffic (traffic going into an interface, regardless of direction). Egress traffic is outgoing traffic (i.e., traffic leaving an interface, regardless of direction).

NetFlow can be enabled as follows [2]. First, a flow exporter must be defined and enabled. This will assign a NetFlow collector to the network device. If there is no exporter, then collected flows will not be sent anywhere for analysis. For NetFlow to be enabled on an interface, there must be a monitor map applied to the interface. If the interface uses an IPv4 address, the monitor map must be an IPv4 monitor map for NetFlow to be enabled. Similarly, if the interface uses an IPv6 address, the monitor map must be an IPv6 monitor map for NetFlow to be enabled. Both types of monitor maps must be enabled if the interface has both IPv4 and IPv6 addresses. Monitor maps are configured for ingress traffic, but not egress traffic. This is to avoid unwanted noise from duplicate traffic (e.g., the traffic entering an interface could be the same traffic leaving the interface in the same direction).

Fig. 1 provides an illustrative example of NetFlow's capabilities. The diagram depicts a NetFlow-enabled interface receiving a packet. NetFlow then creates a flow from the packet's data. For example, the source and destination addresses give insight into who is generating and receiving the traffic, while the ports used indicate the purpose of the traffic (e.g., port 22 would indicate an SSH connection attempt) [2].

When conducting audits, GAO auditors are interested in NetFlow because NetFlow being disabled or incorrectly configured on a network device indicates a lack of internal visibility into the network. A lack of internal visibility leaves open the potential for unchecked lateral movement within the network. Therefore, if bad actors have already penetrated the network's perimeter, they can continue unnoticed with malicious activity, whether passive or active [1]. This is why NVCAT allows users to check a network device for NetFlow.

B. SPAN

SPAN stands for Switched Port Analyzer, which is also called port mirroring outside of Cisco contexts [3]. Like NetFlow, SPAN is a traffic monitoring tool which provides traffic data to a network analyzer. Enabling SPAN requires a source and a destination port to be specified. However, unlike NetFlow, SPAN does not provide its collected traffic in such a structured fashion. Rather, all network packets seen on the source are transmitted to the destination port. This provides full visibility into traffic on the source. Also different from NetFlow, SPAN does not require an IP address to be enabled on an interface; SPAN can be applied to a virtual Local Area Network (VLAN). Therefore, the source can be a source port or a source VLAN. Firewalls and routers do not support SPAN.

Fig. 2 provides a simple example of how SPAN works. In the diagram, device A is designated the source (or monitor) port, while device D is designated the destination (or analysis) port. Given this configuration, device D will be sent (or mirrored) all packets transmitted between devices A and B, or devices A and C [4].

When conducting audits, GAO auditors are interested in SPAN for the same reason they are interested in NetFlow: the auditors want to see that the agency under audit has visibility within their network. Whether that is accomplished via NetFlow or SPAN is incidental. There are cases where both NetFlow and SPAN could be enabled on the same device. This could occur with layer 3 switches, which are common devices to see when conducting audits. Layer 3 switches are simply switches that can perform routing as well, named such because they operate on layers 2 (data link) and 3 (network) of the Open Systems Interconnection (OSI) model [5]. In the case where NetFlow and SPAN are enabled on the same device, auditors want to see total coverage, where every interface has at least NetFlow or SPAN enabled. This is why NVCAT offers SPAN analysis.

C. Access Control Lists

An Access Control List (ACL) is a list of permissions stored on a firewall. These permissions manage incoming and outgoing traffic. For example, a particular ACL entry might determine whether a packet sent from a particular IP address outside the network will be permitted past the firewall. Fortunately, firewall ACLs have an "implicit deny" policy. That is, any packet not explicitly permitted is implicitly denied (dropped) [6]. An ACL entry consists of a source and destina-

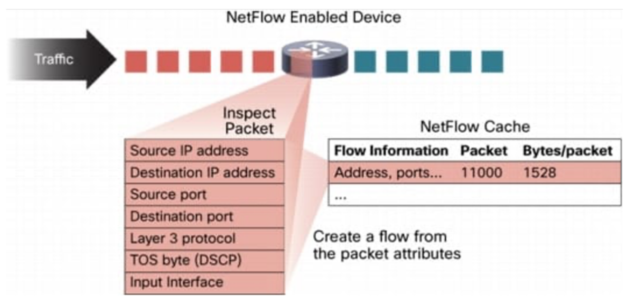


Fig. 1. Example of NetFlow [2].

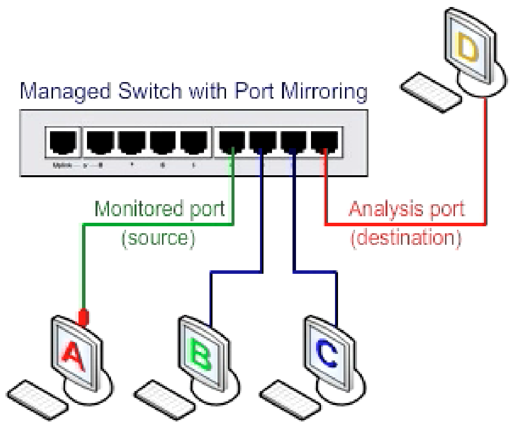

Managed Switch with Port Mirroring
Monitored port (source)
Analysis port (destination)
A
B
C
D

Fig. 2. SPAN diagram [4].

tion IP address (or address range), an IP protocol, ports, and often other parameters such as traffic logging specifications. Using “any” instead of an IP address refers to all possible IP addresses. The IP protocol specified can be, for example, TCP or UDP. Listing IP as the protocol indicates that the entry applies to all protocols.

Fig. 3 shows a flowchart illustrating the process that occurs when a firewall receives a packet. If there is no ACL on the interface, the decision is relegated to the routing table. Otherwise, a matching ACL statement will be searched for linearly. That is, if the first entry in the ACL does not explicitly allow the received packet, the next entry is checked. An explicit denial will immediately drop the packet. Otherwise, each entry in the ACL will be checked in order until either a match is found (in which case the packet is accepted) or the end of the ACL is reached. This introduces an important idea regarding ACLs: the ordering of statements in an ACL can make a difference in security. For example, suppose there is an insecure “permit” entry in an ACL, allowing any source IP address to connect to any destination IP address over TCP, port 445. For example, suppose there is an explicit “deny” entry for a particular IP address range over TCP, port 445. This is a secure ACL entry, as port 445 has been used in malware attacks on Windows machines [7]. Suppose there is also an insecure “permit” entry allowing packets that would otherwise be caught by the “deny” statement. Of course, this statement should not be in the ACL at all. However, if it is listed before the “deny” entry, the “deny” entry will have no effect at all, because the “permit” statement will match the packet before the “deny” statement is ever reached.

When conducting audits, GAO auditors are interested in ACL entries because it is relatively easy to accidentally set an insecure permission. In particular, GAO auditors are concerned with agencies whose ACLs allow connections over ports used by outdated services or services often exploited by bad actors (e.g. telnet, NetBIOS) [1]. This is why NVCAT offers ACL analysis.

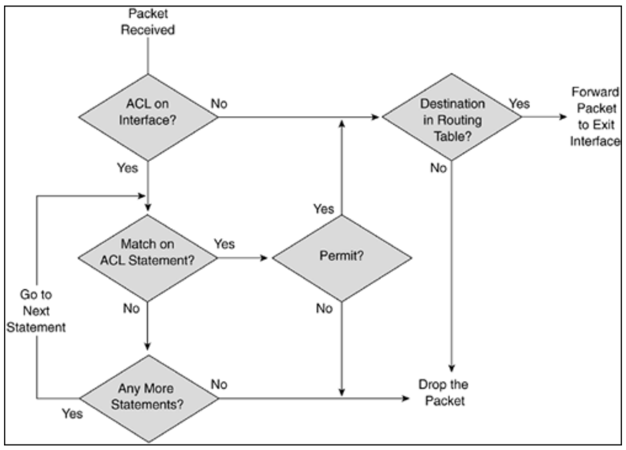


Fig. 3. ACL flowchart [8]

III. Related Work

This section will take note of existing products similar to NVCAT, and describe how NVCAT is different from each one.

The Batfish tool is described as “an open source network configuration analysis tool” [9]. On the surface, Batfish and NVCAT may appear to be quite similar. However, there are two important distinctions to make. First, Batfish is a Python package. Therefore, a user must know Python to use Batfish. NVCAT offers a front-end interface to the user in the style of a desktop application. Beyond the installation of required dependencies, NVCAT is quite straightforward; preliminary evaluation results confirm that NVCAT is simple to use for a first-time user (see section V-B). The second main difference between Batfish and NVCAT is that Batfish only analyzes network configurations. NVCAT analyzes configurations, firewall ACLs, and aims to expand further to support analysis of other common network device command outputs (see section VII).

Another open-source product similar to NVCAT is the Cisco Config Analysis Tool, or CCAT [10]. Like Batfish, CCAT does not offer encryption for any reports generated, and it does not support analysis for firewall ACLs. Additionally, CCAT does not offer a Graphic User Interface (GUI), while NVCAT does. CCAT also does not make the same checks as NVCAT. CCAT’s checks are based on the Cisco Guide to Harden IOS Devices [11], which mentions NetFlow but not SPAN. Finally, as its name suggests, CCAT only analyzes Cisco network device configurations. While NVCAT in its current form also only supports analysis of Cisco network devices, there are plans to expand NVCAT’s functionality to encompass additional network and firewall vendors (see section VII).

There are multiple commercial products which share similar goals with NVCAT. For example, ManageEngine’s Firewall Analyzer [12] and NetFlow Analyzer [13] are two separate products which aim to improve network visibility for the customer. These products analyze network traffic in real time, rather than reporting on insecure defaults. Therefore, in an auditing context, the organizations being audited would benefit more from these tools than the auditors themselves [1]. It is also worth noting that NVCAT combines its NetFlow, SPAN, and firewall analysis into one tool, whereas ManageEngine’s tools must be purchased separately. One additional commercial product is the Network Configuration Manager (NCM) from SolarWinds [14]. This product has assessment features and supports Cisco and Palo Alto devices. However, like ManageEngine’s tools, it is not free. NVCAT is free to use, while NCM prices start at $1,630 [14], and ManageEngine’s NetFlow Analyzer alone could cost an organization with 100 interfaces over $10,000 [15].

In summary, NVCAT offers the following features, which are not offered in combination by any of the aforementioned products.

- Is free to use
- Is open-source
- Has desktop application-style GUI

- Checks for NetFlow, SPAN, HTTP, and insecure ACL entries
- Allows user to notify network administrator
- Allows user to encrypt reports containing sensitive data
- Allows user to remotely connect to network device and obtain configuration for analysis

IV. NVCAT

This section will describe NVCAT in detail. First, the origin, scope, and intended goals of NVCAT will be discussed. The implementation process will then be reviewed, followed by an examination of the challenges faced throughout.

A. Concept

During the summer of 2020, I participated in an internship within GAO's Information Technology and Cybersecurity (ITC) team. Specifically, I worked with the Assistant Director of the Center for Enhanced Cybersecurity (CEC), Saar Dagani. During the internship, Dagani noted that a comprehensive tool for analyzing network device configurations would be extremely valuable to CEC auditors at GAO.

After further discussion of the project's viability, the internship was extended so that I could work closely with GAO while building NVCAT. Throughout the Fall 2020 semester, I held weekly meetings with Dagani. These meetings assisted in tracking the progress of the tool, determining next steps, and managing expectations and goals.

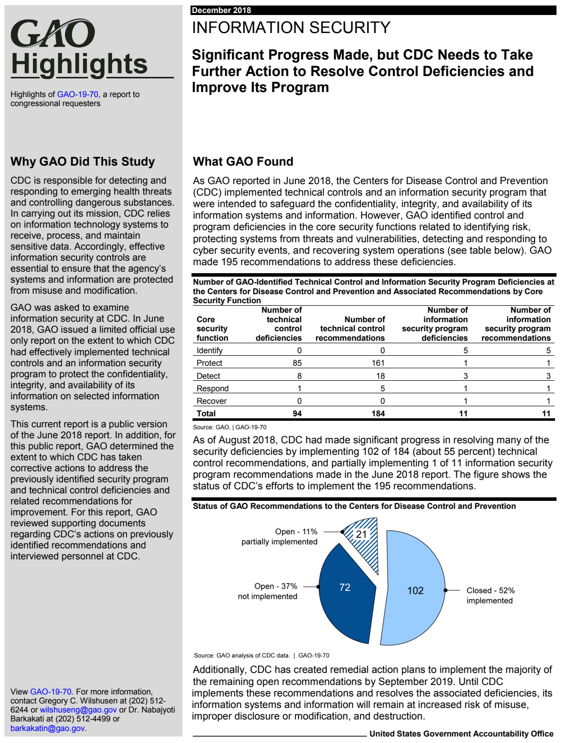
During a preliminary informational meeting, I was provided with the following context, which further established the need for a tool like NVCAT at GAO [16]. When working on an audit, the government entity under audit provides CEC auditors with all configuration files and outputs requested by GAO. These files are easier to parse when they are organized by device type and output type (e.g., the output of the "show running-config" command and the output of the "show access-list" command). However, many organizations will simply provide one folder containing all requested files, with file names that do not provide useful information to the auditors. It is often the case that the information systems of the government agencies under audit will be "hard on the outside, soft in the middle" [16]. In other words, the perimeter of a particular network may be very secure, while failing to adequately monitor and/or allowing unnecessary lateral movement within the network. Sometimes, when insecure default settings are found and findings are reported to the agency under audit, officials from that agency will be unable to explain the presence of the settings in the first place. For example, one agency (which cannot be named explicitly in this report) was contacted with findings from a GAO audit, after insecure ACL permissions were discovered. A representative from this agency was unable to identify why the overly permissive lines in the ACL file were present. Thus, there is occasionally a knowledge gap between the IT officials in the agency under audit and the IT auditors [16]. These conditions necessitate a tool that can automate the process of reporting insecure default settings in an auditing context. Furthermore, if the knowledge gap mentioned previously is present in private sector entities, NVCAT may be marketable towards those entities, whose IT staff could be unknowingly configuring their company's network devices with insecure default settings.

Keeping the above context in mind, long-term goals for NVCAT were determined accordingly [16]. The aim is to develop an interactive network assessment tool that can:

- Analyze many (100+) configurations at a time
- Comprehensively account for various network/firewall vendors and device platforms
- Check for the same misconfigurations and bad defaults that GAO auditors look for
- Generate reports on network visibility and security
- Translate generated reports into reportable findings with ease
- Maintain the confidentiality of sensitive data

Ultimately, the findings generated by NVCAT when it is used in GAO audits will contribute to official reports published by GAO. These reports are typically accessible to the public online, and each report is published along with a one-page summary of major findings, called a Highlights Page. An example of a Highlights Page is shown in Fig. 4. The page shows the most essential findings from a GAO report regarding an information security program at the Centers for Disease Control and Prevention (CDC).

After determining long-term goals for NVCAT, Saar Dagani and I considered the prototypical version of NVCAT that we



GAO Highlights

Highlights of GAO-19-70, a report to congressional requesters

December 2018

INFORMATION SECURITY

Significant Progress Made, but CDC Needs to Take Further Action to Resolve Control Deficiencies and Improve Its Program

Why GAO Did This Study

CDC is responsible for detecting and responding to emerging health threats and controlling dangerous substances. In carrying out its mission, CDC relies on information technology systems to receive, process, and maintain sensitive data. Accordingly, effective information security controls are essential to ensure that the agency's systems and information are protected from misuse and modification.

GAO was asked to examine information security at CDC. In June 2018, GAO issued a limited official use only report on the extent to which CDC had effectively implemented technical controls and an information security program to protect the confidentiality, integrity, and availability of its information on selected information systems.

This current report is a public version of the June 2018 report. In addition, for this public report, GAO determined the extent to which CDC has taken corrective actions to address the previously identified security program and technical control deficiencies and related recommendations for improvement. For this report, GAO reviewed supporting documents regarding CDC's actions on previously identified recommendations and interviewed personnel at CDC.

View GAO-19-70. For more information, contact Gregory C. Wilshusen at (202) 512-6244 or wilshuseng@gao.gov or Dr. Nabajyoti Barkakati at (202) 512-4499 or barkakatin@gao.gov.

What GAO Found

As GAO reported in June 2018, the Centers for Disease Control and Prevention (CDC) implemented technical controls and an information security program that were intended to safeguard the confidentiality, integrity, and availability of its information systems and information. However, GAO identified control and program deficiencies in the core security functions related to identifying risk, protecting systems from threats and vulnerabilities, detecting and responding to cyber security events, and recovering system operations (see table below). GAO made 195 recommendations to address these deficiencies.

Number of GAO-Identified Technical Control and Information Security Program Deficiencies at the Centers for Disease Control and Prevention and Associated Recommendations by Core Security Function

Core security function	Number of technical control deficiencies	Number of technical control recommendations	Number of information security program deficiencies	Number of information security program recommendations
Identify	0	0	5	5
Protect	85	161	1	1
Detect	8	18	3	3
Respond	1	5	1	1
Recover	0	0	1	1
Total	**94**	**184**	**11**	**11**

Source: GAO. | GAO-19-70

As of August 2018, CDC had made significant progress in resolving many of the security deficiencies by implementing 102 of 184 (about 55 percent) technical control recommendations, and partially implementing 1 of 11 information security program recommendations made in the June 2018 report. The figure shows the status of CDC's efforts to implement the 195 recommendations.

Status of GAO Recommendations to the Centers for Disease Control and Prevention

Source: GAO analysis of CDC data. | GAO-19-70

Additionally, CDC has created remedial action plans to implement the majority of the remaining open recommendations by September 2019. Until CDC implements these recommendations and resolves the associated deficiencies, its information systems and information will remain at increased risk of misuse, improper disclosure or modification, and destruction.

United States Government Accountability Office

Fig. 4. Highlights Page, GAO Report on IT systems at CDC [17].

wanted to bring to fruition by December 2020 [16]. These shorter-term goals involved developing an intuitive front-end interface and a back-end parsing logic which would come together to form an assessment tool that can detect whether a network device has NetFlow/SPAN enabled, or is using insecure ACL permissions. Additional goals were added and met throughout the implementation process, which is described in the following subsection.

B. Implementation

The GUI was the first piece of functionality that was implemented. The Python tkinter package was used to provide a simple, user-friendly front-end, and a general flow of use was determined [18]. The intended flow of use is the following:

1) User specifies type of device to be analyzed (router, switch, firewall)
2) User specifies platform (e.g., Cisco IOS)
3) User specifies source file(s) to be analyzed
4) User specifies destination file or folder for generated report(s)
5) User specifies password which is used to encrypt report(s)
6) NVCAT analyzes source file(s) and generates report(s)
7) User can enter password to read report, or any previously generated report

It should be noted that the above flow of use does not represent the total range of functionality of NVCAT, but rather represents one common potential use case. For example, the front page of the application has a "Password Protection" checkbox, allowing the user to decide whether report encryption is needed for their particular setting.

After a flow of use was established, the back-end database storage of reports was implemented using the Python sqlite3 module [19]. Following this, the report encryption and decryption functionality was implemented. Report encryption is an important feature for NVCAT to offer because NVCAT aims to be a useful tool for GAO auditors. During a time when remote work is increasing, the majority of GAO employees are working from home [1]. Because many of these employees may be working from their own personal computers, it is vital that sensitive data is not mistakenly leaked or stored alongside an auditor's personal files. The following measures were taken to ensure that encrypted data is secure:

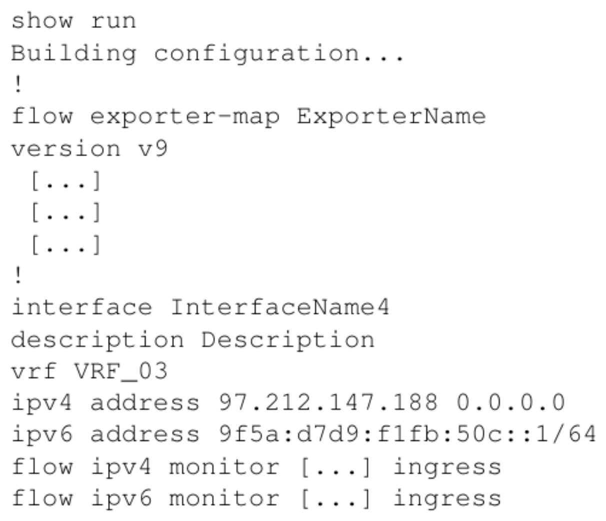
- A sufficiently complex password is required (when password protection is enabled; in a GAO-exclusive version of NVCAT, the aforementioned "Password Protection" checkbox would not exist, thus requiring all reports to be encrypted). The password must comply with Federal Information Processing Standards (FIPS). A FIPS-compliant password must have seven or more characters and include characters from at least three of the following: ASCII digits; lowercase ASCII; uppercase ASCII; non-alphanumeric ASCII; non-ASCII. An uppercase ASCII character or ASCII digit does not satisfy the requirement if it is the first or last character of the password, respectively [20].
- A FIPS-compliant encryption algorithm (Advanced Encryption Standard with a 256-bit key, using the Cipher block chaining mode of operation) was used [21].

After the back-end encryption and decryption functionality was implemented, remote data retrieval was implemented. This allows a user to provide the IP address of the network device to be analyzed, and the configuration or ACL will be retrieved from the device in real time. This functionality provides assurance that the configuration or ACL being analyzed is up-to-date, whereas if it were a file being analyzed, there is no guarantee that the contents of the file represent the associated network device in its current state. For further detail on how this functionality was implemented, see section IV-C.

Next, the back-end parsing logic was implemented to check for NetFlow and SPAN. Fig. 5 depicts an example of the "show running-config" output that is parsed to search for NetFlow and/or SPAN. The figure indicates that there is a flow exporter and an interface which has NetFlow enabled for both its IPv4 and IPv6 addresses.

Once NetFlow and SPAN checking were implemented, ACL parsing logic was implemented. This checks for several insecure permissions that GAO auditors typically look for, some of which are "permit tcp any", "permit udp any", and "permit … eq 23". These checks are determining whether the firewall allows any TCP traffic (i.e., with no limitations), any UDP traffic, or traffic where the destination port is port 23, respectively. Port 23 is used for Telnet, an outdated protocol with vulnerabilities [22]. Fig. 6 depicts an example of the "show access-list" output that is parsed to search for unnecessary or insecure ACL entries. The figure shows several permissions for general and UDP traffic from a host machine with a particular address to another host machine with a different address.

After implementing the analysis functionality for NetFlow, SPAN, and ACL entries, report generation was implemented. This was relatively simple; the Python tabulate module was



```
show run
Building configuration...
!
flow exporter-map ExporterName
version v9
 [...]
 [...]
 [...]
!
interface InterfaceName4
description Description
vrf VRF_03
ipv4 address 97.212.147.188 0.0.0.0
ipv6 address 9f5a:d7d9:f1fb:50c::1/64
flow ipv4 monitor [...] ingress
flow ipv6 monitor [...] ingress
```

Fig. 5. Example of "show running-config" output.

```
show access-list
  access-list inside_access line 3 extended
permit udp host 28.194.141.161 host
186.248.101.148 eq 5247 [...]
  access-list inside_access line 3 extended
permit udp host 143.201.6.46 host
173.45.68.135 eq 12223 [...]
  [...]
  [...]
  [...]
  access-list inside_access line 5 extended
permit ip host 113.151.145.138 host
104.226.243.71 [...]
  access-list inside_access line 5 extended
permit ip host 142.168.181.195 host
37.129.187.128 [...]
  access-list inside_access line 5 extended
permit ip host 152.168.17.247 host
254.84.191.204 [...]
  access-list inside_access line 3 extended
permit udp host 17.93.233.43 host
14.218.50.47 eq 12222 [...]
```

Fig. 6. Example of ”show access-list” output.

used to nicely format data in the text report. Following this, an additional goal was reached: a configuration check for HTTP was implemented. This additional check is based on guidance from the Cisco IOS Hardening Guide, which recommends disabling unused services [11]. This is another check that GAO auditors typically conduct; the incorporation of additional hardening checks is discussed in section VII.

An assessment scheme was determined next, in order to provide auditors with an immediate sense of a network's visibility/security upon viewing a report. Table I depicts an example of the report generated by NVCAT, including the results of the assessment. The particular network device being analyzed has been given a score of 80%, which is deemed to be good. The table columns show each interface associated with the device, the IP address(es) associated with the interface, and whether NetFlow is enabled on the interface (“Found”) or not (“Missing”). If an interface has an IPv4 and IPv6 address, NetFlow must be enabled for both addresses to provide total visibility on that interface. The score is 80% because there are ten interfaces, and eight have empty “Missing” cells. When ACL entries are analyzed, a Pass/Fail score is given, based on whether any insecure permissions were discovered.

Two final additions were made to NVCAT. A notification system was implemented, allowing a user, when viewing a report, to select a “Notify Admin” button. The user can then enter the email address of a network administrator, supervisor, or whoever is responsible for the security of the network device(s) assessed. The full text of the report will be sent to the email address indicated by the user. Finally, batch analysis was implemented, along with an assessment scheme to support aggregate summary reports. This allows a user to select a folder of files instead of a single file, and conduct the same analysis on each file in the folder. A report like the one shown in Table I is generated for each source file, and an additional report is generated, summarizing the findings of all files in aggregate.

C. Challenges

Building NVCAT was no trivial task. Multiple challenges were encountered throughout the development process. Two such challenges are described here.

Remote data retrieval was the first major hurdle in development. That is, successfully and securely authenticating to a remote network device, running a command, and obtaining its output. SSH authentication can be done in one of two ways: via username and password credentials, or with SSH public and private keys. Using SSH keys is the more secure method [23], but for the purposes of this application, it is also the less user-friendly option. This is because for SSH key authentication to work, the network device being connected to must already have the user machine's SSH public key stored. If the user does not have direct or immediate access to the network device being analyzed, this would significantly delay analysis results. Guidance from the National Institute of Standards and Technology (NIST) regarding best practices was used to ensure secure SSH authentication and connection [24]. The solution that was decided upon attempted to find a middle-ground in between maximum security and user-friendliness: NVCAT allows the user to decide between the two authentication methods. If the user selects SSH key authentication, the remote host will automatically be added to the user machine's `~/.ssh/known_hosts` file, but NVCAT will remove this entry before generating a report. The Python module paramiko was useful in connecting to and running commands on a remote host [25] [26].

Additionally, NVCAT's parsing and assessment logic is not at all simple, and will only grow in complexity as the application expands further (see section VII for further detail). The parsing logic had to be crafted in a manner that accurately pulled the needed information about NetFlow, SPAN, or ACL entries. This required sufficient familiarity with the syntax of these security measures. Viewing Cisco documentation [2] [6] [27], holding informational meetings with GAO officials [1] [16] [28], and studying example configurations provided by GAO officials assisted greatly in acquiring the background necessary to proceed further. The assessment scheme also posed some difficulty, as there are many edge cases that can arise from slight syntactical differences. In some cases, additional context may be needed that is not available in the file(s) analyzed, so a particular message stating as much would need to be displayed in the report. For example, if NVCAT is checking for SPAN on a switch, and the configuration file indicates a remote source for packets, the source's details will not be displayed in the same configuration file. Additional information would be needed to accurately assess that switch. If the source were turned off, for instance, no packets would be collected over SPAN, even if the configuration file ana-

TABLE I
NVCAT REPORT EXAMPLE

Suggested Score: 80.0% (Good)

Interface Name	IP Address	Found	Missing
InterfaceName0	66.76.33.23 (IPv4)	NetFlow (IPv4)	
InterfaceName2	254.127.11.137 (IPv4)		NetFlow (IPv4)
InterfaceName4	97.212.147.188 (IPv4); 9f5a:d7d9:f1fb:50c::1/64 (IPv6)	NetFlow (IPv4); NetFlow (IPv6)	
InterfaceName5	131.249.199.3 (IPv4)	NetFlow (IPv4)	
InterfaceName6	132.250.200.4 (IPv4); 74f6:51f4:8d00:b187::1/64 (IPv6)	NetFlow (IPv4)	NetFlow (IPv6)
InterfaceName7	133.251.201.5 (IPv4)	NetFlow (IPv4)	
InterfaceName8	134.252.202.6 (IPv4); 74f8:51f6:8d02:b189::1/64 (IPv6)	NetFlow (IPv4); NetFlow (IPv6)	
InterfaceName9	135.253.203.7 (IPv4)	NetFlow (IPv4)	
Vlan0	122.100.210.89 (IPv4)	NetFlow (IPv4)	
Vlan1	122.100.211.90 (IPv4)	NetFlow (IPv4)	

lyzed would have otherwise received a positive assessment. Implementing the batch analysis functionality while taking these edge cases into account presented yet another logistical obstacle. For the purposes of calculating aggregate scores, a solution was reached to assign a score of zero in cases where additional context is needed.

V. EVALUATION

This section discusses the strategy used to assess NVCAT's effectiveness, the results of that assessment, and an improved assessment strategy for future studies.

A. Methodology

In the interest of assessing the efficacy of NVCAT, the following evaluation strategy was established.

The government agency configuration and ACL files which assisted in the development process were used again. However, because these files are classified Secure But Unclassified (SBU), their contents contain sensitive information that cannot be made public. Therefore, in order to create configuration and ACL files that could be shared publicly, copies of several SBU files were made, and all sensitive information was removed or replaced with either trivial or randomly generated values. GAO official Saar Dagani was provided these safe files in addition to the project source code. Dagani then tested NVCAT and completed a user survey that was created for the purpose of evaluation [29].

The NVCAT user survey was written to assess the usability, accuracy, and projected benefits of the tool [29]. The survey consists of several questions asking the degree to which the respondent agrees or disagrees with a particular statement about NVCAT. Among the metrics measured are the ease of use for a first-time user, whether NVCAT provides results in an acceptable amount of time, whether the tool requires too much user input, whether the tool maintains the confidentiality of secure data, and whether the tool provides results that are comparable in accuracy to auditor's findings otherwise. Also measured is the respondent's estimate of the amount of time that NVCAT would save, if used in their workplace.

B. Results

The following results are based on responses to the NVCAT user survey from GAO official Saar Dagani.

Broadly, the feedback on NVCAT was overwhelmingly positive. The application received the highest marks by all metrics except for one: the amount of user input needed. The respondent indicated that the amount of user input required by NVCAT could be lower.

Notably, the respondent indicated that NVCAT would save GAO auditors an estimated twenty or more hours of work per week if the tool were used in audits. The following assumptions were made in order to obtain meaningful data from this metric. GAO official Saar Dagani indicated in an information gathering meeting that these assumptions are reasonable and representative of the tasks that are typically completed by GAO auditors [28]. First, it is assumed that "hours per week saved" refers to the number of hours per week saved for an entry-level GAO auditor during a period of time when an audit is taking place. Second, it is assumed that such an auditor would otherwise be conducting a manual review of the same number of files analyzed by NVCAT. Third, it is assumed that the entry-level auditor will be given an average of 120 files to analyze per week, and that the auditor will be capable of analyzing, on average, approximately 5 files in one hour. Given those assumptions in addition to NVCAT's ability to analyze the same number of files within seconds, the respondent estimated that using NVCAT would save GAO auditors twenty or more hours of work per week, as compared to a manual review.

C. Improvements

The NVCAT user response survey has since been updated to more reliably assess the tool. The chapter "Survey Research in HCI" from the Springer-published book *Ways of Knowing in HCI* was used as a reference for improving the language of the survey [30]. Questions that originally asked the respondent to indicate the extent to which they agreed or disagreed with a statement were updated to avoid causing acquiescence bias. Acquiescence bias is the tendency of some survey respondents to agree with a statement when presented with an agree/disagree question, regardless of the statement's meaning. These types of questions were changed to multiple choice questions that do not ask the respondent to agree or disagree with a statement. A unipolar construct was used instead, with labels for answer choices that "have been shown to be semantically equidistant from each other" [30]. This improved survey could be used in a real-world user study, and it will be used in all future assessments [31].

VI. Conclusion

The progress made thus far on NVCAT has laid the foundation for work that will continue at GAO. NVCAT received excellent feedback on a sample user study; its automated batch analysis functionality is projected to save GAO auditors upwards of twenty hours per week. This will allow for faster completion of audits, effectively enhancing the security of the audited agencies' network devices. Officials from GAO have expressed interest in improving upon NVCAT and using it to assist ongoing and future audits of other government agencies' information systems. GAO officials additionally noted that if NVCAT were to reach its full potential, the day-to-day workflow of auditors who use the tool may change substantially [1].

Additionally, an evaluation framework has been constructed [31]. This will allow for NVCAT to be assessed regularly, which in turn will allow for the project direction and improvement objectives to be guided by assessment results. The evaluation framework also creates the potential for testing by private sector entities. Any organization which hopes to improve its network security may be interested in using NVCAT and providing feedback on its usefulness. If the projected work saved by NVCAT for GAO auditors is indicative of its performance outside of an auditing context, private sector entities could benefit significantly from using NVCAT to assess their own network security. An exact Return on Investment (ROI) has not yet been measured, but is expected to be quite high as NVCAT is free-to-use.

NVCAT will continue undergoing development in the Center for Enhanced Cybersecurity at GAO. Expected future developments are described in detail in the following section.

VII. Future Work

Due to the wide scope of NVCAT's original concept, there remains ample room for improvement and expansion. Here we discuss some of the future work that will take place at GAO. Broadly, two priorities are increasing the usability and expanding the functionality of NVCAT.

There are three main objectives that, when accomplished, will increase NVCAT's usability. The first objective is to reduce the amount of user input required. This improvement was suggested by GAO official Saar Dagani during NVCAT's evaluation phase [29]. This will simplify the work done by the user, and ensure that reports are generated more quickly. One solution is to update NVCAT to automatically detect the device type and vendor/platform associated with the file(s) being analyzed. This will alleviate the user of the burden of determining the device type and platform prior to running NVCAT. This update could also account for the fact that the user may not have separated the files to be analyzed by device type. Following this update, a user could, for example, provide NVCAT with a folder containing both running configurations and access lists, and NVCAT would provide the desired results.

The second objective towards increasing NVCAT's usability is to make the reports more easily convertible into reportable findings. NVCAT already offers two options for copying a report, including the ability to copy the report data in comma-separated value (CSV) format. The CSV data can be easily pasted into an open Excel spreadsheet, but offering an option for the user to export the report data and save it as an Excel spreadsheet would remove the intermediate step. Because GAO auditors will be reporting their findings in official GAO reports, this would be a necessary and useful feature to implement.

The third and final objective towards increasing NVCAT's usability is to make the application more portable. This will allow any user to run NVCAT with ease, regardless of the operating system that is being used. As a result, the overhead effort associated with installing dependencies will be eliminated or significantly reduced. One solution is to create a Command Line Interface version of NVCAT. This will remove any portability obstacles associated with the tkinter GUI. As of this writing, NVCAT has been tested on macOS devices running Python 3, with the required packages installed appropriately (see requirements.txt in NVCAT GitHub repository [32]).

There are three main objectives that, when accomplished, will expand NVCAT's functionality. The first objective is to implement further parsing logic to support the analysis of devices from firewall and network vendors other than Cisco. GAO officials have indicated that devices from Juniper Networks and Palo Alto Networks are commonly seen during audits [1]. However, support for devices from additional vendors (beyond Cisco, Juniper, and Palo Alto) would be desirable in marketing NVCAT to private sector entities.

The second objective towards expanding NVCAT's functionality is to implement additional parsing logic for the output of other network device commands. For example, parsing the output of the "show version" command on Cisco devices (and the analogous commands on network devices from other vendors) will allow NVCAT to provide reports on patching that has taken place on the device(s) being analyzed.

The third and final objective towards expanding NVCAT's functionality is to offer additional search options for all device types. One solution is to draw from the hardening guides that Cisco has published regarding IOS devices, NX-OS devices, and ASA firewalls [11] [33] [34]. These guides provide a significant number of security recommendations, and NVCAT could be updated to check whether these recommendations are being followed. For example, NVCAT could be updated to check routers for Network Time Protocol (NTP) authentication or secure routing protocols such as Border Gateway Protocol (BGP) and Interior Gateway Protocol (IGP) [11], or to check firewalls for usage of secure protocols, such as using Secure

Copy Protocol instead of File Transfer Protocol or TFTP (Trivial FTP) [34]. For this update, the NVCAT report will also be updated to reflect a new classification of finding. The report will be divided into multiple sections, distinguishing between "important findings" (threats to visibility such as disabled or mis-configured NetFlow/SPAN) [1] and "hardening recommendations".

With these additional features implemented, NVCAT is expected to further improve the efficiency of government information system audits [28].

Acknowledgments

I would like to thank Georgia Tech faculty members Mustaque Ahamad, Nadiya Kostyuk, and Milton Mueller. Their guidance and constructive criticism allowed me to perceive faults and opportunities for improvement that would have otherwise gone unnoticed. I would additionally like to thank my colleague Jayanth Kancherla, whose consistent feedback helped me improve upon this project. Special thanks to GAO's Saar Dagani, who provided significant assistance in the conception and evaluation of this project.

References

[1] Gabriel Siewert, Gary Austin, and Saar Dagani. *Midpoint Informational Meeting*. Oct. 2020.

[2] Cisco. *Introduction to Cisco IOS NetFlow - A Technical Overview*. May 2012. URL: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html.

[3] Shashank Singh. *Catalyst Switched Port Analyzer (SPAN) Configuration Example*. Jan. 2019. URL: https://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/10570-41.html.

[4] MiaRec. *What is port mirroring?* URL: https://www.miarec.com/faq/what-is-port-mirroring.

[5] Cloudflare. *What Is The OSI Model?* URL: https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/.

[6] Cisco. *CLI Book 2: Cisco ASA Series Firewall CLI Configuration Guide, 9.6 - Access Control Lists [Cisco ASA 5500-X Series Firewalls]*. June 2020. URL: https://www.cisco.com/c/en/us/td/docs/security/asa/asa96/configuration/firewall/asa-96-firewall-config/access-acls.html.

[7] SpeedGuide. *Port 445 (tcp/udp)*. URL: https://www.speedguide.net/port.php?port=445.

[8] eTutorials.org. *Access List Overview*. URL: http://etutorials.org/Networking/Router%20firewall%20security/Part%20III%20Nonstateful%20Filtering%20Technologies/Chapter%206.%20Access%20List%20Introduction/Access%20List%20Overview/.

[9] Batfish. *batfish/batfish*. URL: https://github.com/batfish/batfish.

[10] Mikhail Driagunov, Natalia Khodukina, and Nikita Loginov. *Cisco Config Analysis Tool*. Dec. 2018. URL: https://github.com/cisco-config-analysis-tool/ccat.

[11] Shashank Singh. *Cisco Guide to Harden Cisco IOS Devices*. Sept. 2020. URL: https://www.cisco.com/c/en/us/support/docs/ip/access-lists/13608-21.html.

[12] ManageEngine. *Firewall Configuration & Log Management by ManageEngine Firewall Analyzer*. URL: https://www.manageengine.com/products/firewall/features.html.

[13] ManageEngine. URL: https://www.manageengine.com/products/netflow/features.html.

[14] SolarWinds. *Network Configuration Management Tool - NCM Software*. URL: https://www.solarwinds.com/network-configuration-manager.

[15] ManageEngine. *NetFlow Analyzer*. URL: https://store.manageengine.com/netflow/.

[16] Gabriel Siewert and Saar Dagani. *Preliminary Informational Meeting*. Aug. 2020.

[17] U.S. Government Accountability Office. *Information Security: Significant Progress Made, but CDC Needs to Take Further Action to Resolve Control Deficiencies and Improve Its Program*. Dec. 2018. URL: https://www.gao.gov/products/GAO-19-70.

[18] Python Software Foundation. *tkinter - Python interface to Tcl/Tk*. 2020. URL: https://docs.python.org/3/library/tkinter.html.

[19] Python Software Foundation. *sqlite3 - DB-API 2.0 interface for SQLite databases*. URL: https://docs.python.org/3/library/sqlite3.html.

[20] Red Hat. *4.9.3. FIPS 140-2 Compliant Passwords JBoss Enterprise Application Platform 6.3*. URL: https://access.redhat.com/documentation/en-us/jboss_enterprise_application_platform/6.3/html/security_guide/fips_140-2_compliant_passwords.

[21] Oracle. *Using a FIPS 140-2 Enabled System in Oracle® Solaris 11.3*. Oct. 2019. URL: https://docs.oracle.com/cd/E53394_01/html/E54966/fips-refs.html.

[22] SpeedGuide. *Port 23 (tcp/udp)*. URL: https://www.speedguide.net/port.php?port=23.

[23] Mike Chan. *Passwords vs. SSH keys - what's better for authentication?* Mar. 2020. URL: https://www.thorntech.com/2018/12/passwords-vs-ssh/#:~:text=Security%20levels%20of%20passwords%20compared%20to%20private%20keys.&text=And%20unlike%20passwords,%20your%20private,where%20the%20private%20key%20resides..

[24] Tatu Ylonen et al. Oct. 2015. URL: https://nvlpubs.nist.gov/nistpubs/ir/2015/NIST.IR.7966.pdf.

[25] Jeff Forcier. *Welcome to Paramiko's documentation!* 2020. URL: http://docs.paramiko.org/en/stable/.

[26] *Paramiko's SSH and SFTP usage*. July 2020. URL: https://developpaper.com/paramikos-ssh-and-sftp-usage/.

[27] Cisco. URL: https://www.cisco.com/c/en/us/td/docs/routers/nfvis/switch_command/b-nfvis-switch-command-reference/monitor_commands.pdf.

[28] Gabriel Siewert and Saar Dagani. *Final Informational Meeting*. Nov. 2020.

[29] Gabriel Siewert. *NVCAT User Response Form*. Nov. 2020. URL: https://forms.gle/bC5srPvX9TLPZ5568.

[30] Hendrik Müller, Aaron Sedley, and Elizabeth Ferrall-Nunge. *Ways of Knowing in HCI*. Springer-Verlag, 2014, pp. 229–266.

[31] Gabriel Siewert. *NVCAT User Response Form (v2)*. Nov. 2020. URL: https://forms.gle/qcs6qxDePhh6znHS8.

[32] Gabriel Siewert. *NVCAT*. Nov. 2020. URL: https://github.com/siewertg/NVCAT.

[33] Cisco. *Cisco Security Threat and Vulnerability Intelligence*. Nov. 2014. URL: https://tools.cisco.com/security/center/resources/securing_nx_os.html.

[34] Srinivasa Munagala and Dinkar Sharma. *Cisco Guide to Harden Cisco ASA Firewall*. Oct. 2017. URL: https://www.cisco.com/c/en/us/support/docs/security/asa-5500-x-series-next-generation-firewalls/200150-Cisco-Guide-to-Harden-Cisco-ASA-Firewall.html.

Appendix

The GitHub repository containing all NVCAT source code can be found at https://github.com/siewertg/NVCAT.