ROLKEN

FINAL REPORT FROM

EXTERNAL PENETRATION TEST

About SecurityAware:	SecurityAware SA 555, Route de Example 8000 Zürich Switzerland Represented by Mr. Security Manager security.manager@securityaware.ch
About Rolken:	Rolken s.r.o. 護Nademlejnská 600/1 19800 Prague Czech Republic Represented by Mr. Delivery Responsible delivery.responsible@rolken.cz

24 May 2019

Document classification: Confidential

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. EXECUTIVE SUMMARY

This document is a final report from a penetration test done by Rolken s.r.o. (later only „Rolken“). Objectives for this test were:

1. to identify whether it is possible to gain access to a customer database and/or exfiltrate information from a management information system;
2. to gain access to the internal network and establish a persistent presence;
3. to test if incident response procedures will stop a targeted attack.

Out of scope was a regular vulnerability scan (done by the third party) and web application (customer portal) assessment (done in the previous quarter by the third party). Social engineering tests were also out of scope.

The test was executed as a *black-box variant*. Black-box means every acquired information was from public sources (Internet). The test was a simulation of attacker behaviour during a targeted attack against SecurityAware.

Only Internet-facing devices (with external access) were in scope and we tested only services based on a top of TCP and UDP.

Timeframe for the test was 72 hours of penetration testing and 48 hours to deliver the report.

All tests and actions were done in a controlled environment and from explicitly specified IP addresses (as defined in the offering) and only with internal Rolken resources. As requested by SecurityAware during kick-off we used only publicly available tools.

This document is intended solely for the SecurityAware to which is addressed. Access for other entities may be provided only after prior agreement and written consent by the SecurityAware.

2. STORYLINE

2.1. TARGET IDENTIFICATION

The first step during a penetration test is to acquire a list of IP addresses (targets) which are controlled and managed by SecurityAware. We did this by identifying third level domain names under securityaware.de domain.

We were not able to find any other domain used by SecurityAware by browsing public resources, source codes published on GitHub, discussion in discussion groups by SecurityAware employees (list of employees was acquired from LinkedIn and from corporate website).

We used **fierce.pl** tool for guessing third level domain records by using brute force. Utilizing this method, we found that securityaware.de uses wildcard DNS record. All DNS A records were pointed to one IP address, thus a record for notexistingdomainname.securityaware.de was translated to IP address 8.7.6.5. This IP address is pointing to public web hosting operated by MegaHosting Inc.

Using a wordlist of commonly used DNS records we identified following DNS records which were not pointing to catch-all IP:

DNS RECORD	IP ADDRESS
autodiscover.securityaware.de	1.2.3.4
mail.securityaware.de	2.3.4.5
ftp.securityaware.de	3.4.5.6

Table 1: Review of batch 1. DNS records

According to WHOIS record, DNS servers for domain securityaware.de are also hosted by MegaHosting Inc.

autodiscover.securityaware.de DNS record is also pointing to IP address operated by MegaHosting Inc. From online documentation for MegaHosting Inc. hosting company, we found out that records like autodiscover.securityaware.de are created automatically after domain registration and/or migration to MegaHosting. Since this IP address is not directly operated by SecurityAware we de-scoped this address as we do not have written permission to continue with the test on third-party assets.

Primary targets for this assignment were identified IP addresses 2.3.4.5 and 3.4.5.6.

2.2. RECONNAISSANCE AND IDENTIFICATION OF ATTACK VECTORS

After the target acquisition, we started with a port scan. We found multiple services that we could leverage for a successful attack.

We decided to test if the company's FTP server is using protection against password guessing. We found that after 10 unsuccessful logons offending IP address is blocked for 30 seconds. Since the server was also available over IPv6 we decided to utilize IPv6 subnet for password guessing to avoid blocking and to speed up password guessing. After 11 780 and circa 1 000 exhausted IP addresses we were able to login to FTP server under **login** *admin* **and password** *s3curityawaref!les*. We compiled a wordlist as a derivative from the company name and did a mutation utilizing the RSMangler tool.

From the files found on the FTP server, we found out the account is used to manage files shared with the external sales team and for pre-sale technical demo documentation. We also found an executable JAR file that presented a login screen to application SecurityAware PM.

We decided to "decompile" this file to see what is this application doing and if there are security issues we can leverage to achieve agreed goals. During quick lookout, we did not find any obvious security issues like password in the source code, however using Wireshark (packet sniffer) we found out that application is connecting to **pm-backend01.int.securityaware.de.** This DNS record translates to IP address 4.5.6.7. We also did DNS enumeration and found out that there are DNS records **pm-backend{02-07}.int.securityaware.de** pointing to IP address 5.6.7.8. and DNS records **pm-backend-test.int.securityaware.de** pointing to IP address 6.7.8.9. These IP addresses and domain names were added to the scope.

The updated list of targets:

RECORD	IP ADDRESS
mail.securityaware.de	2.3.4.5
ftp.securityaware.de	3.4.5.6
pm-backend01.int.securityaware.de	4.5.6.7
pm-backend{02-07}.int.securityaware.de	5.6.7.8
pm-backend-test.int.securityaware.de	6.7.8.9

Table 2: Review of batch 2. DNS records

We found out **pm-backend01.int.securityaware.de.** has only port 8080/TCP open. This port is used for communication over HTTPS utilizing REST API for SecurityAware PM application. We decided to add API testing to the backlog and move on to other identified targets.

Server with IP address 5.6.7.8. was showing the same behaviour as the server with IP address 4.5.6.7.

We found interesting that 6.7.8.9. has unprotected PostgreSQL database posing to the Internet and decided to look closer if we can leverage this finding to achieve our primary goal - access to the customer database.

There is no secret that PostgreSQL is using **postgres** user as "**superuser**" - this user is created after database initialization and is used to create other users.

We observed during the test run of password guessing there is no protection against this kind of attacks nor TCP wrapping and/or source IP whitelisting on server 6.7.8.9. We decided to repeat the password dictionary attack against PostgreSQL service listening on ports **5431/TCP** and **5432/TCP.**

We did the educated guess this instance of PostgreSQL is used for debugging PM application and developers are logging using some database management tool and entering the password manually. Since people do not like typing over long passwords we decided to limit the maximum size of wordlist entries to 8 characters without upper letters (as a result of password guessing on FTP server).

Final wordlist had **2 821 109 907 456** entries. To create wordlist, we used again RSManger tool. Passwords in wordlist have been ordered by statistical occurrence from 6, 7, 8 characters to lower character length.

We cracked password for postgres user on port **5432/TCP** within the 24-hour internal assigned limit. We were able to login to the database with password **230690x**.

After logging on to database we found these databases:

- postgres,
- test5,
- securityaware,
- securityaware5.

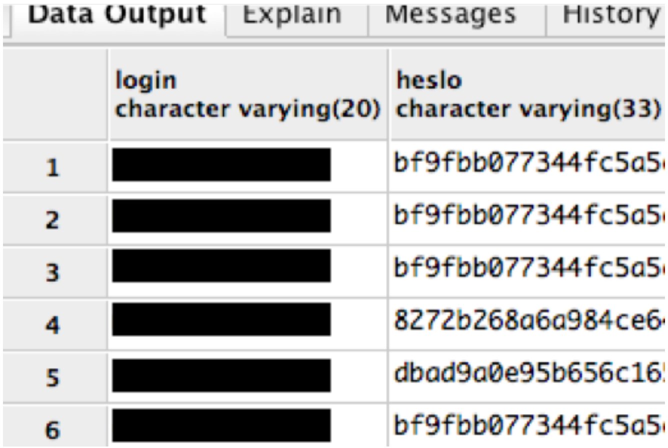
Same databases were on PostgreSQL instance on port **5431/TCP**.

2.3. CRACKING USER PASSWORDS

We found out that database **securityaware** is production copy and database **securityaware5** and others are test databases and thus we should focus on production copy. We considered as interesting the table named **user** in scheme **public**.

Using simple SQL query, we were able to found all passwords:

```
select login,heslo from user;
```



	login character varying(20)	heslo character varying(33)
1	[redacted]	bf9fbb077344fc5a5
2	[redacted]	bf9fbb077344fc5a5
3	[redacted]	bf9fbb077344fc5a5
4	[redacted]	8272b268a6a984ce6
5	[redacted]	dbad9a0e95b656c16
6	[redacted]	bf9fbb077344fc5a5

Figure 1: Screenshot of database records with passwords

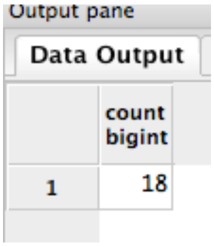
We identified password hash as a result of raw md5() function on password without cryptographical salt. For this case, we were able to utilize rainbow tables (rainbow tables are pre-generated cryptographical values stored in the database for fast searching**). In 15 minutes, we were able to check value circa 40% of passwords.**

If cryptographic salt was used, we would not be able to find that almost 40% of users have the same password in one database query.

```
d16fb36f0911f878998c136191af705e
```

This hash value is for password value **xyz**. Using this information, we were able to isolate 18 users with this password:

```
select count(*) from user where heslo='d16fb36f0911f878998c136191af705e'
```



	count bigint
1	18

Figure 2: Screenshot with number of same password records

Using this simple elimination, we minimized the number of hashes to crack and thus saving time and compute power.

We were able to crack following passwords using rainbow tables and by brute force in two hours:

USERNAME	HASH	DECRYPTED VALUE
user1	d16fb36f0911f878998c136191af705e	xyz
user2	d16fb36f0911f878998c136191af705e	xyz
user3	d16fb36f0911f878998c136191af705e	xyz
user4	--- REDACTED ---	--- REDACTED ---
user5	--- REDACTED ---	--- REDACTED ---
user6	d16fb36f0911f878998c136191af705e	xyz
user7	--- REDACTED ---	--- REDACTED ---
user8	d16fb36f0911f878998c136191af705e	xyz
user9	d16fb36f0911f878998c136191af705e	xyz
user10	d16fb36f0911f878998c136191af705e	xyz
user11	--- REDACTED ---	--- REDACTED ---
user12	--- REDACTED ---	--- REDACTED ---
--- REDACTED ---	--- REDACTED ---	--- REDACTED ---

Table 3: User names and password hashes

2.4. ACQUIRING CONTROL OVER SERVER AND GAINING ACCESS TO INTERNAL NETWORK

Using information above we were able to take control over server using database function copy() and User-defined functions we were able to…

--- REDACTED ---

2.5. LOGIN INTO EMAIL ACCOUNTS THROUGH CRACKED USERNAME AND PASSWORD

After gaining login information from cracking we tried to log in to mail server identified in first scoping. We tried to log in without a domain (user) and with domain (user@securityaware.de). **We were not able to login to the mail system.**

2.6. CONCLUSION

We found serious security issues during penetration test which can have a huge impact on SecurityAware business posture and could lead to losing the biggest company asset - process info, pricing and list of customers.

In case of a targeted attack, it would be possible to get the internal database in 12-48 hours. Therefore, it is not possible to consider the security status of servers, applications and information as sufficient. We strongly recommend implementing security countermeasures described in this report.

3. RECOMMENDATIONS

ISSUE	RECOMMENDATION
-- REDACTED --	-- REDACTED --

Table 4: Recommendations

4. DETAILED INFORMATION ABOUT FINDINGS

In compliance with **NIST SP 800-30** we evaluate findings based on probability and impact.

For information, we are providing information about identified ports and services served on the port.

-- REDACTED --

Some specific ports like 161/TCP, 53/TCP, 53/UDP, 2000/TCP and 1723/TCP indicating that the device is running MikroTik RouterOS. Since we did not have information about ownership of this device (we did not have permission to scan and attack the infrastructure of ISP) we decided to de-scope this device.

Using non-invasive techniques, we found that device has IP -- REDACTED -- and hostname -- REDACTED --. This information was acquired by connecting to port 161/TCP where is listing SNMP service which does not have changed default community name from the **test**. Using SNMP, we were able to get a list of internal IP addresses, MAC addresses and general configuration of a router. If the device is operated by SecurityAware we recommend changing settings to more secure. If the device is operated by ISP (company SuperSpeedyInternet) we recommend to contact company representative and require safer configuration.

-- REDACTED --

4.1. HOST: MAIL.SECURITYAWARE.DE

PORT NUMBER	STATUS	PROTOCOL	IDENTIFIED SERVICE
22	open	TCP	SSH / TCPwrapped
25	open	TCP	SMTP
53	open	TCP	Domain
80	open	TCP	HTTP / TCPwrapped
110	open	TCP	POP3
119	open	TCP	NNTP
143	open	TCP	IMAP
443	open	TCP	HTTPS
465	open	TCP	SMTPS
563	open	TCP	NNTPS
993	open	TCP	IMAPS

995	open	TCP	POP3S
1723	open	TCP	PPTP
2000	open	TCP	Bandwidth test
5222	open	TCP	XMPP client
5223	open	TCP	XMPPS client
8291	open	TCP	TCPwrapped

Table 5: Open ports for host mail.securityaware.de

Port 22/TCP (SSH)

Information about service

There is an SSH server on port. Service immediately closes the connection; thus, we are concluding service is TCP wrapped.

Recommended action: Information only. No action needed.

4.2. HOST: PM-BACKEND-TEST.INT.SECURITYAWARE.DE

PORT NUMBER	STATUS	PROTOCOL	IDENTIFIED SERVICE
21	filtered	TCP	FTP
22	open	TCP	SSH / TCPwrapped
53	open	TCP	Domain
80	open	TCP	HTTP
1723	open	TCP	PPTP
2000	open	TCP	Bandwidth test
2221	open	TCP	SSH
2222	filtered	TCP	Not identified
5431	open	TCP	PostgreSQL DB
5432	open	TCP	PostgreSQL DB
8080	open	TCP	HTTP
8291	open	TCP	Not identified
10050	open	TCP	Not identified
41122	filtered	TCP	Not identified

44421	open	TCP	FTP

Table 6: Open ports for host pm-backend-test.int.securityaware.de

Port 21/TCP (FTP)

Information about service

Port seems to be filtered. Based on a port number we conclude there is FTP server listening.

Recommended action: Information only. No action needed.

Port 80/TCP (HTTP)

Information about service

There is HTTP server on port. The server was identified as **Apache/2.4.7**.

Recommended action: Information only. No action needed.

Directory browsing

After accessing the IP address of server there is document root presented **(/var/www**).

Recommended action: disable directory browsing.

PHPINFO

On URL http://6.7.8.9/info-server.php is available to file calling PHP function phpinfo(), which is exposing sensitive information about the server like system paths, configuration values and other information used to precise attack.

Recommended action: remove file.

Web application

On URL http://6.7.8.9/xyz/ is available to the web application. Application is used for client access to resources provided by SecurityAware. Application was not thoroughly tested.

Recommended action: Information only. No action needed.

Debugging information presented on screen

The server is presenting debug information on the user screen. This allows an attacker to precise attack due to information about the path on the server, SQL errors etc.

Recommended action: log debug information to a log file, disable logging on screen.

Public availability of sensitive application data

The application allows free access to sensitive information like debug log on http://6.7.8.9 /xyz/log/ or source codes http://6.7.8.9/xyz/src/ or temporary files from web application http://6.7.8.9/xyz/tmp/.

This type of information is extremely valuable during attack modelling.

Recommended action: disable access to information, properly configure the web server and access controls.

Port 5431/TCP (PostgreSQL DB)

Information about service

There is PostgreSQL listening on the port.

Recommended action: Information only. No action needed.

Connection to PostgreSQL

We were able to connect to the database using login **postgres** and password **230690x.** This information was acquired by password guessing from the wordlist.

Recommended action: postgres is a privileged user, set a stronger password. If there is no special need to have database exposed hide database in VPN and/or inside of management network.

Port 5432/TCP (PostgreSQL DB)

Information about service

There is PostgreSQL listening on the port.

Recommended action: Information only. No action needed.

Connection to PostgreSQL

We were able to connect to the database using login **postgres** and password **230690x.** This information was acquired by password guessing from the wordlist.

Recommended action: postgres is a privileged user, set a stronger password. If there is no special need to have database exposed hide database in VPN and/or inside of management network.

Port 8080/TCP (HTTP)

Information about service

There is Apache Tomcat server listening on the port identified as **Apache Tomcat/Coyote JSP engine 1.1**.

Recommended action: Information only. No action needed.

Default folders

There are default folders in web server root from Apache Tomcat. These files and demo application are extending the attack surface and providing detailed info about application server.

```
/examples/servlets/index.html
/examples/jsp/snp/snoop.jsp
/examples/jsp/index.html
```

Recommended action: remove default folders.

Port 8291/TCP

Information about service

There is unknown service listening on the port. Service immediately closes the connection and thus is considered service as TCP wrapped.

Recommended action: Information only. No action needed.

Port 44421/TCP (FTP)

Information about service

There is FTP server listening on the port identified as **Synology DS210j NAS device ftpd**.

Recommended action: Information only. No action needed.

SSL certificate

The FTP server is offering during session negotiation self-signed certificate. This certificate is not trusted in any operating system.

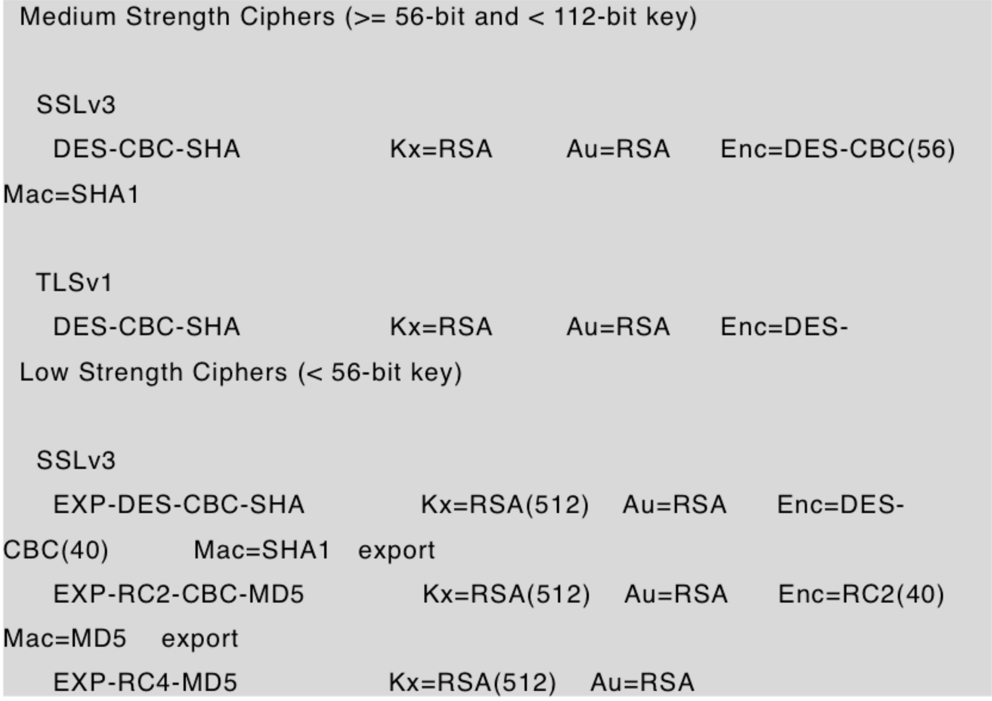
If a user is accessing NAS over an unsecured network there is a possibility for man-in-the-middle attack and a high probability that user will accept forged certificate since certificate warning is in this configuration standard behaviour.

```
|-Subject: C=TW/ST=Taiwan/L=Taipei/O=Synology Inc./OU=FTP Team/
CN=synology.com/E=product@synology.com
|-Issuer : C=TW/ST=Taiwan/L=Taipei/O=Synology Inc./OU=Certificate
Authority/CN=Synology Inc. CA/E=product@synology.com
```

Recommended action: Use internal certificate authority if present or use free service like **Let's encrypt** for a valid certificate.

Weak cryptographic algorithms

The FTP server has support for weak / medium strength cipher suites. This increases the risk of successful decryption of traffic and lowers communication security and integrity.



```
 Medium Strength Ciphers (>= 56-bit and < 112-bit key)

  SSLv3
   DES-CBC-SHA             Kx=RSA       Au=RSA     Enc=DES-CBC(56)
Mac=SHA1

  TLSv1
   DES-CBC-SHA             Kx=RSA       Au=RSA     Enc=DES-
 Low Strength Ciphers (< 56-bit key)

  SSLv3
   EXP-DES-CBC-SHA           Kx=RSA(512)   Au=RSA     Enc=DES-
CBC(40)      Mac=SHA1   export
   EXP-RC2-CBC-MD5           Kx=RSA(512)   Au=RSA     Enc=RC2(40)
Mac=MD5    export
   EXP-RC4-MD5             Kx=RSA(512)   Au=RSA
```

Figure 3: Screenshot with weak cryptographic algorithms

Recommended action: disable weak/medium strength cipher suites.

5. APPENDICES

Appendix 1: Glossary

The glossary contains an explanation of terms used in the report where we do not expect wide knowledge.

WORD	EXPLANATION
Bugtraq	full-disclosure mailing list focused on computer security including vulnerabilities, methods of penetration and bug fixes.
CVE	(Common Vulnerabilities and Exposures) list of information security announces sponsored by US-CERT and maintained MITRE Corporation.
	--- REDACTED ---

Table 7: Glossary

Appendix 2: The full table of cracked passwords

--- REDACTED ---

Appendix 3: Network infrastructure map as seen by the testing team

--- REDACTED ---