Automation of Server Security Assessment

Varun CP
REVA Academy for Corporate Excellence, REVA University, Bengaluru, India
varun.cs02@reva.edu.in

Rashmi Agarwal
REVA Academy for Corporate Excellence, REVA University, Bengaluru, India
rashmi.agarwal@reva.edu.in

***Abstract*—While system hardening concepts are general, one of the leading causes of the breaches is human error in the misconfiguration. Depending on the type of hardening, different tools and techniques are used. The whole lifespan of technology, from initial installation through setup, maintenance, and support, to end-of-life decommissioning, necessitates system hardening. Additionally, mandated by regulations like PCI DSS (Payment Card Industry Data Security Standard.) and HIPAA (Health Insurance Portability and Accountability Act), systems hardening is something that cyber insurers are increasingly requesting. This paper explains how to automate server security assessments using an ansible agentless framework and utilize them to continue security audits and compliance evaluations throughout risk assessments. The technique and ideas discussed in this paper are more effective when the server environment is undergoing continual change.**

Keywords—Linux Server, Automation, Security Assessment, Compliance, Server hardening

I. INTRODUCTION

Application servers are frequently constructed using the Linux operating system. The Ubuntu server is one of the most extensively used distributions. The Ubuntu server skillfully juggles complexity with usability. Ubuntu has all the tools required to build a server operating system and is dependable, speedy, and well-equipped. Because the operating system houses programs and user data, its security needs to be considered. The Data Breach Investigations Report (DBIR) for 2022, issued by Verizon [1] stated that 23,896 cyber incidents and 5,212 cases caused data leaks. Hardening the operating system is one way to increase security. Hardening is not a one-time activity; In Fig. 1 provides information on how error continues to be a dominant trend and is responsible for 13% of the breaches [1], It is an ongoing task to mitigate the risk of performing high-quality computing. Hardening of the operating system should support the high integrity, reliability, availability, privacy, scalability, and confidentiality at the lowest level of risk to achieve the highest level of objective from the organization's critical IT (Information Technology) infrastructure. To thrive in today's competitive environment, businesses must continually alter, grow, and reinvent themselves. The pressure for server administrators to keep up with business as usual is tremendous. Hybrid environments are referred to by a variety of words, including private, public, on-premises, off-premises, managed, self-service, physical, virtual, and others. By integrating and providing a seamless experience across numerous viewpoints of infrastructure as a service, the dynamic cloud, based on open standards, enables the IT manager to answer the quickly changing demands of the business.

The firm puts ongoing demands on administrators to deploy servers fast. With the dynamic cloud, these administrators can serve their users with speed, efficiency, and access to services they have never had before. However, this comes with the danger of having little or no room to examine whether the deployed infrastructure is set up or maintained to comply.

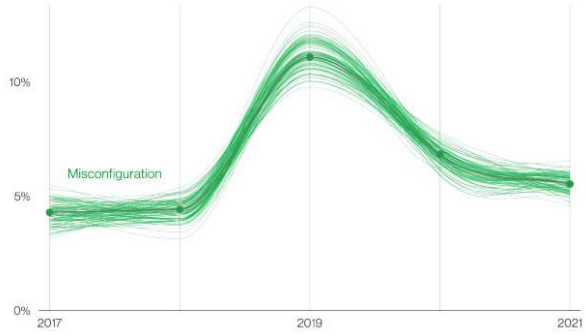

Fig. 1: Misconfiguration over time [1].

Resources continuously change in the server infrastructure environment, mistakes are made, and things stop working as intended. Periodic, point-in-time audits do not always provide the whole story, and they frequently exclude vital information that management needs to make important decisions. In contemporary server infrastructure, the manual security assessment that uses screenshots, dated reports, samples of servers' point-in-time configurations, etc., will not be effective. Without automation, it is almost difficult for an audit expert to execute a technically correct security evaluation on a server infrastructure since manual security assessments are inefficient and do not give enough data on the operating efficacy of security measures

II. LITERATURE REVIEW

This paper reviews the literature to understand how the hardening configuration is achieved using various purposes and approaches for automated assessment of the server environment.

Server security and upkeep would have to be done manually, which would require a lot of administrative work. Automated security techniques are necessary to identify, stop, and defend servers from intruders. Security policies are crucial to keep network security from being compromised, impacting server security [2]. One of the operating systems that can accommodate the accelerating advancement of internet technology is Linux. Security must be considered in addition to process speed [3].

The author in [4] has determined a novel approach for detecting malware in Linux machines as they are becoming a severe threat to private data, including computer resources. The paper describes the use of machine learning to identify malicious executable linkable files. Establishing and monitoring more than 130 security rules constitute the

challenging work of information security management as many controls as feasible must be automated to increase process efficiency. The number of commands that can be automated using the ISO 27001 and NIST SP800-53 standards. Additionally, the automation capabilities of the controls are listed in the Consensus Audit Guidelines. Finally, to improve the effectiveness of information security management, to give a brief review of security apps that allow automation in the operation of information security controls [5].

The Linux operating system offers a built-in security configuration and management mechanism to handle the security needs of a business setting. The Linux system needs to be configured by the system administrator to increase the security assurance of the system, and information security auditors need to verify the configuration of the Linux system by audit standards to ensure the enterprise has a security system in place. Securing the production system from malicious assaults is an onerous responsibility for a system administrator [6].

Security firms are now experiencing difficulties and current methods for automating security inspections. A more thorough analysis of the Open Vulnerability and Assessment Language (OVAL) and Extensible Configuration Checklist Description Format (XCCDF) languages is provided [7]; the Chef Inspec application was used to successfully add 232 controls on the CIS Benchmark [8]. The study's findings showed that 27 controls were omitted, 118 controls failed, and 87 controls were successful [3].

A baseline configuration approach called "security baseline hardening" tries to make the operating system more resilient while reducing the likelihood and severity of breach situations. The term "Security Baseline Audit" refers to a series of operations required for security baseline hardening per standard best practices to keep the system under control. The resources above in a large production environment can be increased by the size of the system in the production environment [9].

The planning and execution of an automated hardening process for a cleanly installed Linux web server. The article [10] discusses the study of the configuration's emphasis, scope, and cybersecurity. The system administrator needs to configure the Linux system to get more security assurance from the system. The information security assessor must check the Linux system configuration per compliance standards to ensure the security system is in place in the enterprise. It is a difficult task for a system administrator to secure the production system from malicious attacks [11].

III. Design Of Proposed Solution

A security auditing program called Lynis has the drawback of requiring administrators to install it on each server and run it before they can see the findings, which come in the form of log output that cannot be easily shared and kept. The solution proposed is to install Lynis on all the servers, and execute and convert the generated console report to HTML (Hypertext Markup Language) format, this is achieved by ansible, which is an agentless automation tool that is installed on a single host, is referred to as the control node.

A control node is nothing but an endpoint/server where ansible is installed and ansible playbooks are executed from the control node, Ansible can manage an entire fleet of machines and other devices (referred to as managed nodes) remotely with SSH, PowerShell remoting, playbooks and numerous different modes of transport, all from a simple command-line interface with no databases or daemons required.

The Environment for the assessment using the module and Ansible has been created in a virtualized environment, and the Lynis module is installed and executed to generate an automated assessment of the servers and is made available as a report.

Ansible deploys the Lynis module on the ansible nodes; the module is used to gather the server configuration posture and compare it to a set of predetermined rulesets. The resource configuration fetcher is an essential part of the framework. The fundamental foundation of the rule system is the analysis of those configurations, and then data is then analyzed in correlation to ensure incoherent information is visualized in the dashboard of the solution, which is readily available to the user.

The solution architecture as shown in Fig. 2 contains the following components:

1. Control Node: Ansible is installed on the Ansible Control node, which also acts as the foundation for the automation framework. Playbooks and other Ansible-related items are stored here, and also manage the requirements for the modules described above.

2. Playbook: It oversees establishing the connection and installing the prerequisites for the automation framework to function properly. It stores lists of actions that automatically run against the controlled nodes.

3. Lynis: It evaluates server infrastructures' security posture, gathers configuration information for manual inspection, and identifies risk areas using the scripts. Lynis immediately gives a clear image of the attack surface, saving the user from navigating through dozens of pages on the configuration interfaces.

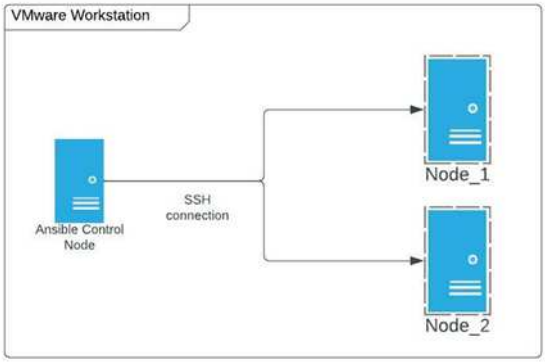

Fig. 2. The architecture of the automation server security assessment.

The Implementation of the automated server security assessment using an ansible framework, along with ansible Lynis and ansi2html is used to generate the report. Lynis is responsible for the security assessment, and Ansible is for automating the installation of the required modules and initiating the assessment on the target servers, the steps involved in deploying the same are:

1. Installation of Ubuntu VM on VMware workstation.
2. Installation of Ansible on the Ubuntu VM.
3. Ansible playbook for automating the server security assessment as shown in Fig. 3, it consists of host details to where the list of task should be executed.
4. Configuring key-based authentication which enables enable password-less login for all the communication from the ansible control node to servers, which includes:
    a. Creating the key pair using the ssh-keygen command.
    b. Copy and install the public key using the ssh-copy-id command.

```
---
- hosts: ansible_nodes
  become: true
  tasks:
   - name: adding repo key
     apt_key:
       keyserver: keyserver.ubuntu.com
       id: 013baa07180c50a7101097ef9de922f1c2fde6c4
       state: present
   - name: prequisites
     package:
       name:
        - apt-transport-https
        - lynis
        - kbtin
       update_cache: yes
       state: present
   - name: ansi2html installation
     pip:
       name: ansi2html
   - name: adding repo
     apt_repository:
       repo: "deb https://packages.cisofy.com/community/lynis/deb/ stable main"
       state: present
   - name: Starting security assessment of the server.
     shell: lynis audit system | ansi2html -la > /tmp/assessment.html
   - name: downloading report locally
     fetch:
      src: /tmp/assessment.html
      dest: ./{{ inventory_hostname }}-assessment-report-{{ansible_date_time.dat
e }}.html
       flat: yes
   - name: report location
     debug:
      msg: "Report can be found at ./{{ inventory_hostname }}-assessment-report-
{{ansible_date_time.date }}.html"
```

Fig. 3. The Ansible playbook for server security assessment.

IV. Testing and Validation

Fig. 4 shows the execution of the playbook for the security assessment, which is executed without any errors, but one of the dependencies that were observed during the execution of the playbooks in the new control environment is whether the IP should be static, or the DNS should be configured in the server environment to resolve the domain names. The tasks listed are successfully executed and the report is generated in the current working folder as shown in Fig. 5.

```
varun@ansiblecontroller:~/Projects/capstone_2$ ansible-playbook server_assessment.yml

PLAY [ansible_nodes] ***********************************************************

TASK [Gathering Facts] *********************************************************
ok: [192.168.10.13]
ok: [192.168.10.12]

TASK [adding repo key] *********************************************************
ok: [192.168.10.13]
ok: [192.168.10.12]

TASK [prequisites] *************************************************************
ok: [192.168.10.13]
ok: [192.168.10.12]

TASK [ansi2html installation] **************************************************
ok: [192.168.10.12]
ok: [192.168.10.13]

TASK [adding repo] *************************************************************
ok: [192.168.10.12]
ok: [192.168.10.13]

TASK [Starting security assessment of the server.] *****************************
changed: [192.168.10.13]
changed: [192.168.10.12]

TASK [downloading report locally] **********************************************
changed: [192.168.10.13]
changed: [192.168.10.12]
```

Fig. 4. Executing the Ansible Playbook

```
ler:~/Projects/capstone_2$ ls -la

varun  4096 Aug 25 02:18 .
varun  4096 Jul 29 03:07 ..
varun 42587 Aug  7 08:27 192.168.10.12-assessment-report-2022-08-07.html
varun 42633 Aug 25 02:18 192.168.10.12-assessment-report-2022-08-25.html
varun 42587 Aug  7 08:27 192.168.10.13-assessment-report-2022-08-07.html
varun 42633 Aug 25 02:18 192.168.10.13-assessment-report-2022-08-25.html
varun   104 Aug  7 06:28 ansible.cfg
```

Fig. 5. Reports generated

V. Analysis and Results

The goal of the research is to evaluate the resources that are constantly changing in the server computing environment to execute an automated security evaluation of the servers. Since manual security assessments are ineffective and do not provide sufficient information on the operational efficacy of security measures, automation ensures that real-time point information is available for the audit expert to carry out a technically accurate security evaluation on a server system. The created report provided the security configuration of the servers in several areas, as illustrated in Fig. 6. It also gives comprehensive information about the complete server setup. The dashboard created for the security evaluation of the Ubuntu servers is depicted in Fig. 7.

System tools	system binaries
Boot and services	boot loaders, startup services
File systems	mount points, /tmp files, root file system
Kernel	run level, loaded modules, kernel configuration, core dumps
Memory and processes	zombie processes, IO waiting processes
NFS	
Ports and packages	vulnerable/upgradable packages, security repository
Shells	
Software: name services	DNS search domain, BIND
Storage	usb-storage, firewire ohci
Users, groups and authentication	group IDs, sudoers, PAM configuration, password aging, default m
Networking	nameservers, promiscuous interfaces, connections
Printers and spools	cups configuration
Software: e-mail and messaging	
Software: firewalls	iptables, pf
Software: webserver	Apache, nginx
SSH support	SSH configuration
SNMP support	
Databases	MySQL root password
LDAP services	
Software: php	php options
Squid support	
Logging and files	Syslog daemon, log directories
Insecure services	inetd
Banners and identification	
Scheduled tasks	crontab/cronjob, atd
Accounting	sysstat data, auditd
Time and synchronization	ntp daemon
Cryptography	SSL certificate expiration
Virtualization	
Security frameworks	AppArmor, SELinux, security status
Software	file integrity
Software	malware scanners

Fig. 6. Hardening Data points

Fig. 7. Report Summary

VI. Conclusion

The use of Lynis and the Ansible automation platform to execute a security assessment of each server, regardless of how many there are, and without having to worry about dependencies. The report generated is centrally stored in HTML format. By evaluating the configuration and deviation that is indicated, visualization makes it simple to understand the risk posture of the server infrastructure with ease. Additionally, this method might be further modified depending on the next viewpoint to benefit the administrative user where the environment has different types of the operating system.

VII. References

[1] Verizon, “DBIR Data Breach Investigations Report,” 2008.

[2] S. Patra, N. C. Naveen, and O. Prabhakar, “An automated approach for mitigating server security issues,” in 2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings, Jan. 2017, pp. 1075–1079. doi: 10.1109/RTEICT.2016.7807996.

[3] W. K. Sedano and M. Salman, “Auditing Linux Operating System with Center for Internet Security (CIS) Standard,” in 2021 International Conference on Information Technology, ICIT 2021 - Proceedings, Jul. 2021, pp. 466–471. doi: 10.1109/ICIT52682.2021.9491663.

[4] K. A. Asmitha and P. Vinod, “A machine learning approach for Linux malware detection,” Proceedings of the 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques, ICICT 2014, pp. 825–830, 2014, doi: 10.1109/ICICICT.2014.6781387.

[5] R. Montesino and S. Fenz, “Information Security Automation: How Far Can We Go?,” in 2011 Sixth International Conference on Availability, Reliability and Security, 2011, pp. 280–285. doi: 10.1109/ARES.2011.48.

[6] Isaca, “Auditing Linux/Unix Server Operating Systems.” [Online]. Available: www.isaca.

[7] G. Koschorreck, “Automated Audit of Compliance and Security Controls,” in 2011 Sixth International Conference on IT Security Incident Management and IT Forensics, 2011, pp. 137–148. doi: 10.1109/IMF.2011.12.

[8] “CIS_Ubuntu_Linux_20.04_LTS_STIG_Benchmark_v1.0.0”.

[9] H. Dabthong, M. Warasart, P. Duma, P. Rakdej, N. Majaroen, and W. Lilakiatsakun, “Low-Cost Automated OS Security Audit Platform Using Robot Framework,” in 2021 Research, Invention, and Innovation Congress: Innovation Electricals and Electronics (RI2C), 2021, pp. 31–34. doi: 10.1109/RI2C51727.2021.9559826.

[10] M. Olenčin and J. Perháč, “Automated Hardening of a Linux Web Server.”

[11] HUNTSMAN | TIER-3 PTY LTD, “Cyber security audit challenges in 2020,” Chatswood NSW 2067, 2020. [Online]. Available: www.huntsmansecurity.com/blog/audit-compliance-risk-cyber-security-