UDC 004.9:005.8

Vulnerability detection in the network traffic flow of the RADIUS protocol based on the object-oriented model

Leonid Galchynsky[1], Amina Murtazina[1]

[1]*National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Educational and Research Institute of Physics and Technology*

Abstract
The RADIUS protocol was analyzed from the point of view of its functionality and security. The internal structure and the division into functions were shown. There were described the structure of the RADIUS network, the functions of the network access server or NAS, and the RADIUS server. The advantages of the centralized secure data processing technology based on the RADIUS protocol were shown. Mechanisms of secure processing of requests at the authentication and authorization stage were described. Sources and types of protocol vulnerabilities were studied and possible attack scenarios were identified. The relevance of creating models for evaluating the vulnerabilities of the RADIUS protocol was substantiated, and the methodology for building the model was chosen. An object-oriented model of the RADIUS protocol has been developed. A software application was developed and various attacks on the RADIUS protocol were tested. A number of potential vulnerabilities have been identified.

Keywords: RADIUS protocol, vulnerabilities, types of attacks, object-oriented model

Introduction

The rapid progress of communication technologies has significantly affected the way of life all over the world over the last decades. The number of users of communication networks is growing rapidly. Users are offered more and more services. At the same time, another trend is also observed - the level of cyber threats is increasing. Not only the service as such, but also security and privacy are becoming important for online users. Network security is an area where it deals with understanding, controlling and managing the risks of the infrastructure and its assets. Network security has a number of properties that must be satisfied. In modern networks, in particular in the networks of the IoT, the protocols by which data is transmitted in the networks are one of the decisive factors for both the provision of services and a reliable level of cyber security. For a long time, the most common method of connecting users to work networks was SSID + shared password. This method is time-consuming and complicates access control, and it creates significant network vulnerabilities. The complexity of networks and the simultaneous growth of cyber threats required the development of more relevant protocols.

One of such protocols are protocols under the common name AAA[1]. Representatives of this class of protocols that ensure availability, integrity, authentication, confidentiality and non-failure are: RADIUS (Remote Authentication Dial-in User Service) and TACACS + (Terminal Access Controller Access-Control System Plus) and Diameter. DIAMETER belongs to P2P protocols, where each node (reeg) can function both as a client and as a server.

Authentication is the process of verifying and confirming the user's identity. Before granting access to any resource, such as a device or service on a network, you basically need to verify the end user. Most authentication systems require a password to verify and confirm a user's identity.

Authorization is the process of granting rights to the user. The network infrastructure offers a significant number of services and is available to a large number of users for use. Different user levels require different rights. Authorization is also included to grant various privileges to service consumers along with authentication.

Accounting is a more efficient part of the entire network technology. Available services, devices and processes must track information about the end users and their activities. Therefore, certain types of logs are kept to provide accountability and auditability for both consumers and service providers. Figure 1 shows the working diagram of the AAA protocol using the example of the the RADIUS protocol.

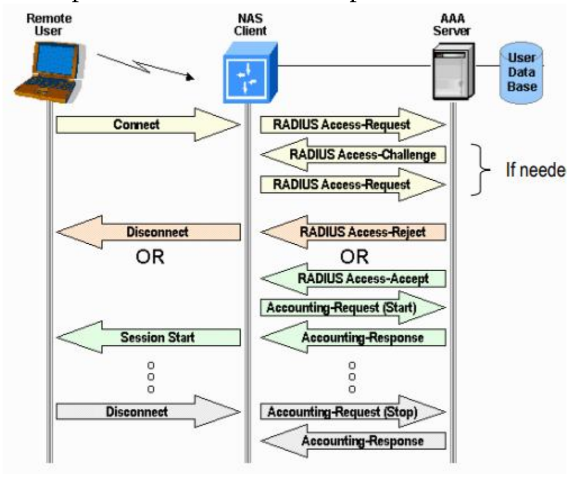


Figure 1: Working Mechanism of RADIUS Protocol

RADIUS is a network protocol widely used in network services for security purposes. Since many systems cannot handle large numbers of users with unique credentials, because this requires a lot of memory and devices, there is a need to use more complex protocols than PPP protocols. AAA protocols significantly improve the level of cyber security of networks compared to the class of PPP protocols. As a result, RADIUS is one of the most widespread AAA protocols used in real-world scenarios.

In addition to improving cybersecurity, this protocol also provides centralized user administration, and its support is nearly ubiquitous and uniformly supported. The purpose of using RADIUS is to create a hub for user authentication where users or clients from different locations can request network access and services. The simplicity, efficiency, and ease of use of the RADIUS system have made it popular among network service providers to the extent that RADIUS is now considered an industry standard[2]. At the same time, the experience of operating networks using RADIUS systems shows that they also have certain vulnerabilities and are also subject to numerous attacks[3]-[4]. Moreover, cybercriminals use a significant number of different strategies with the aim of both attacking networks from the outside and illegally penetrating the network. The complexity of the network itself, multiplied by a significant number of options for attackers to penetrate the network, raises the question of the development of a tool to assess potential vulnerabilities of the RADIUS protocol.

The aim of the study is to assess the potential vulnerabilities of the RADIUS network based on the use of an object-oriented model of the RADIUS protocol.

1. RADIUS protocol functionality analysis

The RADIUS system, as shown in Figure 1, is divided into four components: user/devices, network access server, RADIUS server, and databases. In turn, the database contains user passwords, data on their rights, and configuration data. This protocol has three main functions:

1) To authenticate users before admitting them to the network;

2) To authorize these users to access network services;

3) To record the use of these services.

RADIUS is an open-source protocol that provides a client-server data exchange scheme where the network access server or NAS is the RADIUS client and the centralized software server is the RADIUS server [5]. This protocol regulates access not only to the Internet, but also to e-mail services, internal and wireless networks. RADIUS works using the UPD transport protocol [6] and encrypts only the user's password during its transmission between the client and the server. At the same time, complex data storage in the database, which can be used by all clients, is supported. According to the protocol, the remote user connects to the network using the NAS network access server by entering a login and password [7]. After that the NAS sends a request to the RADIUS server - the AAA server - in order to obtain all the necessary information about the remote user. Such a request is called ACCESS-REQUEST. The user database and the accounting database are stored on the AAA server. Therefore, when the RADIUS server receives a request with a login and a password, it checks the presence of these records in the first database. If the records match, the client receives permission for the user to access the network, his session and a list of rights. This happens by sending an ACCESS-REQUEST packet. To receive the RADIUS accounting status, the client sends an ACCOUNTING-REQUEST. As a result, a record about this user appears in the accounting database, where all his activity during this session will be entered. And after that, the NAS can allow the user access to the network and send the received data to him (Figure 2). In the case when the network is accessed by a local user - the communication between the client and the server is almost not modified, a WAP wireless access point will be used instead of the NAS.

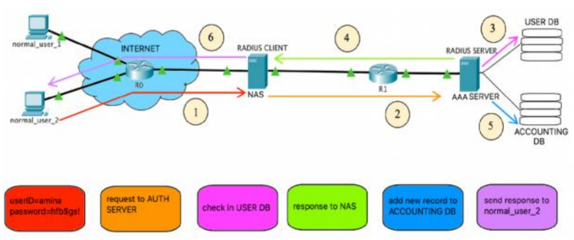


Figure 2: RADIUS message exchange scheme

The RADIUS message consists of a header and attributes. Each RADIUS attribute defines a piece of information about a connection attempt and is described by variable-length attribute-length-value 3-tuples. RADIUS attributes are described in the RFC 2865. A RADIUS server is a shared authentication server that has a list of valid clients. A shared secret exists between the RADIUS server and these clients. This secret cannot be empty, but the protocol standard does not specify how secure it should be. It is only recommended that it be a minimum of 16 octets. This secret is used to authenticate the RADIUS server on the NAS and to hide the user's password. For these purposes, the secret is part of the value being hashed, and it is this hash value that is sent [8]. The RADIUS protocol is assigned UDP port 1812. The choice to use UDP over TCP is mostly because UDP is a lighter protocol than the more reliable TCP. RADIUS is a stateless protocol that does not transmit much data because the maximum UDP packet size is 4096 octets. Since RADIUS is used for user authentication, a delay of a few seconds is acceptable.

1.1 Authentication and authorization

A network access server (NAS) works as a client for the RADIUS server. The NAS asks for the necessary authentication information, including username and passwordand it can then use the RADIUS server to authenticate the user. In doing so, the NAS sends a request to the RADIUS server containing attributes that have user information to the RADIUS server. When sending a request containing a user's password, the password is not sent in clear text, instead it is sent encrypted. After that, the NAS waits for a response from the RADIUS server. The server can accept or reject the request, or prompt the user to respond. If the request is accepted, the server may also provide the NAS with configuration data and the type of service provided to the user. If the RADIUS server does not respond within the specified time, the NAS can retransmit the request or can use possible alternative RADIUS servers.

Six types of packets are exchanged between the client and the user. The code is 1 byte long and defines the packet type. For example, a code value of 1 means access request type, 2 means access-accept, 3 means access-deny, and so on. The identifier is also 1 byte long and is used to match responses to requests. The length is 2 bytes and defines the length of the packet. The authenticator is 15 bytes long and is used by the client to verify the validity of the server's response and to hide the password

Attributes may contain authentication, authorization, and configuration in TLVs (type, length, and value), and may include username, password, CHAP password, NAS IP address, NAS port, service type, EAP message, and message authenticator.

RADIUS protocol operation has six types of packets. An access request is sent by the user to the server and contains information to determine whether the user is authorized for a particular NAS and whether the services in the request are available to the user.

Access-Accept is sent by the server to the user with the information necessary to start the delivery of the request. In this message, the authenticated field is computed using an MD5 hash.Access-Reject is sent by the server to the client if the attribute value is invalid. An Access-Challenge is sent by the server to the user via the NAS as a challenge that requires a response. This is a request to the client for additional data, such as the MAC address. The accounting request is sent by the NAS or the user to the server that also performs the accounting. The Accounting-Response is sent by the server with the code field set to 5, and no attributes are required.

Figure 3 shows RADIUS authentication and authorization procedure [8]. They unfold according to the following simplified scenario:

1. The network user acts as the Applicant and sends authentication attributes to the NAS using the EAPoL network protocol, which stands for Extensible Authentication Protocol over LAN. EAPoL (Extensible Authentication Protocol over LAN) —which is used in 802.1x (Port Based Network Access Control). In other words, it is an encapsulation protocol used between the Supplicant and the Authenticator. At the beginning, the requester does not know the MAC address of the authenticator. So, it sends this message to the multicast group to see if there is an authenticator on the local network.

2. The NAS adds them to the access request sent to the RADIUS server.

3. Server response types: OK, NO, Challenge (for some AUTH) if Y, user profile, authorization and configuration data added.

4. The NAS notifies the user.

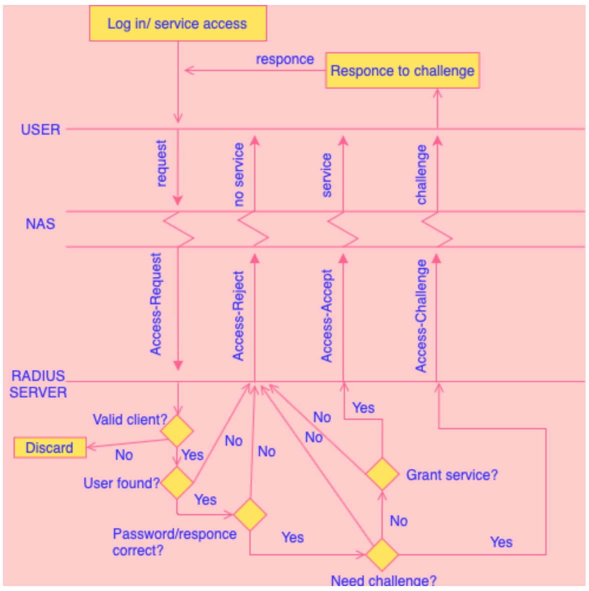


Figure 3: RADIUS authentication procedure.

The authentication process is closely related to the authorization process and is a chain of checks of the validity of the client and his rights. It should be noted the interesting three-digit logic of these checks (Yes, No, Challenge). First, the validity of the client is checked, after which a search is made in the database. If such a client is present in the database, his rights are checked. If the rights are confirmed, only then is the Access-Accept sent. But the Challenge request must be additionally checked.

1.2 Accounting

UDP port 1813 is used for RADIUS accounting. There are also two RADIUS message codes and 12 attributes for RADIUS accounting. In addition, the Request Authenticator is calculated differently when using RADIUS accounting [8]. Accounting begins with the NAS sending a RADIUS packet with the Accounting-Request code having an Acct-Status-Type attribute of Start to the RADIUS server. In the request for the start of accounting, attributes containing information about the user and the service used. All attributes that can be used in an Access-Request can also be used in an Accounting-Request with five exceptions. These are User-Password, CHAP-Password, Reply-Message, State and CHAP-Challenge attributes [8]. When the NAS wants to stop accounting, it sends a RADIUS packet with an Accounting-Request code having an Acct-Status-Type attribute of Stop to the RADIUS server. This packet may have attributes containing information about the service that was used and usage statistics [8]. After receiving the request packet, the RADIUS server records the Accounting-Request and, after successfully recording the packet, acknowledges it by sending an Accounting-Response packet to the NAS. If the request could not be recorded, no confirmation is sent. If the NAS does not receive an acknowledgment of its request, it will retransmit the request or forward the request to another RADIUS server. There are no attributes in the Accounting-Response package, with the possible exception of Proxy-State and Vendor-Specific.

1.3 The vulnerabilities of the RADIUS protocol

Experience using the RADIUS protocol has shown not only its notable advantages, but also certain shortcomings in terms of cybersecurity, which are either inherent in the protocol itself or caused by poor client implementation. The UDP protocol, as a stateless system, makes it relatively easy to forge and mask packets from an unknown source as messages from a known reliable source. A deep analysis of protocol vulnerabilities was carried out on the basis of the RADIUS frame [10]. The RADIUS frame is presented in Figure 4.

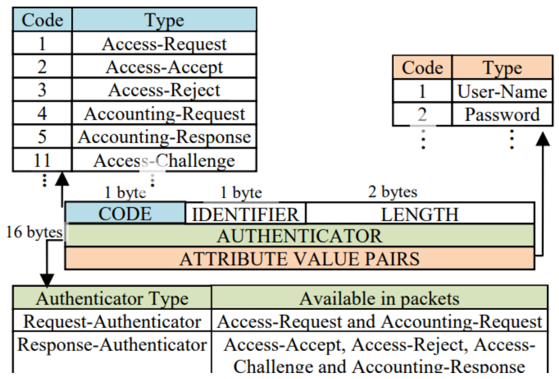


Figure 4: RADIUS frame

The following three fields in the RADIUS packet are relevant from the point of view of the attack [7]:

1. Code field: sets the type of RADIUS package. Of the value and attribute pairs, the following are relevant to the attack:

a) Access-Request, b) Access-Accept c) Access-Reject.

2. Authenticator: used to authenticate the response from the RADIUS server. It encrypts passwords and has the length of 128 bits.

3. Attributes: The only relevant attributes that are delimited are the User-Name and User-Password.

A detailed review of the vulnerabilities that generate the specified fields is too large for this article. The vulnerabilities listed below are not a complete list of protocol problems and all methods of bypassing user authentication, but they are based on facts known to network users and the expert community.

The purpose of this study is related to the analysis of vulnerabilities of the RADIUS protocol. Since the protocol involves encryption, its potential vulnerabilities are rooted in insufficiently thorough encryption, as well as in certain features of the RADIUS protocol [11]. Firstly, it is advisable to consider the methods of encryption when transmitting messages between network objects [12]. To calculate the value of the response authenticator, the field transmitted in the packet from the server, the following expression is used:

$$RespAuth = MD5(Secret + ReqAuth + RawRespData) \tag{1}$$

where:

- RespAuth – response authenticator,
- MD5 - is a hash algorithm based on the Merkle–Damgaard hash function, a method of constructing collision-resistant cryptographic hash functions from collision-resistant one-way compression functions,
- Secret - shared secret,
- ReqAuth – request authenticator,
- RawRespData – transmitted data.

The value of the user's encrypted password is implemented by the expression:

$$EncrPasswd = MD5(Secret + ReqAuth) \oplus Passwd \tag{2}$$

where:

- EncrPasswd – encrypted password,
- MD5 - is a hash algorithm based on the Merkle-Damgaard hash function,
- Secret - shared secret,
- ReqAuth – request authenticator,
- Passwd – plaintext password.

The client and the server have a secret. This shared secret, followed by the request authenticator, is passed through an MD5 hash to produce a 16-octet value that is XORed with the password entered by the user. If the user's password is longer than 16 octets, additional MD5 calculations are performed using the previous ciphertext instead of the request authenticator. The calculation is performed according to the expression:

$$EncrPasswd \oplus Passwd = MD5(Secret + ReqAuth) \tag{3}$$

Attacks on the RADIUS protocol components include:

- Response authenticator attack. Since the response authenticator is an MD5 hash with a key, if an attacker detects a pair of ACCESS-REQUEST and ACCESS-ACCESS or ACCESS-REJECT packets, he can easily implement an attack on the shared secret offline. Moreover, having previously calculated the values of the response authenticator according to Formula 1, the attacker will be able to compare these values with the hash of each received secret.
- A shared secret attack using password attributes. Provided an attacker has access to the network traffic and attempts authentication, he will be able to obtain information about the shared secret. This is done by choosing a stream cipher to protect the User-Password attribute. Accordingly, the attacker authenticates with a known password. After that, it intercepts the ACCESS-REQUEST packet with the request authenticator. Using Formulas 1 and 2, an

attacker can launch an offline attack on the shared secret.

- Password based attack. It is very similar to the previous one, knowing the shared secret, the attacker is able to determine the user's password by modifying and resending the ACCESS-REQUEST packet. Moreover, if the server does not check the authenticity of the user in any way, it is possible to perform an online search for the password of the corresponding user. Such an attack can be prevented by introducing a correct authentication scheme for the data transmitted in the ACCESS-REQUEST packet.
- Authenticator based attack. In order for communication using the RADIUS protocol to be secure, it is advisable to generate a unique and unpredictable request authenticator. However, sometimes the implementation of this requirement leaves hope for the best — the values are repeated and generated quite obviously, which allows the attacker to forge the authenticator with his own hands.
- Server reply attacks. Accordingly, an attacker is able to create his own database, where authenticators, request IDs, and related server responses will be stored by listening to the traffic between the client and the server. As a result, it can masquerade as a server and use the answers available in the database if it sees a packet with a familiar authenticator.
- Protocol implementation attacks. Incorrect processing of received packets can occur both on the client side and on the server side, which leads to DoS attacks or even RCE attacks.
- The problem of using a shared secret should be noted separately. The fact is that the RADIUS standard allows many clients to use the same shared secret. This is dangerous, because any client with malicious intent can compromise several machines at once. Therefore, the process of creating and assigning a shared key should also be given sufficient attention.

2. RADIUS OBJECT-ORIENTED PROTOCOL MODEL

Both the advantages and disadvantages of the RADIUS protocol from the point of view of cybersecurity have been evaluated in sufficient detail by the efforts of the community of experts. However, there are few works that attempt to assess the vulnerabilities of this protocol by using formalized models. On the basis of which it would be possible to develop software tools for comprehensive assessment of RADIUS vulnerabilities. A common approach to developing such a model is the concept of a finite automaton. Thus, in particular, [13] shows a variant of such a model based on a classical finite state machine (FSM), and in [14] based on an extended finite state machine (EFSM). The results of using these models showed that from the point of view of the finite state machine formalism, the RADIUS protocol has no vulnerabilities. However, when all conditions are considered, some of them appear more vulnerable than others and this makes them a source of vulnerabilities. However, this can only be assessed by an expert. It is possible that this is a promising approach, which, however, requires a conceptual deepening of the finite automaton model itself.

In this work, it is proposed to approach the development of the RADIUS protocol model based on an object-oriented approach. In favor of this choice, certain features of this approach for modeling such a complex system as a set of attacks on the RADIUS protocol testify. In contrast to the automatic approach, in which there is only a legitimate client hat will undergo authentication and authorization. In the OOP concept, the presence of an intruder in the network can be postulated immediately, and his actions will not be limited to unsuccessful attempts to penetrate the network without knowing the password and shared secret. Figure 5 shows the presence of an attacker in the RADIUS network, whose goal is to illegally penetrate the corporate network.

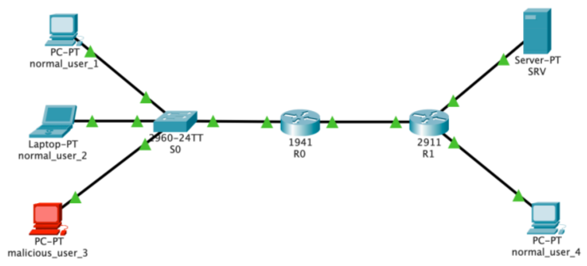


Figure 5: Corporate network with an attacker diagram

The reason for this choice is that each object has its own behavior - a set of methods - and identity - a set of fields. The use of an object-oriented model will allow predicting the behavior of the system during attacks under various scenarios. In addition, such a model can be easily expanded due to the addition of types of cyber attacks and changes to the parameters of the RADIUS network.

According to the OOP methodology [15], the complete model of the subject area should be described by the following diagrams:

- class diagram;
- object diagram;
- diagram of states and transitions;
- sequence diagram;
- module diagram.

The practice of using object-oriented models shows that, as a first approximation, it is enough to implement two diagrams: a class diagram and an object diagram/sequence diagram. A class diagram describes fundamental entities and the relationships between them, while object diagrams and sequence diagrams describe the behavior of instances of object classes. From the point of view of OOP methodology, these two diagrams are equivalent, but different in form. Class and sequence diagrams are implemented in this study.

2.1 Class diagram

The OOP methodology allows the freedom to choose a set of classes of the subject area and the connections between them. The following classes were selected for the OO model of the RADIUS protocol: server (RADIUS SERVER), client (RADIUSCLIENT), network interface (NETWORK INTERFACE), packet (RASKET, ACCES REQUEST, ACCESS ACCEPT, ACCOUNTING REQUEST, ACCOUNTING RESPONSE, ACCESS REJECT, ACCESS CHALLENGE, ACCESS CHALLENGE RESPONSE), abstract class user (USER) and its child classes: normal user (NORMAL USER) and attacker (ATTASKER). A complete diagram of classes and relationships is presented in Figure 6.

- PACKET is an abstract class of RADIUS communication packets. Its interitors are all possible packets participating in the data exchange between the RADIUS client and the RADIUS server. The PACKET class contains public header and payload attributes.

- ACCESS_REQUEST is a child class that represents network access request packets.

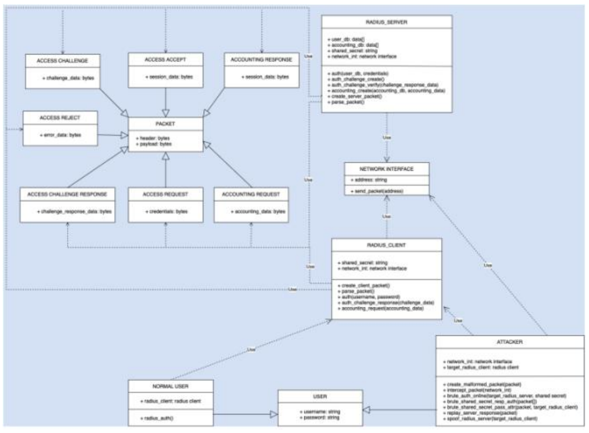

ACCESS CHALLENGE
+ challenge_data: bytes
ACCESS ACCEPT
+ session_data: bytes
ACCOUNTING RESPONSE
+ session_data: bytes
ACCESS REJECT
+ error_data: bytes
PACKET
+ header: bytes
+ payload: bytes
ACCESS CHALLENGE RESPONSE
+ challenge_response_data: bytes
ACCESS REQUEST
+ credentials: bytes
ACCOUNTING REQUEST
+ accounting_data: bytes
RADIUS_SERVER
+ user_db: data[]
+ accounting_db: data[]
+ shared_secret: string
+ network_int: network interface
+ auth(user_db, credentials)
+ auth_challenge_create()
+ auth_challenge_verify(challenge_response_data)
+ accounting_create(accounting_db, accounting_data)
+ create_server_packet()
+ parse_packet()
NETWORK INTERFACE
+ address: string
+ send_packet(address)
RADIUS_CLIENT
+ shared_secret: string
+ network_int: network interface
+ create_client_packet()
+ parse_packet()
+ auth(username, password)
+ auth_challenge_response(challenge_data)
+ accounting_request(accounting_data)
ATTACKER
+ network_int: network interface
+ target_radius_client: radius client
+ create_malformed_packet(packet)
+ intercept_packet(network_int)
+ brute_auth_online(target_radius_server, shared secret)
+ brute_shared_secret_resp_auth(packet[])
+ brute_shared_secret_pass_attr(packet, target_radius_client)
+ replay_server_response(packet)
+ spoof_radius_server(target_radius_client)
NORMAL USER
+ radius_client: radius client
+ radius_auth()
USER
+ username: string
+ password: string
Use

Figure 6: RADIUS class diagram

- ACCESS_ACCEPT is an child class that represents packets that the server sends to the client in case of successful authentication. Contains the session_data attribute, which, accordingly, stores data about the session of the given user.

- ACCESS_REJECT is an child class that represents the packets that the server sends to the client in case of failed authentication. Contains the error_data attribute, which stores information about the authentication error.
- ACCESS_CHALLENGE is an child class that is the embodiment of packets that transmit a challenge message. This class contains the login and password that the user must provide to the server for authentication.

- ACCESS_CHALLENGE_RESPONSE is an child class that represents the type of packets that are generated on the client side and contain an encoded challenge_message that is passed in the challenge_response_data attribute.

- ACCOUNTING_REQUEST – an child class that represents accounting request packages created by the client and containing the accounting_data field for sending the necessary accounting data to the server.

- ACCOUNTING_RESPONSE is an child class that is a generic representation of response packets from the server that contain a session_data attribute with updated session data, respectively.

- RADIUS_SERVER – RADIUS server class. It contains two databases: user_db and accounting_db. To share a secret, server class store the shared_secret assigned attribute. This class supports such methods as:
  - auth() - user authentication method using the PAP protocol;
  - auth_challenge_create(), auth_challenge_verify() - generation and verification challenge_message;
  - accounting_create() - creating a record in the accounting db;
  - create_server_packet(), parse_packet() - creation and processing of a packet.

- NETWORK_INTERFACE is a class that represents the method of communication between the server and the client. It supports only the send_packet(address) method of sending a packet to a specified address and, accordingly, has the attribute address.

- RADIUS_CLIENT – RADIUS client class. It also has a shared_secret attribute to establish a secure connection. This class has the following set of methods:
  - create_client_packet(), parse_packet() - creation and processing of a packet;
  - auth(),accounting_request() - creating an authentication and accounting request;
  - auth_challenge_response() - response to the sent challenge message, according to CHAP.

- USER is an abstract class representing a RADIUS user. It has a login and password attributes and two child classes.

- NORMAL_USER is a child class representing a normal user. It has the radius_auth() method, that is, it can send a request for an authentication attempt using the RADIUS protocol to the RADIUS client.

- ATTACKER is an child class that is the embodiment of an attacker, and therefore ideally should not be able to pass RADIUS authentication and authorization. The list of its methods is a large set of methods of compromise and penetration into the network. The set of methods of this class is quite large and can be replenished.

The relationships shown in the class diagram define two types of relationships between RADIUS classes - usage and inheritance.

2.2 Sequence diagram

Sequence diagrams, like object diagrams, should describe the dynamics of the area. In our case, this is a description of cyberattacks on the RADIUS protocol. Since there are quite a lot of types of cyberattacks on this protocol, there should be a lot of sequence diagrams accordingly. Therefore, let's limit ourselves to an example of one of the sequence diagrams. which describes the behavior of the network objects during an attack. Figure 7 shows the sequence diagram of the server response forgery attack sequence. According to it, the attacker configures his own interface to intercept traffic between the client and the server. When the interface is configured, the attacker analyzes all intercepted packets for coincidence with the packets available in the database. If an identical packet is found, the attacker immediately uses the spoof_radius_server() method and sends a message to the RADIUS CLIENT that is currently the new RADIUS server. After that, using the reply_server_response() method, the attacker extracts the required packet response from the database and sends it directly to the client. The client will now analyze the contents of the packet and send session data or authentication error information to the user.

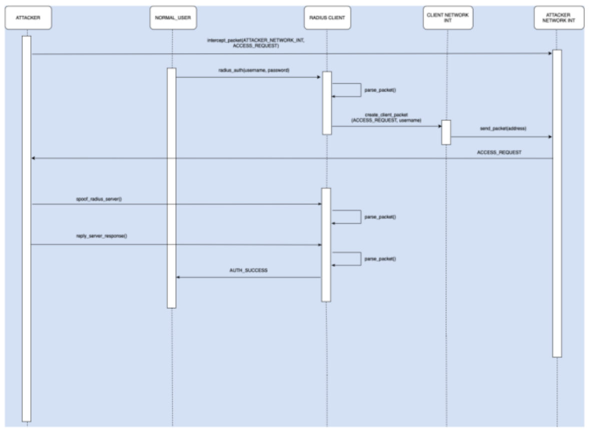


Figure 7: Sequence diagram of spoof RADIUS server attack

As we can see, sequence diagrams reflect the interaction of objects ordered by time. In Figure 7, in the form of vertical lines, simultaneously existing various objects and processes are shown, and with the help of arrows, messages are depicted, ordered by the time of sending. Messages are RADIUS communication packets, and objects are instances of classes described in the class diagram. With a suitable number of diagrams, it is possible to simulate different scenarios of attacks on the RADIUS protocol.

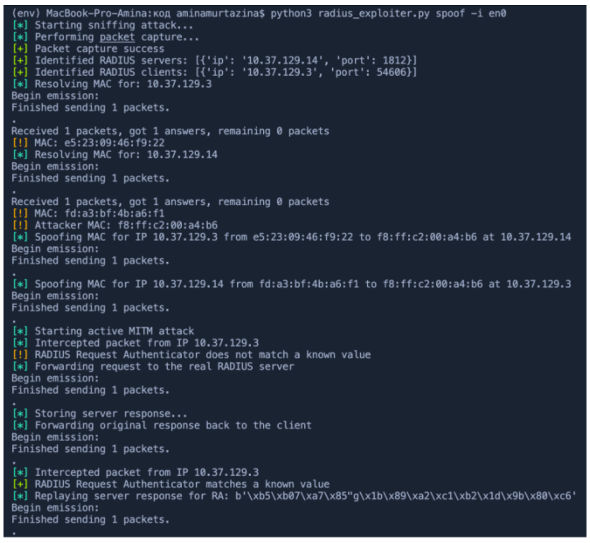
3. Testing the proposed RADIUS protocol object model

Based on the built object-oriented model, an application was created that simulates attacks on the RADIUS protocol. The application is written in the Python programming language. The application can:

- perform an audit of the RADIUS protocol;
- check RADIUS server configurations;
- contain all basic attacks on the RADIUS protocol;
- be extended - new attack scenarios can be added, as well as adapted to the DIAMETER and TACACS+ protocols.

To simulate attacks on the RADIUS protocol based on the described application, both RADIUS-server and RADIUS-client were implemented using Windows Server 2012 OS. On the server side, in Windows Server Manager, the NPS toolkit were required to be indstalled as a proxy server for remote authentication of users to balance the load and correctly redirect requests to the RADIUS server. On the client side, in the Windows Server Manager section, RRAS were installed and configured to support the connection of remote users with VPN. After adding a RADIUS client, by specifying the client's hostname, IP address, and shared secret, and configuring RRAS to use RADIUS authentication, attack simulation testing could be performed.

Next, simulations of attacks on the RADIUS protocol were tested under different scenarios. In particular, the attack of unauthorized data acquisition (spoofing) by masquerading as RADIUS_SERVER. As shown in Figure 8, this mode initially activates the interception of packets passing through the en0 interface.



```
(env) MacBook-Pro-Amina:код aminamurtazina$ python3 radius_exploiter.py spoof -i en0
[*] Starting sniffing attack...
[*] Performing packet capture...
[+] Packet capture success
[+] Identified RADIUS servers: [{'ip': '10.37.129.14', 'port': 1812}]
[+] Identified RADIUS clients: [{'ip': '10.37.129.3', 'port': 54606}]
[*] Resolving MAC for: 10.37.129.3
Begin emission:
Finished sending 1 packets.
.
Received 1 packets, got 1 answers, remaining 0 packets
[!] MAC: e5:23:09:46:f9:22
[*] Resolving MAC for: 10.37.129.14
Begin emission:
Finished sending 1 packets.
.
Received 1 packets, got 1 answers, remaining 0 packets
[!] MAC: fd:a3:bf:4b:a6:f1
[!] Attacker MAC: f8:ff:c2:00:a4:b6
[*] Spoofing MAC for IP 10.37.129.3 from e5:23:09:46:f9:22 to f8:ff:c2:00:a4:b6 at 10.37.129.14
Begin emission:
Finished sending 1 packets.
.
[*] Spoofing MAC for IP 10.37.129.14 from fd:a3:bf:4b:a6:f1 to f8:ff:c2:00:a4:b6 at 10.37.129.3
Begin emission:
Finished sending 1 packets.
.
[*] Starting active MITM attack
[*] Intercepted packet from IP 10.37.129.3
[!] RADIUS Request Authenticator does not match a known value
[*] Forwarding request to the real RADIUS server
Begin emission:
Finished sending 1 packets.
.
[*] Storing server response...
[*] Forwarding original response back to the client
Begin emission:
Finished sending 1 packets.
.
[*] Intercepted packet from IP 10.37.129.3
[+] RADIUS Request Authenticator matches a known value
[*] Replaying server response for RA: b'\xb5\xb07\xa7\x85"g\x1b\x89\xa2\xc1\xb2\x1d\x9b\x80\xc6'
Begin emission:
Finished sending 1 packets.
.
```

Figure 8: Spoofing mode of application

After that, the identification of the RADIUS server and client takes place - obtaining their IP addresses, with the help of which the MAC addresses, which will be needed for spoofing, can be easily identified. Next, the valid MAC address of the server is replaced with the attacker's MAC address and the updated data is sent to the client. Similar actions take place on the server side. After a successful replacement of addresses, a MiTM attack is implemented - packet interception, Request Authenticator (RA) values check, and response forgery in the case of a known RA.

Implementations of sniffing, spoofing, denial of service attacks, shared secret attacks, and user password mining were given. It was possible to establish that each of these attacks can be implemented in principle, but with a different degree of complexity and in terms of security due to the thoroughness of encryption.
For example, a denial-of-service attack is quite easy to implement, but it was not even possible to compromise the shared secret through the use of password attributes.

Conclusions

This paper presents the results of research into the vulnerabilities of the RADIUS protocol, one of the most popular means of functioning of modern corporate networks with a high level of security. However, there is a lot of evidence about the shortcomings of this protocol to ensure cyber security. The conducted study of the protocol's functioning mechanism showed the existence of potential opportunities for the appearance of vulnerabilities, due to which incidents of illegal intrusion into the RADIUS network may occur in cases of insufficiently maintained rules for using this protocol. An object-oriented model of the RADIUS protocol was built, which included a potential attacker with a certain set of penetration strategies. Based on the built model, a software application was developed on which a number of cyber attacks were simulated. The simulation showed that illegal intrusions are possible for some types of cyber attacks, which indicates the need to modify the RADIUS protocol.

References

[1] Arun Pratap Singh Sikarwar, Preeti Saxena. "An Analytical and Experimental Study of AAA Model with Special Reference to RADIUS and TACACS+", International Journal of Computer Applications (0975 – 8887), Volume 169 – No.9, July 2017.

[2] Jindrich Jelinek i Pavel Satrapa, Jiri Fiser. "Experimental Issues of the Model of the Enhanced RADIUS Protocol", IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM), 2015.

[3] Leonid Galchynsky, Mykola Graivoronskyi, and Oleh Dmytrenko. Evaluation of Machine Learning Methods to Detect DoS/DdoS Attacks on IoT. Selected Papers of the XXI International Scientific and Practical Conference "Information Technologies and Security" (ITS 2021) Kyiv, Ukraine, December 9, 2021, (pp.225-236).

[4] Nour Moustafa, Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic, PhD thesis, University of New South Wales, Canberra, Australia.

[5] RFC 2869, "RADIUS Extensions", by C. Rigney, W. Willats, P. Calhoun. June 2000.

[6] RFC 2865, "Remote Authentication Dial In User Service (RADIUS)", by C. Rigney, S. Willens, A. Rubens, W. Simpson. June 2000.

[7] An Analysis of the RADIUS Authentication Protocol by Joshua Hill https://www.untruth.org/~josh/security/radius/radius-auth.html

[8] Rehman, Md. Hashmathur et al. "Design and Implementation of RADIUS – An Network Security Protocol." Global journal of computer science and technology (2010).

[9] Vulnerabilities, http://books.gigatux.nl/mirror/wireless/0321202171/ch13lev1sec4.html https://www.ccexpert.us/radius-server-2/vulnerabilities-attacks-and-common-exploits.html

[10] Snehasish Parhi Attacks Due to Flaw of Protocols Used In Network Access Control (NAC), Their Solutions and Issues: A Survey April 2012 International Journal of Computer Network and Information Security 4(3).

[11] Wang, W., and Wang, H. (2011). Weakness in 802.11w and an improved mechanism on protection of management frame. International Conference on Wireless Communications and Signal Processing (WCSP), (Nov 2011), pp. 1-4.

[12] Vladimirov A.A., Gavrilenko K.V., Mikhailovsky A.A. // Addison-Wesley Professional; Illustrated edition — 2004. — C. 588.

[13] Feng Jian and Nan Tian-zhu Design, Extension and Implementation of RADIUS Client International Journal of Future Generation Communication and Networking Vol. 9, No. 5 (2016), pp. 181-188 http://dx.doi.org/10.14257/ijfgcn.2016.9.5.18 ISSN: 2233-7857.

[14] L. Galchynsky and A. Sereda, Vulnerability assessment of the RADIUS protocol based on an extended automatic model in Proc. of the XXII International Scientific and Practical Conference "Information Technologies and Security" (ITS-2022)16 November 2022, Kyiv.

[15] Object-oriented analysis and design with applications / Grady. Booch, 2nd ed. Includes bibliographical references and index. ISBN 0-201-89551.