TOC

# Penetration Test Report of Findings

THM | Kenobi

2024-11-06

Table of Contents

[Penetration Test Report of Findings 1](#_5wmprsfeq3j4)

[Executive Summary 3](#_2n702dgfvhay)

[Overview 3](#_nv0jnwcdhwfe)

[Risk Assessment 3](#_9oxgflwm7o34)

[Recommendations 3](#_1xer5jcbpzii)

[Engagement Overview 4](#_bskl2nuky82a)

[Scope 4](#_xbr8z6b33mr6)

[Methodology 4](#_iuuerrhi2v97)

[Compromise Walkthrough 5](#_uejbqvpywztp)

[Reconnaissance 5](#_m46uf26ucvyi)

[Initial Access 11](#_9d1mpo8xtwnz)

[Privilege Escalation 12](#_r0ntcram0rko)

[Engagement Results 15](#_i1lf5ls9nrd)

[Findings 15](#_y7qv59hpnouh)

[Remediations 17](#_9idzghrgv3lt)

[Vulnerability 01 (High) 17](#_86p5nblzt7av)

[Vulnerability 02 (Critical) 17](#_k6k6qezcxiar)

[Vulnerability 03 (Critical) 17](#_9ozb0bt59ny)

# 

# Executive Summary

## Overview

The machine “Kenobi” contains multiple vulnerabilities which can be exploited in conjunction to create a critical risk to system integrity (root user access). A threat actor with root access to the file transfer server could create an exposure of sensitive / proprietary data or lead to further network exploitation and create a ransomware attack.

## Risk Metrics

**Vulnerabilities**

* Critical (2)
* High (1)

**Exploitability -** High

**Impact -** Critical

**Overall Risk Assessment -** Critical

## Recommendations

* Implement a regular system patching / updating cycle to address common vulnerabilities in systems.
* Implement system hardening best-practices.
* Implement strong authentication policies for network services.

# Engagement Overview

## Scope

**Hosts**

* 10.10.34.142

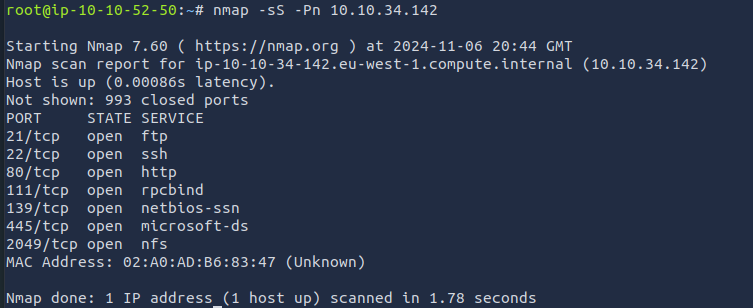
## Methodology

The engagement will follow a simple CTF-style gray-box assessment. All TTP’s and tools are permitted for this engagement. Goal for this engagement is root privileges over the target Linux machine. Author has provided information regarding vulnerable components of the machine: Samba shares, vulnerable FTP Service, and insecure binaries with SUID bit set.

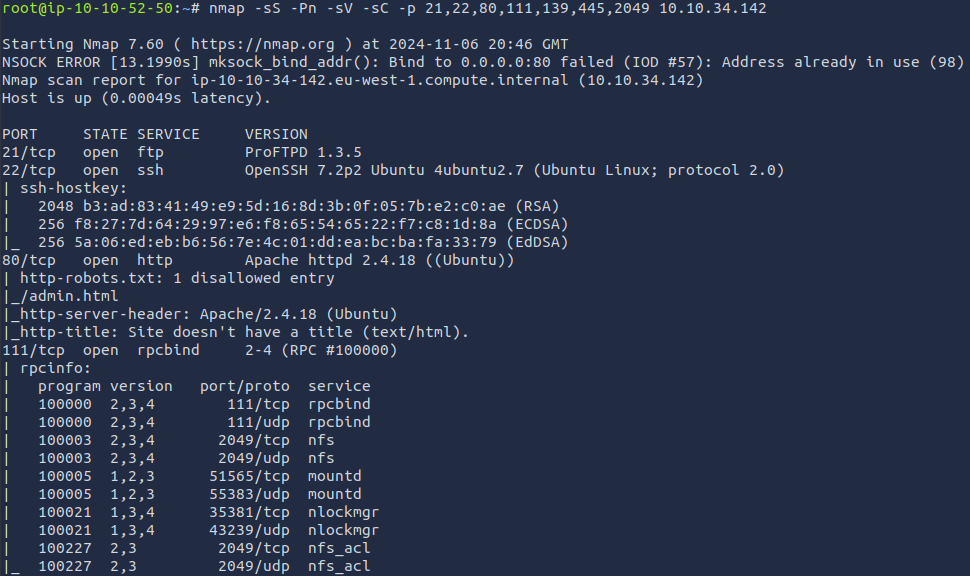
# Compromise Walkthrough

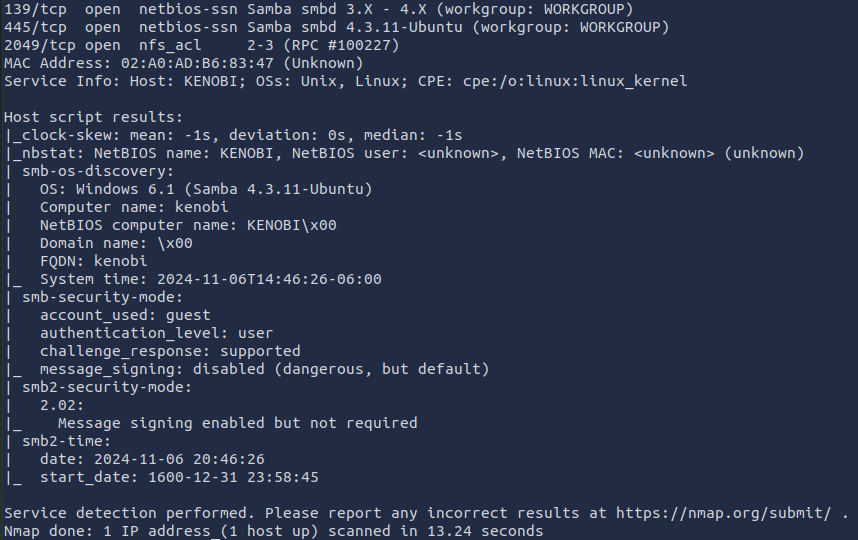
## Reconnaissance

Initial Nmap scan to enumerate open ports:

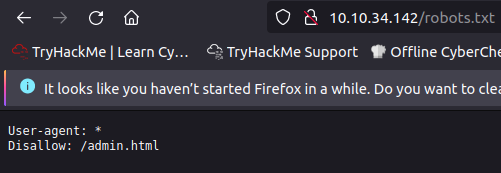


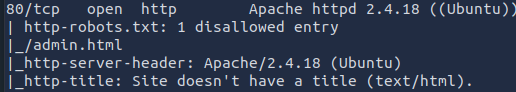
Deeper Nmap scan to enumerate services and vulnerabilities:





Started with investigating the web server on port 80 running Apache 2.4.18. By checking robots.txt as well as the nmap scan output a potentially sensitive directory (/admin.html) is revealed:

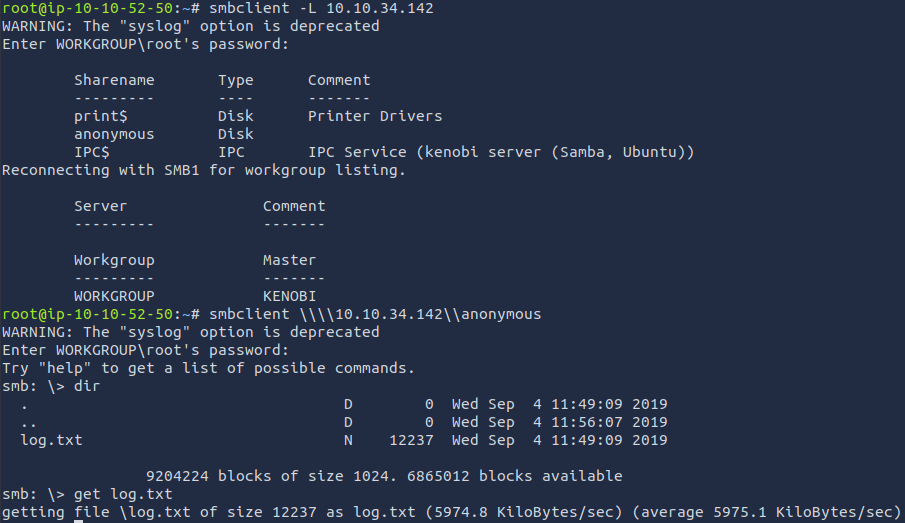


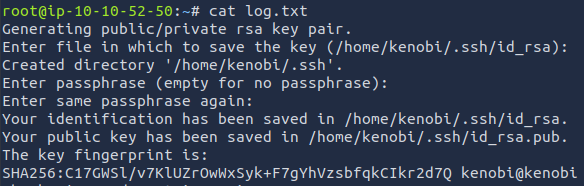


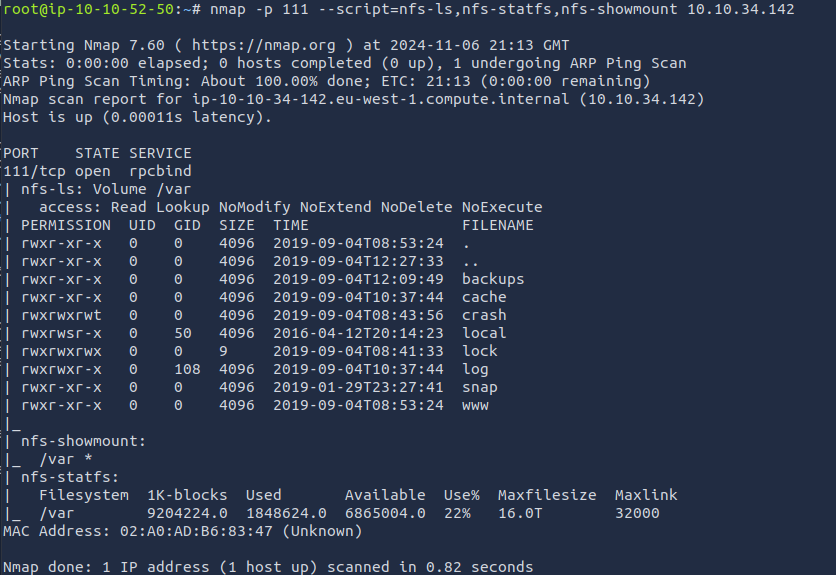
This directory does not lead to any useful info.

Checked for any other directories that may be hidden on the web server with dirbuster, which returned no further results.

Investigated the Samba shares over port 139/445, found a share “anonymous” which contained a file “log.txt.” Downloaded the txt file to attacker workstation for analysis:

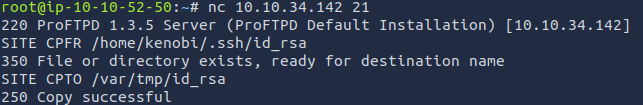


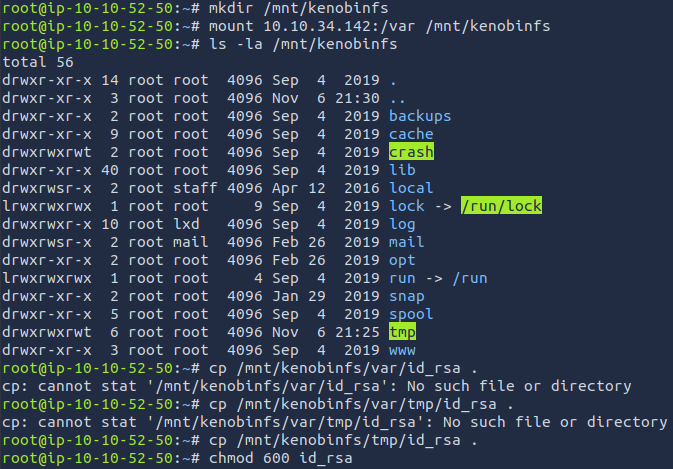
Log.txt contains the configuration details for the ProFTPD server running on the target machine, and contains information about the FTP server and the location of the SSH key generated for user “Kenobi.”:  
  
Used Nmap to enumerate the network file system service running on port 111:



A vulnerability in ProFTPD 1.3.5 allows an unauthenticated user to copy any file to any destination in the vulnerable system. (<https://www.rapid7.com/db/modules/exploit/unix/ftp/proftpd_modcopy_exec/>)

By leveraging this vulnerability to move the private RSA SSH key, whose file location was revealed by the Samba share, can be moved to network file share and copied to the attacker’s machine:



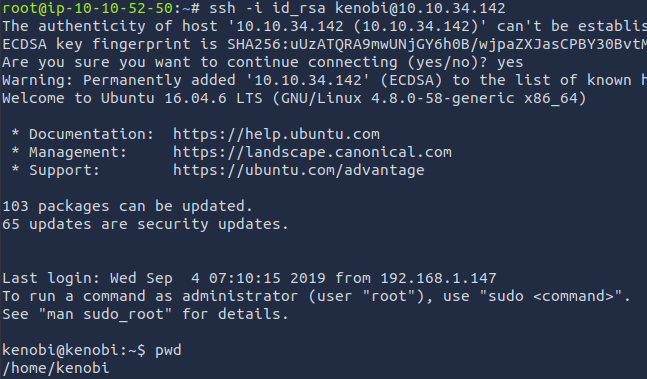


Initial Access can be achieved by connecting via SSH using this RSA key.

## 

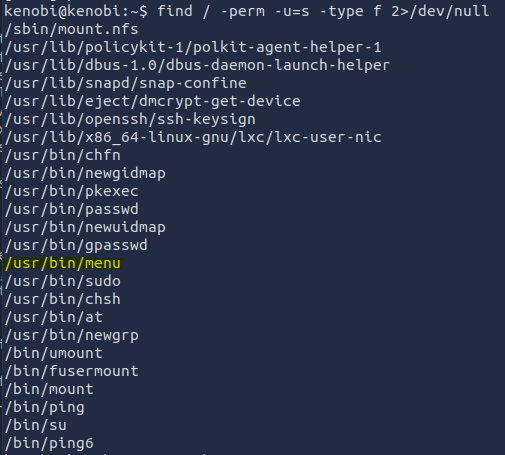
## Initial Access

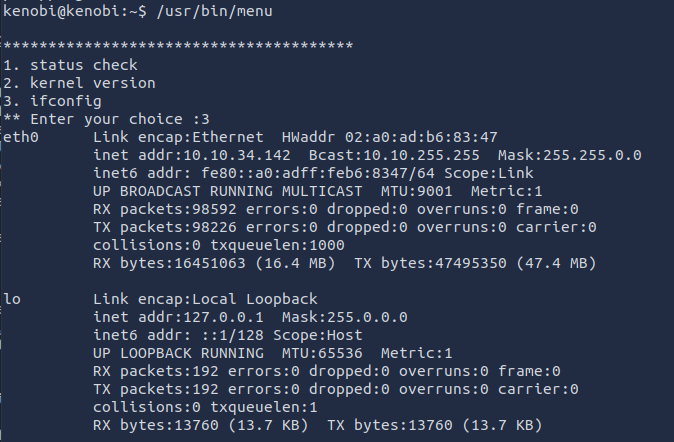
A vulnerability in the ProFTPD version 1.3.5 as well as insecure Samba and Network File Share configurations discovered in the reconnaissance phase allowed exfiltration of a local user’s private RSA key which can be used to connect to the target machine via SSH:



/home/kenobi/user.txt = d0b0f3f53b6caa532a83915e19224899

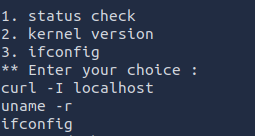
## Privilege Escalation

Listing all files with the SUID bit set reveals one out-of-place binary:  


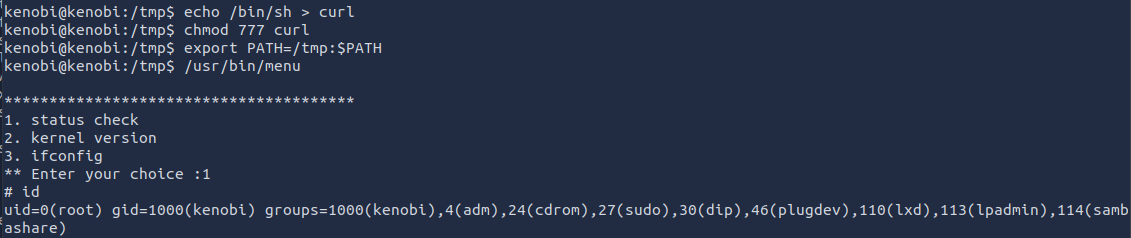
This binary seems to be a simple debugging tool:  


Checking the binary with Strings reveals that the commands are not called via the full file path:





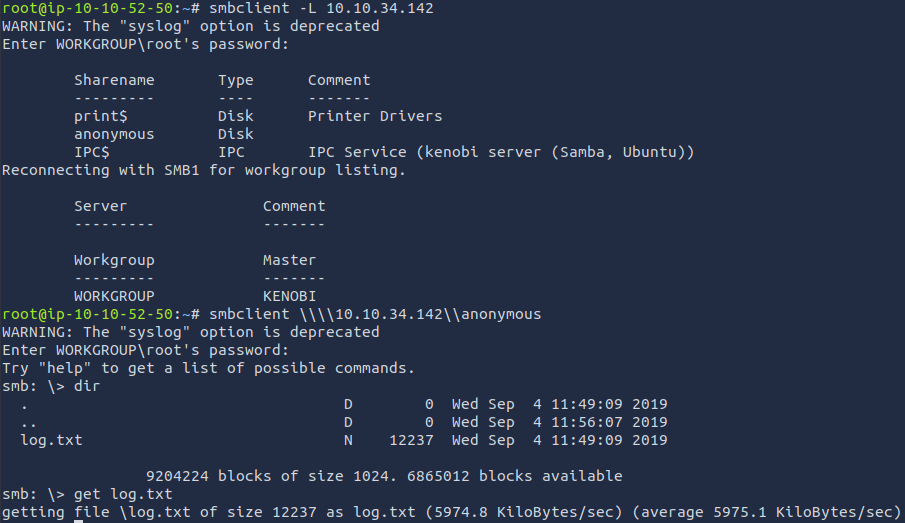
By creating a new file called “curl” which opens /bin/sh, and manipulating the $PATH variable to check the directory that this new file named curl exists in, a shell with root permissions can be accessed since /usr/bin/menu runs as root:

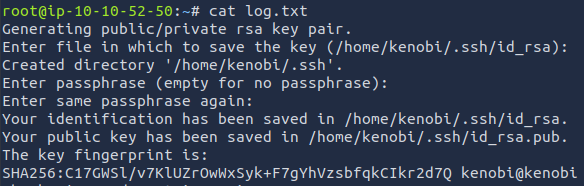


/root/Root.txt - 177b3cd8562289f37382721c28381f02

# Engagement Results

## Findings

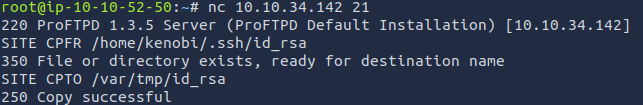
**Vulnerability (1):** ProFTPD server configuration file stored in Samba share that allows anonymous login contains file location of “kenobi” user’s private RSA key. (CWE-200, CWE-532).  
  
**Proof of Exploitation:**



## 

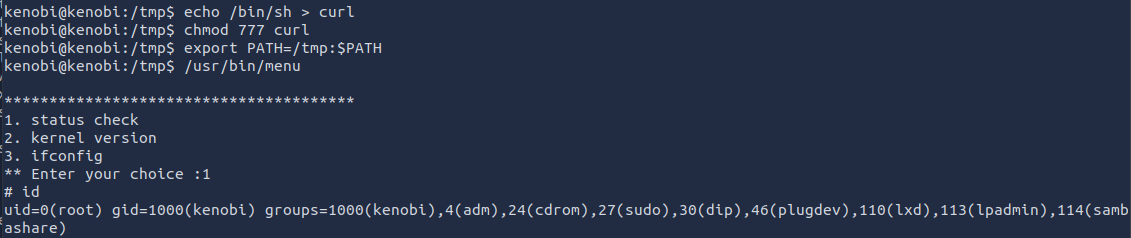
**Vulnerability (2):** Out-of-date ProFTPD server contains RCE vulnerability and allows an unauthenticated attacker to copy files from anywhere to anywhere on the file system. (CVE-2015-3306).

**Proof of Exploitation:**

****

**Vulnerability (3):** Vulnerable binary with SUID bit set allows for privilege escalation to root via path variable manipulation. (CWE-269).

**Proof of Exploitation:**



## 

# Remediations

## 

## Vulnerability 01 (High)

* Restrict/disable anonymous login for Samba shares to avoid exposing sensitive data to unauthenticated users.
* Enforce access control and strong authentication mechanisms for users accessing Samba shares containing sensitive data.
* Enforce access control on directories containing sensitive data such as private RSA keys.

## Vulnerability 02 (Critical)

* Update the ProFTPD service to the latest version to address major vulnerabilities.
* Restrict / disable unauthenticated access to the ProFTPD service and implement strong authentication mechanisms and/or policies.

## Vulnerability 03 (Critical)

* Remove the SUID bit on the vulnerabile binary to follow best-practice of least privilege.
* Configure the binary to use absolute paths. The file is vulnerable to a path variable manipulation exploit because it does not specify the full file path of the binaries it attempts to run.
* Set secure\_path in sudoers file to restrict file locations of the $PATH variable when a binary is executed with sudo.

# 