

# [VulnWatch] OpenBSD radius authentication vulnerability

**Source:** <http://www.derkeiler.com/Mailing-Lists/VulnWatch/2004-09/0021.html>

---

*E.Bos\_at\_reseau.nl*

**Date:** 09/21/04

Date: Tue, 21 Sep 2004 08:49:40 +0200

To: vulnwatch@vulnwatch.org

**Title:** OpenBSD radius authentication vulnerability

**Summary:** Authentication can be bypassed when radius-authentication is used on OpenBSD.

**Impact:** Unauthorized access to the system

**Software:** OpenBSD 3.2 and OpenBSD 3.5 confirmed vulnerable.

**Workarounds:**

- 1) Place the Radius server on an isolated lan so that only the system (radius-client) can reach it.
- 2) Don't use radius login.

**Solutions:**

- Apply the patch released by OpenBSD.

OpenBSD 3.6 will ship with this fixed.

**Description:** When using radius authentication on OpenBSD it is possible to login on the OpenBSD when traffic from the radius-server can be spoofed. Since radius uses UDP, this is not hard to do. Radius authentication is not enabled by default on OpenBSD.

When connecting to an OpenBSD machine that does radius authentication when configured in /etc/login.conf (see man(5) login.conf and man(8) login\_radius), the OpenBSD machine will ask for userid and password. This userid and password is sent to the radius server. The radius-server will respond with either an 'REJECT' or 'ACCEPT'. More information on the protocol can be found in RfC 2865.

From this RfC, Chapter 7.1:

=====

The NAS at 192.168.1.16 sends an Access-Request UDP packet to the RADIUS Server for a user named nemo logging in on port 3 with

## VulnWatch: [VulnWatch] OpenBSD radius authentication vulnerability

password "arctangent".

The Request Authenticator is a 16 octet random number generated by the NAS.

The User-Password is 16 octets of password padded at end with nulls, XORed with MD5(shared secret|Request Authenticator).

```
01 00 00 38 0f 40 3f 94 73 97 80 57 bd 83 d5 cb
98 f4 22 7a 01 06 6e 65 6d 6f 02 12 0d be 70 8d
93 d4 13 ce 31 96 e4 3f 78 2a 0a ee 04 06 c0 a8
01 10 05 06 00 00 00 03
```

```
1 Code = Access-Request (1)
1 ID = 0
2 Length = 56
16 Request Authenticator
```

Attributes:

```
6 User-Name = "nemo"
18 User-Password
6 NAS-IP-Address = 192.168.1.16
6 NAS-Port = 3
```

The RADIUS server authenticates nemo, and sends an Access-Accept UDP packet to the NAS telling it to telnet nemo to host 192.168.1.3.

The Response Authenticator is a 16-octet MD5 checksum of the code (2), id (0), Length (38), the Request Authenticator from above, the attributes in this reply, and the shared secret.

```
02 00 00 26 86 fe 22 0e 76 24 ba 2a 10 05 f6 bf
9b 55 e0 b2 06 06 00 00 00 01 0f 06 00 00 00 00
0e 06 c0 a8 01 03
```

```
1 Code = Access-Accept (2)
1 ID = 0 (same as in Access-Request)
2 Length = 38
16 Response Authenticator
```

Attributes:

```
6 Service-Type (6) = Login (1)
6 Login-Service (15) = Telnet (0)
6 Login-IP-Host (14) = 192.168.1.3
```

=====

Since the Response Authenticator in the reply uses the Request Authenticator from the request, the client must be able to verify the 'origin', it should have a corresponding request pending.

This is where it fails. I used the following setup:

## VulnWatch: [VulnWatch] OpenBSD radius authentication vulnerability

```
[--- LAN -----]
  |||
  || OpenBSD | Radius-
  [ ] client [ ] server [ ] Server
  10.10.1.3 10.10.1.2 10.10.1.1
```

Step 1-3 is preparation phase.

- 1) Setup an environment where radius login is used and that you control.
- 2) Login via radius, sniff the packets and save the 'ACCEPT' packet.
- 3) Transform the 'ACCEPT'-packet data so it can be used by e.g. socat or hping.
- 4) From the client, login to the OpenBSD server. The OpenBSD server will send a REQUEST to the radius-server, and awaits an answer. You can either use arp-spoofing to let the OpenBSD server think another machine you have control of is the radius-server (assuming local network) or you must use a perfect timing, spoofing a packet w/ the correct source- and dest. portnumbers
- 5) You can either use arp-spoofing to let the OpenBSD server think another machine you have control of is the radius-server (assuming local network) or you must use a perfect timing, spoofing a packet w/ the correct source- and dest. portnumbers. Send the ACCEPT packet. This can be done w/ e.g. socat or hping. OpenBSD will use the ACCEPT-packet and grant login.

In the above scenario's, you don't need to know the shared secret (as you would have to when setting up another radius-server) nor the password of the account you use for logging in.

Sample messages from not-vulnerable systems:

```
-----
With FreeBSD 5.2.1, the following message is logged:
Aug 31 11:40:39 server login: rad_send_request: No valid RADIUS \
responses received
```

With Fedora Core2/pam\_radius\_auth.so  
([http://www.freeradius.org/pam\\_radius\\_auth/](http://www.freeradius.org/pam_radius_auth/)) the following message is logged:  
10.10.1.1 fails verification: The shared secret is probably incorrect.

Changelog:

10-09-2004 Informed the OpenBSD crew at 21:19 CEST  
11-09-2004 Received patch at 02:30 CEST

Author: Eilko Bos, Le Reseau B.V.

--

Le Reseau B.V.

Bieslookstraat 31

9731HH Groningen, Netherlands

Telefoon: +31 505497014 Fax: +31 505492310

Internet:

<http://www.reseau.nl>