Exploiting Application Layer Services

A lot of different services can work across the internet. These services can have a lot of different use cases.

Attacking services always starts out with enumeration. We need to understand what services are running and where they are running to begin attacking them. We would do enumeration using nmap or similar scanning tools. One aspect that makes our jobs easier is that services are known to run on particular ports. Each service has its own default port, which also makes it easier to identify.

Once we've identified a service that is running on an open port, we can think about potential attack vectors. Below is the most common way of attacking *known* services:

- Exploiting common misconfigurations
- Using publicly available exploits

Services are usually set up by administering the computer, and most of the time, these services have weak default configurations. Other times, we would use information such as version numbers and service type to look for public exploits(vulnerabilities other people have found). It is good practise to always start of by exploiting misconfigurations. If public exploits do exist for a running service, it is not advised to carry out destructive testing(exploits may be unreliable and could affect the availability of systems).

We'll walk through some common services and understand how to exploit common misconfigurations.

FTP

FTP is the file transfer protocol. The protocol usually runs on port 21 on top of the TCP protocol. FTP is a fairly old protocol that is used to transfer files. A user would make a file available by uploading it on the FTP server, and another user would use an FTP client to connect to the server and download the file. Some common misconfigurations for FTP include:

- Anonymous login: FTP servers would usually need users to authenticate themselves to the server to access files. Most FTP servers allow anonymous login where a user can authenticate with the username: anonymous and password: anonymous. While these are the most common anonymous credentials, they can differ depending on the variant of FTP running.

- Cleartext communication: by default, an FTP client does not encrypt its communication when interacting with the FTP server. This means that an attacker at any point on the network can intercept user credentials and downloaded files.
- Poor file system permissions: FTP servers can be configured to allow an FTP user to browse the rest of the file system. This would help an attacker to enumerate the file system to find even more sensitive information e.g. users personal files, credentials and more.

To connect to an FTP server, you can use the command line command ftp:
ftp ip-address

Is this realistic: **very.** Even though FTP servers are old, big corporations that have legacy environments still tend to use FTP to store and transfer files.

NFS

NFS is a network file share that runs on both TCP and UDP on port 111 and 2049. Like FTP, NFS is used to transfer files. They are still quite different. While FTP uses a client-server model to communicate, NFS acts as a distributed system. This means that a user can access a share(think of this as a directory) on their own file system. While FTP uses username and password to manage authentication and authorization to files, NFS uses the linux permission system to manage these things. Common misconfigurations include:

- Publicly accessible shares: some administrators may expose NFS shares to anyone. An attacker could mount the share onto their file system and access files on the share.
    - When the permissions a file are different, an attacker would have to change their user ID/group ID to match the permissions of the file.

The first step would be enumerating a system to check if NFS is running. Once NFS is running, we would check to see if any shares are available. This is done using this command
showmount -e ip-address
This command shows all the shares exported by NFS.

If this command outputs any shares, you can try mount the shares on to your file system
mount ip:/file/path /local/file/path
Note that this command would require sudo permissions

Once it is successfully mounted, you can browse to the location on your file system and try access the files. After completing this, you need to unmount the file system using the command:
umount /local/file/path
Note that this command would require sudo permissions

Is this realistic: it is! Like FTP, NFS is still used by big organisations in their legacy environments. Note that NFS has had version improvements that improve some default security weaknesses.

MySQL

MySQL is a service that runs an SQL server. A SQL server is a particular type of a database server that uses structured query language(SQL) to add and manipulate data(in a database). MySQL uses TCP and

runs on port 3306 by default. SQL does not have any common misconfigurations that allow direct access, but is usually used as part of an attack chain. An attacker may find other information and use this to compromise a MySQL server. If you find a MySQL service running, you should attempt to connect to the service. If you manage to connect to the service, you should enumerate the database in the following ways:

- Examine databases
- Examine tables
- Look for sensitive information like passwords and personally identifiable information.

Is this realistic: **yes.** It is not uncommon to find MySQL services exposed to the internet. While some services may use default credentials, you would usually need some form of username and password to access the database.

Check out these cheat sheets to get started with basic commands:

- https://gist.github.com/hofmannsven/9164408
- https://gist.github.com/bradtraversy/c831baaad44343cc945e76c2e30927b3