**Milestone 3 Instructions**

*By Drs.*[*Anthony Vance*](https://anthonyvance.com/)

The objective of Milestone 3 is to revise your security assessment report according to the feedback received from the instructor. You will no longer have access to the server, and no additional findings from the server will be scored for points.

The below instructions are repeated from Milestone 2.

**Objectives**

Your objectives are threefold:

1. Document vulnerabilities that you are able to successfully exploit on the server. Describe in detail what you did and what level of access you were able to obtain. If you obtain a user account with limited privileges, document whether you were able to escalate the privileges to root. Document each exploit that you are able to successfully launch.
2. Document potentially sensitive information that you are able to obtain from the server. These could include user files or web, database, or other server files.
3. For both 1 and 2 above, suggest ways that vulnerabilities exploited or sensitive information obtained could be protected.

The scope of your project is restricted to the computer belonging to the IP address communicated to you via Slack.

The server you are to evaluate is running on a private network that you can only get access to if you connect Kali to a VPN. Download client.conf to Kali from Canvas (under the “Files” page). Open a separate Terminal session, and run openvpn client.conf. Leave this terminal running for as long as you need to connect to the Milestone 2 VM. Running this will give you an ip address on VPN in the 10.8. network space.

**Use this new ip address as your LHOST whenever needed, not your 192. one.**

You can view your ip address by running ifconfig. You will be able to connect to the private 172.32.0.0/16 address of your target server even though it is on a different subnet, because the VPN server passes traffic through for you.

**Written Report Deliverable**

You should write the report for a managerial audience, one that isn’t versed in information security concepts. In other words, you need to explain the concepts in terms that can be easily understood by managers without technical experience. If you use technical or unfamiliar terms, include a glossary of the terms used.

There is no length requirement for the report, but your report must not exceed 20 pages (not including appendices).

In writing your report, organize for impact. This means you should discuss the most serious vulnerabilities first. Further, in your description, start by describing macro-level issues and then discuss micro-level details. This practice makes it easier for a readers to quickly process your report.

You can think of this process like a pyramid, where at the top you have the one-page executive summary of your findings, and each successive section provides more granular detail. At the end of the main body of your report, the “supporting details” section should have sufficient details on how to replicate the exploits you found, including step-by-step commands run in Metasploit or other tools. This way, a manager can quickly get a sense of the report by reading the first page and then can choose to continue reading to get lower-level details.

Continuing the pyramid analogy, appendices are at the very base. Appendices are for very technical information that would bog down the report if included in the main body. For example, a Nessus report or detailed output from NMAP do not belong in the report because the information is too technical for a managerial reader to process. Also, they tend to be lengthy and would interrupt the flow of your report. Instead, refer the reader to the appendices for very technical and lengthy information. (I am not interested in seeing a Nessus report. Don’t give me one.)

Finally, whenever you show a command or output from a command in the main body of your report, use excerpts or highlighting to point out the most relevant information, and explain what you show with accompanying text. Imagine you are writing to a manager or executive who doesn’t understand security and needs you to clearly explain your findings and their implications.

Writing technical material for a managerial audience is crucial skill for information systems practitioners and managers alike – especially in information security.

**Rubric**

Your report will be graded using the following rubric:

|  |  |
| --- | --- |
| 5% | One page executive summary that highlights the most important findings of your report. |
| 5% | Description of the scope of the project, objectives, and your authorization to perform the assessment (i.e., my instructions to you in this document and in your email). |
| 5% | Information about the server examined (OS, user accounts, applications installed, databases stored). |
| 25% | High-level description of vulnerabilities successfully exploited and sensitive data obtained. |
| 30% | Supporting details of successful exploits and sensitive data obtained. The detail should be sufficient for another person to replicate the results of your main findings using your report. |
| 25% | Explanation for how to protect against vulnerabilities successfully exploited and sensitive data obtained. Provide concrete steps. |
| 5% | Clarity of writing for a managerial audience. |

Use [this report template](https://anthonyvance.com/security-assignments/Midterm_template.docx) to create your report as a PDF file. For submission, have one person on your team submit the report on Canvas.

I look forward to reading your report. Let me know if you have any questions.

**Tips – General**

* Did your scan show that the server is running something on port 80? It’s probably a web page! Try browsing to it by using Kali’s firefox, and put your server’s ip address into the address bar.
* You will want to read up on using the following tools:
  + scp - one way to copy files from one computer to another, including from your server to Kali. You could also use a meterpreter shell to download files if you have one.

Example scp code (from Kali):

scp Milestone2-server-username@Milestone2-server-ip-address:/full/path/to/file/on/Milestone2/server .

That will open an ssh connection to the Milestone 2 server as the specified user, and copy a file (that the user must have permission to read!) from the specified path down to the current directory (. means ‘current directory that I’m in on Kali’).

Note that this must be run on Kali. Meaning it cannot be run from within an exploited shell on the Milestone 2 server or from within an ssh connection to the Milestone 2 server. You cannot scp a file *from* the Milestone 2 server *to* Kali, because you cannot open an ssh session from the Milestone 2 server to Kali, because there is no ssh server listening on Kali. (If there were, you all could ssh into each others’ Kali VMs while connected to the VPN, and we can’t have that!).

* + ssh - for logging into remote servers
  + sudo (including sudo -l)
  + id
  + hydra to crack ssh logins
* hyrda can try to bruteforce ssh logins. It has some nice flags for that task. Read the documentation for hydra’s -eflag. For example, to try the reverse of a username of a password, you would pass -e r. You can pass multiple values for -e, like it shows in the documentation.

**Tips – Password Cracking**

* hashcat expects hashes to be fed to it in a certain format. See [here](https://hashcat.net/wiki/doku.php?id=example_hashes) for guidance for that.
* Remember that you need to tell hashcat what type of hash you are trying to crack (with the -m flag). It can sometimes be tricky to know what kind of hash you are dealing with. Try installing and using the package hash-identifier.
* In your experience with hashcat, you passed it a hash, and then if it cracked it, it would by default match the plaintext with the hash. For example, if I had:
* user: foobar
* password: dogman
* hash: a8uf33kljufd88

Then I could ignore the username, and feed the following to hashcat:

a8uf33kljufd88

And if it cracked it, it would output:

a8uf33kljufd88:dogman

However, if you have several hashes you are trying to crack at once, it is convenient if hashcat also associates a hash with a username. You can do that by passing the username:hash to hashcat as follows:

foobar:a8uf33kljufd88

… and also set the --username flag in your hashcat call, so that hashcat knows that you are feeding in a prepended username.

If you do this, hashcat will output the crack thusly:

foobar:a8uf33kljufd88:dogman

Remember that you can view cracked passwords that are saved in hashcat’s potfile by using the --showcommand. e.g.:

hashcat --show a8uf33kljufd88

* If you want to crack usernames and passwords at the same time, you can ‘unshadow’ the files first. This puts the usernames and passwords into the same file.

Example usage:

unshadow [passwd file] [shadow file] > myunshadowed\_file

* Just like in the password-cracking lab, if you want to take a crack at hashes in /etc/shadow, you need to put them into a format that hashcat can understand. Read man shadow, and then read man crypt, to understand how to interpret the values in /etc/shadow. man crypt will also help you figure out which hash type is being used.
  + You have to manually edit the unshadowed file and remove everything except for the username and the hash. See [here](https://samsclass.info/123/proj10/p12-hashcat.htm) (except, you can leave in the usernames, if you pass the --username flag)