IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

REVERSE SHELL USING PYTHON: MULTI-CLIENT APPROACH

Jaya Prakash Veganti, Venkata Sai Saka, Bharath Sakinala
Department of Electronics and Communication Engineering,
Sreenidhi Institute of Science and Technology, Hyderabad

ABSTRACT

Now-a-days, almost every house in the world contains at least one device which is connected to the internet. This makes it easy for attackers to target anyone from anywhere easily if he knows the technology. Attackers can have many ways of attacking the target system and gaining access to the remote systems. One of the ways is creating a remote shell. In this project, we implemented a traditional method of gaining access of remote systems by Remote Shell in a new way using Python. Gaining shell is nothing but gaining access to the terminal of the target where we can control any process on the system. Here, Target connects to the Attacker where we created two programs, one for the Attacker (Server code) and one for the Target (Client code). Since the target system connects to the hacker, it is called as The Reverse Shell. Conventional shells cannot handle more than one target. But in this project, we configured the server in such a way that it can connect to multiple targets and have their shells running in the background simultaneously. Attacker can decide which target to work on and gain access.

KEYWORDS

Reverse Shell, Cybersecurity, Socket Programming, Vulnerability, Hacking, Python.

INTRODUCTION

In current world, security in cyber space has become one of the major risks of IT industry. Many people do not even consider to apply security to their systems both in home and commercial environments. They are not aware of attacks happening around them. They are in a misperception of why would a hacker target me. This

lack of security awareness in people gives a huge advantage to hackers who try to do malicious actions and steal valuable information. So, it has become very important to evaluate the efficiency of our security being provided frequently. We need to know how hackers can target our systems in order to prevent them from doing so. For that we need to analyze our systems first for any potential vulnerabilities to evaluate security level of our cyber environment. This is called Penetration Testing.

Remote Shells is one of the major security vulnerabilities the penetration testers try to examine in the testing a system. Since the hackers can do anything when they get remote shell, penetration testers know how important it is to keep an eye out on the system to check if it is vulnerable to remote shell. But reverse shell is something that they cannot examine. Any firewall checks for the in-bound traffic (the traffic coming into the network or system), but not the out-bound traffic (the traffic going out of the network or system). Since the connection between the attacker and the target is initiated from the target, the network traffic will be out-bound. So, any firewall, Intrusion Detection System (IDS), Intrusion Prevention System (IPS) does not interfere the connection between the attacker and the target. This is the reason why attackers mostly try to use reverse shells for gaining remote access to the target computer.

REVERSE SHELL

In a typical remote shell scenario, the attacker is the client and the target will be the server. The user initiated the connection to the target and the target just listens to the server or the attacker. The roles of the attacker and the target gets reversed in the case of the Reverse Shell. Reverse Shell is a security methodology which is used to gain access to a remote computer. If an attacker tries to connect to a target computer, there are several ways in which the target can prevent this from happening. Some of the ways are configuring a firewall from receiving unknown connections, Applying IDS and IPS, etc. So, this time we do not connect to the target and make the target connect to our system and we can gain access to the system. Since the connection is initiated from the target, firewalls or any network filters do not filter traffic from our system and allows us to interact with the target.

Most of the target systems have set the firewall configuration to allow incoming traffic through specific ports which they use for their own purposes like HTTP (80) / HTTPS (443). Firewalls are generally configured to block any incoming connection to the server through any other ports. But they do not block the outgoing

connection in any port. So, we can use any other port to connect to the attacker. This is the major reason why most of the hackers use the Reverse Shells for accessing remote systems' terminal.

PROBLEM STATEMENT

There are lot of types of attacks that hackers are using now-a-days to hack into systems of targets. One of those attacks is Reverse Shell. Attacker need a specific program to run on both his and target computer to get the reverse shell. This project is used to generate these python scripts for attacker and victim.

SOLUTION

Solution for this problem is implemented in this project of Reverse Shell. In this project, we used python to create the Reverse Shell. We write two programs, one for the Client-side and one for the Server-side. Since the target computer is trying to connect with our system, target will be the Client and our system will be the Server. All that we need to do is make the target run the Client-side program in his system and the rest will be handled by the Server.

EXISTING METHODOLOGY

The basic idea of creating a reverse shell is to make the attacker (Server) system listening to the incoming connections through a specific port and make target (Client) to send interactive shell traffic using the same port number to the listening attacker's computer.

A. Server Side

Server-side code is the code that is to be running on the attacker's computer. The main task of the attacker's computer is to create a socket on his computer, bind it to a specific port and listen to the incoming connections to that port. When a connection is found, establish the interactive shell.

A socket is created initially on the attacker's computer. Then, it is bound with the IP of the attacker's system and a port number which we do not use commonly. The socket which is created in the server is set to listening mode.

Whenever it finds the incoming connection to that port, it is configured to accept the connection. By now connection from the client to the server is successfully established. Now it is time to get an interactive shell.

Server need to send the commands to the client which are supposed to execute in the client system and sent back to the server.

After the transaction is completed, attacker can break out of the loop of interactive shell with “quit” or “exit” command.

B. Client Side

Coming to the client code, it is the program that is to be executed in the client or target system. The major function of this client code is to loop around receiving the command sent from the attacker, executing the command, and sending the result back to the server so that attacker can have an interactive shell with this system.

The client creates a socket and connects to the server IP address of the server with the same port that is used by the server. After the server accepts the connection to the client, the connection will be successfully established. The client will get the Current Working Directory (CWD) using OS module and send it to the server as soon as the connection gets established as this makes it look more a real terminal.

C. Server-Client Synchronization

When we are working on a Server-Client model, it is very important that we make sure that server and client are synchronized. It is the job of attacker to make sure the server is running before the client runs. If the client runs before the server, client cannot find server to connect and socket will be closed because the connected host cannot respond.

The sequence of operation will be as follows:

- Server starts running.
- A socket is created by the server.
- Socket is bind with host IP and port number.
- This socket starts listening.
- Client starts running.
- Client tries to establish connection with the server.
- Server accepts the connection from client.
- Client sends the Current Working Directory to the attacker.

- Attacker prints it to make it look more like a terminal.
- Client goes on the loop to receive commands from the attacker.
- Attacker sends the commands to client in the same loop.
- When attacker want to quit, he gives “quit” or “exit” command.

PROPOSED METHODOLOGY

As we have seen above, in the conventional method of reverse shell, an attacker can connect and attack one system at a time. While in our approach, we tried to make it possible for attacker to have multiple targets at a time. While attacker is in shell of one target, rest of the targets connected to the attacker will be in sleep mode until the attacker selects the target.

Now he binds his own IP address and any port which is mostly unused. Because, it we use the common port for the socket binding, that port may have some other function to do like receiving web traffic, mails, or something. The traffic through this port will interfere with the shell traffic through the same port. So, it is always advisable to use uncommon port for manual port allocation to any services.

Since the socket is bound with the IP address of the attacker and the port, all we need to do is set this socket in listening mode. So that we can hear any incoming traffic to this IP address through the selected port. While specifying listen mode, we need to give the backlog. Backlog indicated that whenever the unaccepted connections number exceeds the backlog, the new connection trying to connect will be refused to connect. We specify the backlog as 5.

Now the server is all set to receive and accept the connections. When the client script runs, first thing it does is to create a socket and connect to the attacker’s computer through the socket created using attacker’s IP address and same port number used by the server to bind the socket.

This connection request sent by the target is accepted by the attacker automatically and this new connection established will be notified to the attacker.

Attacker can view all the active connections by typing “list” command in the prompt. This will list out all the active connections to the attacker including the index.

Out of all the active targets, attacker can select the target which he wants to get the shell using “select” option with the index. This will give the attacker the shell of the target computer. Now attacker can enter any command

he wants to execute in the victim computer based on the operating system of the target system he was working on.

If the attacker enters "exit" command when he is not inside the terminal of target, then this the program will exit the present thread and program will exit due to join functionality of thread.

TECHNOLOGIES USED

PYTHON

All the work done in this project is based on python. Both the client and Server code are scripted using python. In the recent times, many of the security scripts are being written in Bash, Python, and PHP. As python is simple and robust language, it is used mostly among all other option.

MODULES USED

- ***Socket Module***

  This module is used for communication among devices in the network. Socket acts as a communication link between the two systems in communication.

- ***Subprocess* Module**

  Subprocess is used to execute the command line commands and store the result into an object.

- ***Pyfiglet module***

  Pyfiglet is a module in python used to generate graphic designs of the text entered based on the arguments given to it.

- ***Threading module***

  Threading is used to break the program into multiple parts and make them execute parallelly so that the execution of the program is boosted.

IMPLEMENTATION

Implementation of this project requires a server which runs actively which is used by the attacker. Since the target system needs to connect to the attacker, attacker computer needs to be listening to the incoming connections all the time. So, the implementation will be as follows:

STEP-1: Running the Server Script on the Attacker's Computer

Initially any active firewalls on server side need to be turned off as it may refuse connection from the attacker. Now run the server code on the attacker's computer. This will set up the listening socket on attacker side.

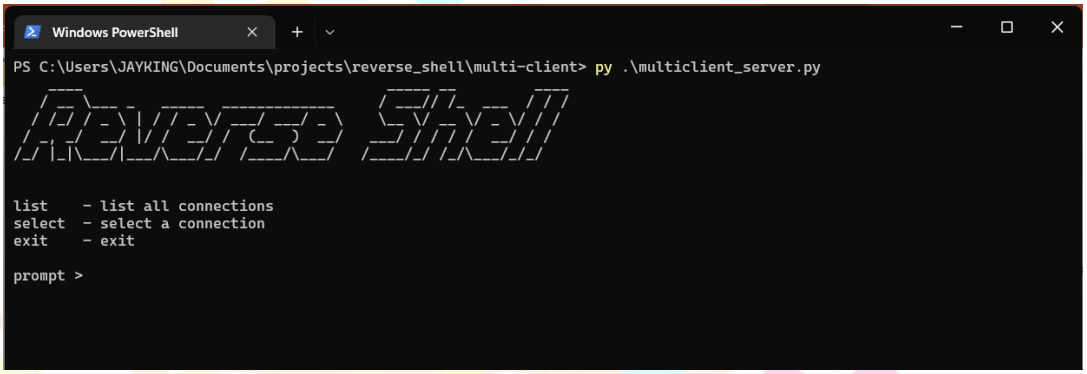

Fig.1 Initial Interface of the Attacker's Computer

STEP-2: Running the Client Script on the Target Computer

As the server code is already running on attacker computer, connection between server and client establishes as soon as the client code starts running on target computer. There should not be any intimation of this process running on target side. So, nothing will be printed on his side. The terminal will be blank as shown:

Fig. 2 Execution in Client Computer

STEP-3:Gaining Interactive Shell

Established connection will be intimated at attacker terminal. Attacker have several options to select in his program:

- **List:** This lists out all the active targets connected to attacker computer.
- **Select:** This command is used to select a target connected to gain the interactive shell.

- **Exit:** This command will help you exit from the program.

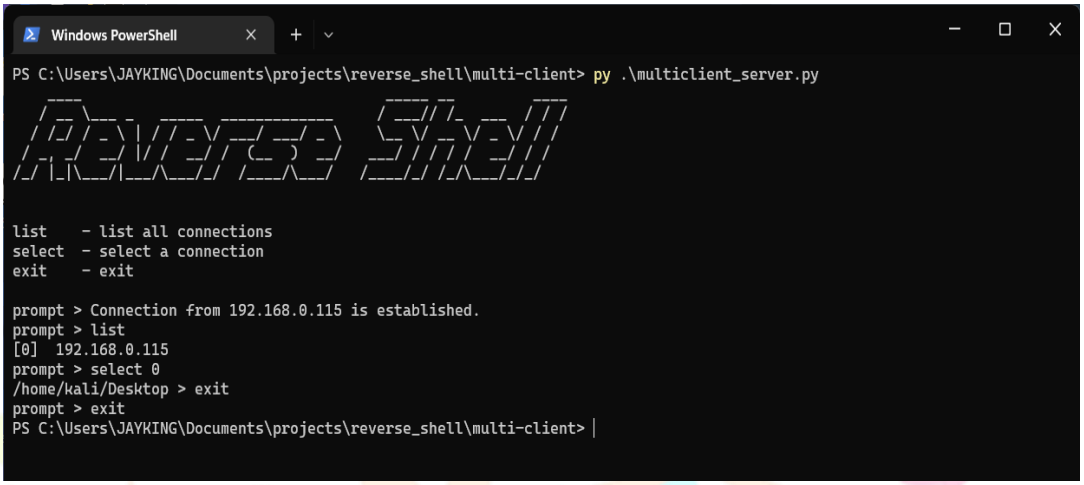

Fig. 3 Obtained Interactive Shell

OUTPUT

After the target is selected from the list of active connections, interactive shell will be started. Attacker can do anything he wants in the target computer now.

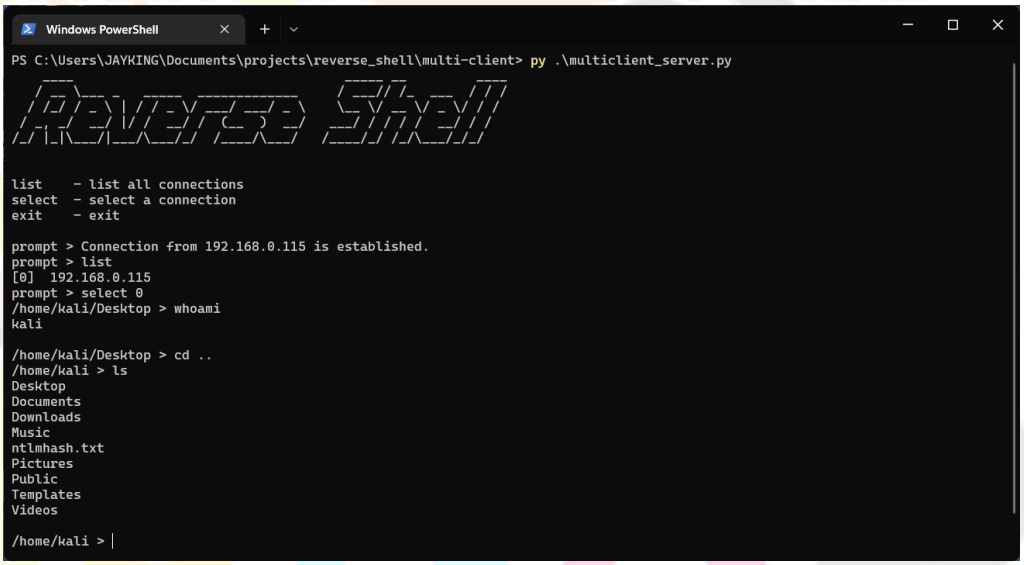

Fig. 4 Using Target's Terminal on Attacker's Computer

PREVENTIVE MEASURES

Most of the attackers use reverse shell for gaining access to a remote computer. Due to its ability of remote administration, it is most often used by attackers in most of the attacks. According to the client, it is very difficult to block the reverse shell connections due to outgoing traffic instead of incoming connections. So, there is no direct approach of gaining resistance from reverse shell attacks.

All we can do to minimize the effect of reverse shell is harden our systems with security best practices. Some of the techniques which are used to harden the security of system are as follows:

- Blocking all the outgoing traffic and new connections help us in keeping attackers away from our computer as the reverse shell cannot be produced without outgoing connections from the target system.
- Using a Proxy Server help us in a great extinct in preventing reverse shell as it appears to be some other server IP rather than our own IP address to the attacker. If attacker tries to target your IP, he will be targeting the proxy server.
- It is always advisable to use any anti-virus software as it will be running in the background always and help us in identifying the known malwares and any suspicious files in the system.
- Updating the system regularly helps us in covering security patches which were identified for the system from the recent security patch.
- Keeping any application which, you do not use is an added security threat to your computer. We never know which application is vulnerable to which attack. So, it will be best if you regularly check and remove unused applications for a long time.
- In case of victims who are unaware of security and cyber-attacks, attackers generally try to send mails including something that the victim would be tricked to open. This might contain malicious scripts which can generate a reverse shell to attacker.
- Everyone in the present world use browsers to surf internet. We go through millions of links in our daily life. We do not know which malicious file is behind which link. So, never click on unknown links.
- Using the firewall is always a best option to prefer. It helps us in preventing the incoming connections to specified ports, from specified Ips, etc. It helps in keeping attackers away from our systems.
- If we have a web server running, always ensure the filetype if the server takes file input from the users. This may lead to file inclusion vulnerability if we do not verify the filetype of server.
- If the same server takes input in text form, always sanitize the input as it may lead to command injection, SQL injection, and some other harmful vulnerabilities.
- Always maintain the systems password protected. This will help us in keeping the data encrypted even if attacker got access to the physical system.

- It is always advisable to use strong passphrases instead of passwords as it will be hard for dictionary and brute-force attacks.
- Changing the passwords time to time also helps in keeping security system strong. Even if the attacker gets the password once, it will be no longer useful for him if we change our password.
- Never share any information (either personal or professional) on unsecure and unknown lines and networks. Attackers can listen to these lines and get the information we transmitted over that line.
- Always prefer using the user account with minimum privileges and permissions. Even if we do something wrong, this will prevent us from doing the tasks which may harm the system and make it vulnerable. Even if the attacker gets the shell of the user, he will not have high level privileges as the user account has low level privileges.

CONCLUSION AND FUTURE SCOPE

CONCLUSIONS

By this project, we facilitated the use of Reverse Shell without any intrusion from the target system. Applied a new implementation of interacting with multiple clients to the Traditional Reverse Shell. We also succeeded in speeding up the process of interaction many times faster. Applying this security project for the good of companies will help them to analyze the extent of security they have. Penetration testers can use this to generate the reverse shells from many targets without creating servers many times.

SCOPE FOR FUTURE ENHANCEMENTS

This process can further be upgraded by making this work successfully outside the LAN. We can also add several new functionalities like managing the connections from the prompt without entering the system. Also, we can make this advance by making the client script run once and add it to the system processes so that it will be running even after we restart the target computer.

ACKNOWLEDGEMENT

We are also grateful to Mr. S. K. Satyanarayana, Assistant Professor, ECE Department, Sreenidhi Institute of Science and Technology for assistance with our approach towards Reverse Shell, for sharing your knowledge with us during the course of the project, for your comments on the initial versions of manuscript.

REFERENCES

[1]. Keshav Kaushik, Sakshi Aggarwal, "A novel approach to generate a reverse shell: Exploitation and Prevention" in Researchgate article, September, 2021, pp. 83-93.

[2]. M. Sullivan, "8 Types of Cyber Attacks your Business Needs to Avoid," Intuit, online.

[3]. X. Yue, W. Chen, and Y. Wang, "The Research of Firewall Technology in Computer Security," pp. 1-4, 2009.

[4]. M. Bongard and D. Illi, "Reverse Shell via Voice (SIP, Skype)," Dec. 2019.

[5]. C. Atwell, T. Blasi, and T. Hayajneh, "Reverse TCP and Social Engineering Attacks in the Era of Big Data," pp. 1- 6, 2016.

[6]. L. Chenke, Y. Feng, G. Qiyuan, Y. Jiateng, and X. Jian, "Anti-reverse-engineering tool of executable files on the windows platform," in Proceedings - 2017 IEEE International Conference on Computational Science and Engineering and IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, CSE and EUC 2017, Aug. 2017, vol. 1, pp. 797–800, doi: 10.1109/CSE-EUC.2017.158.

[7]. J. Uitto, S. Rauti, J.-M. Mäkelä, and V. Leppänen, "Preventing malicious attacks by diversifying Linux shell commands."

[8]. "Understanding Reverse Shells | Netsparker." https://www.netsparker.com/blog/web-security/understanding-reverse-shells.

[9]. Y.-G. Li, Y.-C. Chung, K. Hwang, and Y. Li, "Virtual Wall: Filtering Rootkit Attacks To Protect Linux Kernel Functions," IEEE Trans. Comput., pp. 1–1, Sep. 2020, doi: 10.1109/tc.2020.3022023.

[10]. "Command injection: how it works, what are the risks, and how to prevent it | Snyk." https://snyk.io/blog/command-injection.

[11]. "Unrestricted File Upload | OWASP." https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload.

[12]. X. Lin, L. Lei, Y. Wang, J. Jing, K. Sun, and Q. Zhou, "A measurement study on linux container security: Attacks and countermeasures," in ACM International Conference Proceeding Series, Dec. 2018, vol. 18, pp. 418–429.