Pentest-Report CaseBox Production 08.2014

Cure53, Dr.-Ing. Mario Heiderich / Dipl.-Ing. Johannes Dahse / Norman Hippert

Index

Intro

> “Casebox is being developed jointly by HURIDOCS and KETSE.com since 2011. It started as a project to provide a sophisticated case management solution to one NGO, and was subsequently expanded to become a flexible task, document and record management system.”
>
> From https://www.casebox.org/about/

This penetration test was carried out by two testers of the Cure53 team over the period of two days. Although the test yielded five new vulnerabilities and several weaknesses, none of the findings were considered critical. Importantly, this test is a follow-up to an earlier pentest and source-code audit conducted back in late June and early July this year. The focus of this assignment was to test the security features of a CaseBox instance running on a hardened server, which entails the majority of bugs reported during and after the first test fixed. Any formerly discussed vulnerabilities originating from the first testing phase and remaining unfixed were left out of scope within the framework of this pentest. This applies to, for example, a Flash-XSS in one of the ExtJS modules, which was reported as CB-01-020[1] in the earlier communication and report.

A follow-up investigation covering the server configuration was performed after this penetration test, involving one Cure53 senior tester for another two days. The purpose was to check the server configuration for flaws and make sure the production instances run on hardened systems. This investigation yielded another twelve weaknesses that should be addressed to further secure the production machine.

Scope

- **CaseBox Production Server**
  - https://swissdata.io/demo/
  - `rmack:  casebox`
- **SSH Access**

[1] https://swissdata.io/demo/libx/ext/resources/charts.swf?YUISwfId=alert%281%29&YUIBridgeCallback=eval

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in a chronological order rather than by their degree of severity and impact, which is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier for the purpose of facilitating future follow-up correspondence.

CB-02-001 Missing Security Checks allow Privilege Escalation *(High)*

A missing security check in CaseBox allows for a privilege escalation attack. Several methods of the *CB\Browser* class are exposed by the router API. When modifying items, they fail to verify the access privileges of the current user. This can lead to arbitrary item modifications or item information disclosures. For example, the *takeOwnership()* method sets both the owner ID (*oid*) and the user ID (*uid*) of any item in the tree to the current user's ID. Consequently, a malicious user may take ownership of the item with id *25413* by sending the following request:

```
POST /demo/remote/router.php HTTP/1.1
Host: swissdata.io
X-Requested-With: XMLHttpRequest

{"action":"CB_Browser","method":"takeOwnership","data":
[["25413"]],"type":"rpc","tid":14}
```

The targeted item's id can be either obtained from the web interface or brute-forced. Once the `oid` and `uid` of this item are changed to the current user's ID, other methods that actually do check for the access privileges can be (ab)used.

Furthermore, the following methods can be accessed from the router API and are suspected to be vulnerable due to the missing security checks. They all allow specifying a target object's *id* or *pid*, even though the security permissions of a given object are not validated with the current user's ID.

Class	Method	Impact
`CB_Browser`	`saveFile`	write a file to another users' secret folder
`CB_Browser`	`uploadNewVersion`	write a file to another users' secret folder
`CB_Browser`	`toggleFavorite`	TBD
`CB_Browser`	`getObjectsForField`	leak meta-information on secret objects
`CB_Browser`	`subscribe`	none (feature currently not in use)
`CB_BrowserView`	`getChildren`	leak names of secret paths

`CB_Objects`	`getPluginsData`	leak meta-information on secret objects
`CB_Objects`	`addComment`	add comments to secret objects (Figure 1)

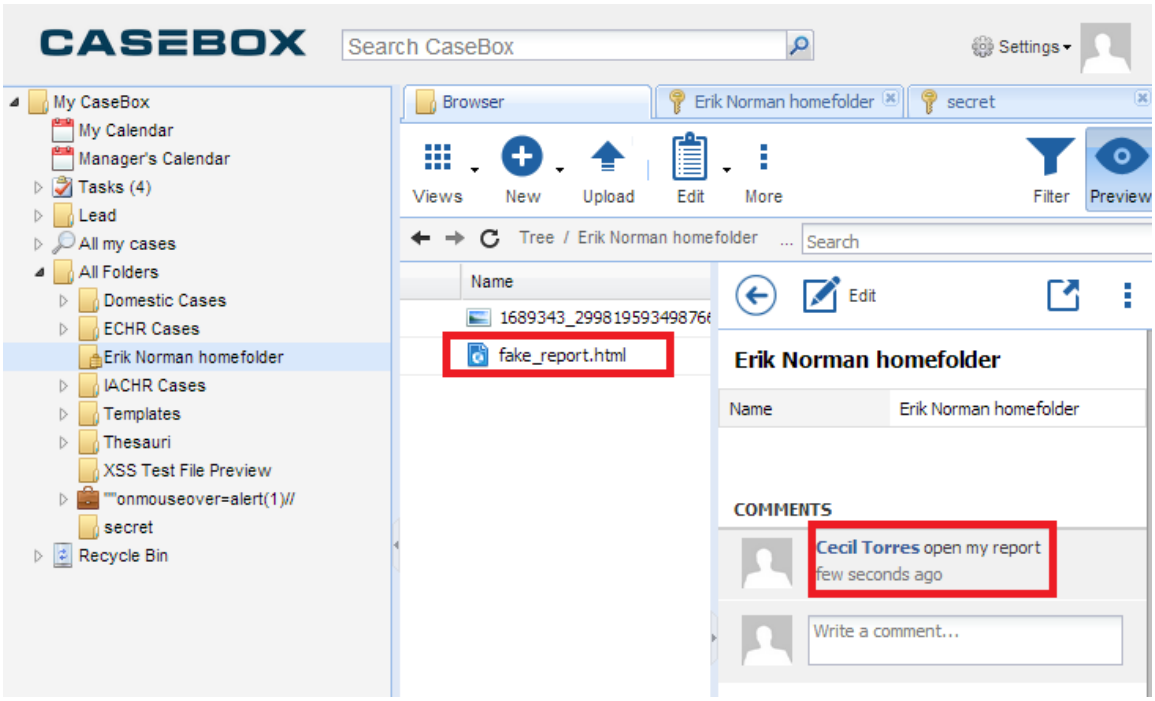

Fig. 1: Cecil Torres commented on Erik Norman's private homefolder and uploaded a file fake_report.html into his homefolder.

It is important to keep in mind that other methods might be equally vulnerable. Therefore, all accessible methods should be checked for proper privilege validation regarding the current session user's ID. It is further recommended to address the general topic of CSRF[2] to facilitate mitigation of alike attacks triggered via social engineering.

CB-02-003 User Information Disclosure *(Medium)*

Another privilege escalation issue spotted during the pentest allows attackers to illegitimately gain admin access to the CaseBox application by disclosing information and using it for brute-forcing / guessing passwords connected to the leaked usernames. The reason for this can be traced to the privilege model that CaseBox employs for certain classes and methods. More specifically, two methods of the *CB\Security* class are not limited to administrators. They can be accessed directly by any user without the privileges to obtain the lists of all user groups (*searchUserGroups*) and all active users (*getActiveUsers*):

[2] http://en.wikipedia.org/wiki/Cross-site_request_forgery

```
{"action":"CB_Security","method":"getActiveUsers","data":[[]],
       "type":"rpc","tid":14}
{"action":"CB_Security","method":"searchUserGroups","data":[[]],
       "type":"rpc","tid":14}
```

To illustrate, the following list of user groups and active users (highlighted) could be extracted from the *swissdata.io* server:

{"id":"235","name":"Administrators","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"236","name":"Managers","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"246","name":"Buenos Aires","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"234","name":"Everyone","system":"1","enabled":"1","iconCls":"icon-users"},
{"id":"249","name":"Lima","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"245","name":"London","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"242","name":"Moscow","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"243","name":"New York","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"244","name":"Paris","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"248","name":"San Francisco","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"233","name":"SYSTEM","system":"1","enabled":"1","iconCls":"icon-users"},
{"id":"247","name":"Tokyo","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"238","name":"Users","system":"0","enabled":"1","iconCls":"icon-users"},
{"id":"270","name":"Cecil Torres","system":"0","enabled":"1","iconCls":"icon-user-"},
{"id":"240","name":"Erik Norman","system":"0","enabled":"1","iconCls":"icon-user-m"},
{"id":"267","name":"Lorraine Adkins","system":"0","enabled":"1","iconCls":"icon-user-"},
{"id":"268","name":"Marc Crawford","system":"0","enabled":"1","iconCls":"icon-user-"},
{"id":"269","name":"test" test"","system":"0","enabled":"1","iconCls":"icon-user-"},
{"id":"1","name":"Administrator","system":"0","enabled":"1","iconCls":"icon-user-m"},
{"id":"266","name":"Robin Stone","system":"0","enabled":"1","iconCls":"icon-user-"}

Based on this information, one can discover credentials of additional users, demonstrating a possible risk stemming from this information leakage:

```
ctorres:casebox
mcrawford:casebox
```
`enorman:casebox` (a website Administrator)

Moreover, it is possible to abuse the *searchUserGroups* method to extract all email addresses of the users. The *query* field of the *data* parameter is embedded in the SQL query, namely in the WHERE clause, which is used for querying the *user_groups* table. The *searchField* column includes the email address of the user.

CB/Security.php, line 139
```
 $where[] = 'searchField like $1';
 $params[] = ' %'.trim($p['query']).'% ';
```

The SQL operator in use is `LIKE`, and, as it allows wildcard characters (such as _ and %), email addresses can be enumerated character by character:

```
{"action":"CB_Security","method":"searchUserGroups","data":[{"query":"da
%@"}],"type":"rpc","tid":16} // result shown, email with string da exists
{"action":"CB_Security","method":"searchUserGroups","data":[{"query":"db
%@"}],"type":"rpc","tid":16} // no result, email with string db does not exist
```

CB-02-004 WebDav Software causing crashes and Privilege Escalation *(High)*

At present, users can directly edit documents with WebDAV. Regrettably, the software is flawed and full of bugs, and as such must be considered a security risk. In its current state, the tool crashes every time that a user wishes to edit an ODT document (LibreOffice[3]). It appears that, once installed, the software does not discover the binary. Thus, it attempts to run an executable that is not present and thereby denies service by crashing. It is assumed that this crash is exploitable on older versions of Windows. The stack trace looks as follows:

```
0:000> .ecxr
eax=00000000 ebx=0013efbc ecx=00000000 edx=00000033 esi=00243088 edi=00000000
eip=006007a0 esp=0013ee64 ebp=0013eec0 iopl=0         nv up ei ng nz ac pe cy
cs=001b  ss=0023  ds=0023 es=0023  fs=003b  gs=0000             efl=00010297
006007a0 8b01            mov eax,dword ptr [ecx]   ds:0023:00000000=????????
0:000> !exploitable -v
HostMachine\HostUser
Executing Processor Architecture is x86
Debuggee is in User Mode
Debuggee is a user mode small dump file
Event Type: Exception
Exception Faulting Address: 0x0
Second Chance Exception Type: STATUS_ACCESS_VIOLATION (0xC0000005)
Exception Sub-Type: Read Access Violation

Faulting Instruction:006007a0 mov eax,dword ptr [ecx]

Basic Block:
       006007a0 mov eax,dword ptr [ecx]
       Tainted Input Operands: ecx
       006007a2 call dword ptr [eax+28h]
       Tainted Input Operands: eax, ecx

Exception Hash (Major/Minor): 0x57181662.0x4a6c0f06

Stack Trace:
Unknown
Unknown
mscorwks!CallDescrWorker+0x33
mscorwks!CallDescrWorkerWithHandler+0xa3
mscorwks!MethodDesc::CallDescr+0x19c
mscorwks!MethodDesc::CallTargetWorker+0x1f
mscorwks!MethodDescCallSite::Call_RetArgSlot+0x1a
mscorwks!ClassLoader::RunMain+0x223
mscorwks!Assembly::ExecuteMainMethod+0xa6
mscorwks!SystemDomain::ExecuteMainMethod+0x45e
```

[3] http://www.libreoffice.org/

```
mscorwks!ExecuteEXE+0x59
mscorwks!_CorExeMain+0x15c
mscoreei!_CorExeMain+0x10a
mscoree!ShellShim__CorExeMain+0x99
mscoree!_CorExeMain_Exported+0x8
kernel32!BaseThreadInitThunk+0xe
ntdll!__RtlUserThreadStart+0x70
ntdll!_RtlUserThreadStart+0x1b
Instruction Address: 0x00000000006007a0

Description: Data from Faulting Address controls Code Flow
Short Description: TaintedDataControlsCodeFlow
Exploitability Classification: PROBABLY_EXPLOITABLE
Recommended Bug Title: Probably Exploitable - Data from Faulting Address
controls Code Flow starting at Unknown Symbol @ 0x00000000006007a0 called from
mscorwks!CallDescrWorker+0x0000000000000033 (Hash=0x57181662.0x4a6c0f06)

The data from the faulting address is later used as the target for a branch.
```

Further, the tool attempts to convince users to adopt bad practices for installation purposes. While it requires an administrative user during installation, it does not provide a cleanly documented routine for deinstalling and ultimately performs registry changes that are undocumented. They also allow any other website in any other browser to arbitrarily launch URIs with the use of the “cbdav” scheme. Eventually, any website can now provoke the crash by simply executing the following simple JavaScript code:

```
<script>location='cbdav:.odt'</script>
```

This is not only a security risk but it also crucially allows other websites to remotely find out whether the targeted user has the tool installed. A user would be instantly known to be using CaseBox with WebDav functionality. The latter unnecessarily extends the attack surface, making it desirable to either re-work the tool entirely to enhance its security-relevant robustness and reliability, or, alternatively, remove it from the website completely.

Note: Given the time constraint, only the Windows-based version was checked. A security test for the OSX version was not performed.

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid attackers in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

CB-02-002 Apache SOLR Exception Information Disclosure *(Low)*

The reported Apache SOLR injection (CB-01-012) seems to be fixed, however, it is still possible to trigger a SOLR exception that outputs a stack trace. If the *dateStart* parameter is set but there is no *dateEnd* parameter, the SOLR query fails.

```
POST /demo/remote/router.php HTTP/1.1
Host: swissdata.io
X-Requested-With: XMLHttpRequest

{"action":"CB_Browser","method":"getObjectsForField","data":
[{"sort":"name","dir":"[","path":"/","source":"tree","renderer":"listObjIcons","
autoLoad":true,"scope":24265,"value":24274,"multiValued":true,"editor":"form","d
ateStart":"a"}],"type":"rpc","tid":82}
```

The stack trace reveals path names and function names. Therefore, it should not be echoed to the user in the production mode.

CB-02-005 Possible Passive XSS in MSIE using poisoned PDF *(Medium)*

CaseBox allows a user to upload PDF files which may later be available for download or displayed in preview for other users. The preview feature uses an HTML `<object>` element to simply embed the loaded PDF in the CaseBox HTML, with the PDF file residing on the same domain as the application. The feature works well on Chrome but appears broken on Firefox and MSIE. Rather than the actual PDF file, the string “PDF” is being displayed. This is considered a bug and needs to be fixed.

Once the bug is fixed, however, a security problem in Acrobat Reader 10 (and some newer versions) will turn this feature into a passive XSS vulnerability for the user employing MSIE. A PDF can contain links and these links can point to JavaScript URIs. Upon being clicked, those will be blocked by Acrobat Reader and a user will be shown a security notification. Conversely, if an attacker uses a VBScript URI (`vbscript:alert(1)`)[4], the Reader will happily delegate the request to the browser and execute the VBScript in the context of the embedding domain.

It is recommended to disallow a direct embedding of the user-controlled PDFs. A library, such as the *PDF.js,* should be used instead for rendering the PDF and displaying it safely[5]. The current framework of embedding PDFs directly delegates the security responsibility to the installed reader software which, given the track record of these tools,

[4] http://msdn.microsoft.com/en-us/library/t0aew7h6%28v=vs.84%29.aspx
[5] http://mozilla.github.io/pdf.js/

is not recommended. In addition, it should be considered to store uploads on a different domain, in attempts to having the SOP as an aid against passive and active XSS attacks stemming from the uploaded active files. Note that ZIP files, among others, will be considered first-class citizens soon[6] for the fact that they will introduce XSS attacks from their compressed contents.

Note that under current conditions, a Firefox-only JAR-XSS attack *cannot* be successfully performed here. This is due to the fact that the download script is sending proper Content-Disposition headers, making the script safe against this particular variation:

```
<iframe src="jar:https://swissdata.io/demo/download.php?id=25306!/test.html">
</iframe>
```

CB-02-006 World Readable SSL Certificates can lead to MitM attacks *(Medium)*

It was found that the swissdata.io SSL certificates are world readable. They can therefore be read by each and every user and process of the system. Furthermore the PHP *open_basedir* restriction is set to `/var/www/casebox`, which could allow an attacker to extract those certificates with an LFI/RCE/SQLi vulnerability. More than one issue of this kind was already found in the first audit (see e.g. CB-01-001, CB-01-013), meaning that this a plausible attack scenario.

```
-rw-r--r-- 1 www-data www-data  944 Aug  5 19:38 casebox.crt
-rw-r--r-- 1 www-data www-data  696 Aug  5 19:38 casebox.csr
-rw-r--r-- 1 www-data www-data  887 Aug  5 19:38 casebox.key
-rw-r--r-- 1 www-data www-data  963 Aug  5 19:38 casebox.pem
-rw-r--r-- 1 www-data www-data 1.8K Aug  5 20:12 swissdata.io.crt
-rw-r--r-- 1 www-data www-data 1.7K Aug  5 20:07 swissdata.io.key
```

The SSL certificates should not only be moved from the upper webroot to a secure location like `/etc/ssl/private`, but permissions need be set to be at least “640”.

CB-02-007 Process Core Dumps can lead to potential Data Disclosure *(Low)*

Core dumps are not properly configured on the system. A memory image taken at the time when the operating system terminates an application is included in the core dump. As it stands, the memory image can contain sensitive data and is generally useful solely for the developers who are trying to debug problems. To disable core dumps for all users, the following line should be added to the following file:

`/etc/security/limits.conf`:

```
* hard core 0
```

[6] http://wiki.whatwg.org/wiki/Zip

CB-02-008 UMASK settings are too lax and allow for world readable files *(Medium)*

A *umask* setting of 022 was discovered to be in use. This setting generates world readable files per default and is considered bad practice, leading to problems like CB-02-006.

A umask setting of at least 027 should be the system's default. To accomplish this, two files have to be modified:

```
/etc/login.defs:
UMASK 027

/etc/pam.d/common-session:
session optional pam_umask.so
```

CB-02-009 Apache Server discloses Version Number *(Low)*

The Apache web server is configured to include its version number in the HTTP response headers. Although it is not a vulnerability in itself, this does provide an attacker with additional information, possibly valuable for subsequent searches for known vulnerabilities.

The following entry should be changed:

```
/etc/apache2/conf.d/security
ServerTokens Prod
```

CB-02-010 Unnecessary Apache Modules are enabled *(Low)*

Certain Apache modules should be disabled, pending their verification as dispensable. The fact that they are known to cause problems if not properly configured or used should lead to their removal:

```
mod_status
mod_cgid
mod_autoindex
```

CB-02-011 Multiple Processes are running as root *(Medium)*

Multiple processes have been identified as running with the super-user privileges. This is considered bad practice and can lead to a full system compromise if an attacker is able to exploit a vulnerability or misconfiguration in those programs.

The following processes should be initiated under an unprivileged user:

```
/usr/lib/libreoffice/program/soffice.bin
java -Dsolr.solr.home=multicore -jar start.jar
```

CB-02-012 The TCP/IP configuration should be hardened *(Low)*

Multiple best practice settings are not met within a standard linux installation. The following settings should be added to `/etc/sysctl.conf`:

```
net.ipv4.icmp_echo_ignore_broadcasts=1

net.ipv4.conf.default.accept_redirects=0
net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.default.send_redirects=0
net.ipv4.conf.all.send_redirects=0

net.ipv4.conf.all.secure_redirects = 0
net.ipv4.conf.default.secure_redirects = 0
net.ipv4.conf.all.accept_source_route=0
net.ipv4.conf.default.accept_source_route=0
net.ipv4.conf.all.forwarding=0
net.ipv4.conf.default.forwarding=0
net.ipv4.conf.all.mc_forwarding=0
net.ipv4.conf.default.mc_forwarding=0
net.ipv4.conf.all.rp_filter=1
net.ipv4.conf.default.rp_filter=1
net.ipv4.conf.all.log_martians=1
net.ipv4.conf.default.log_martians=1
net.ipv4.tcp_max_syn_backlog=1280
net.ipv4.tcp_syncookies=1

# disable ipv6 as it is not used on this system
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
net.ipv6.conf.eth0.disable_ipv6 = 1
```

Keep in mind that one has to run the command “`sysctl -p`” upon introducing new settings for the changes to be effective.

CB-02-013 SSH Server uses a weak Server Key Length *(Medium)*

The SSH Server is configured to use a key length of only 768 Bits. This is considered too short for ensuring the operations that are cryptographically-secure.

In order to address this, change the following line in `/etc/ssh/sshd_config`:

```
ServerKeyBits 4096
```

Afterwards, one has to run the following commands to regenerate the SSH server keys:

```
/bin/rm -v /etc/ssh/ssh_host_*
dpkg-reconfigure openssh-server
```

One should consider adding the following settings to the SSH Server configuration:

```
KeyRegenerationInterval 3600
StrictModes yes
UseDNS no
Ciphers aes256-gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes128-
ctr,blowfish-cbc
MACs umac-128-etm@openssh.com,hmac-sha2-512,hmac-sha2-256,hmac-ripemd160
```

Please be sure to have an old SSH connection open at any time to verify all changes you have made. Otherwise, getting locked out from the server is a real possibility.

CB-02-014 MySQL allows for local file access *(Medium)*

MySQL gives all users an option to employ the LOAD DATA INFILE[7] command, which could allow an attacker to read arbitrary files through an SQLi vulnerability. Furthermore, the CaseBox database user does have file privileges granted, which could be used to write arbitrary files to the file system[8]. This should be avoided.

Please add the following line to `/etc/mysql/my.cnf`:

```
set-variable=local-infile=0
```

Run the following SQL command as root:

```
REVOKE FILE on *.* FROM 'casebox'@'localhost';
```

CB-02-015 Missing File Integrity and Rootkit Checking *(Medium)*

As a security-in-depth measure, one should consider running tools like *rkhunter*[9] or *lynis*[10] regularly to timely detect any system anomalies or compromises.

As for *rkhunter*, the following command should be run as a follow-up to installation in order to set up the file property database and monitor changes to the filesystem:

```
rkhunter --update
rkhunter --propupd
```

Keep in mind that those properties change with system updates like `apt-get update/upgrade`. One should run *rkhunter* manually before the system update is applied to ensure that no compromise occurred. Then, the property database should be rebuilt with the commands noted above.

CB-02-016 Temporary Directory should be more restrictive *(Medium)*

The temporary directory often constitutes a starting point for attackers. If they have no other privileges, they often run their executables from this location. This is why the */tmp* directory should be moved to a separate partition on a tmpfs, with its size limited and no privileges allowing a start of programs from this directory.

---

7 http://dev.mysql.com/doc/refman/5.0/en/load-data-local.html
8 http://websec.wordpress.com/2007/11/17/mysql-into-outfile/
9 http://rkhunter.sourceforge.net/
10 http://rootkit.nl/projects/

To achieve that, the following line has to be added to `/etc/fstab`:

```
none /tmp tmpfs rw,noexec,nosuid,nodev,size=250000000 0 0
```

This restricts the size to roughly 250MB. While it may be necessary to adjust this parameter, keep in mind that the tmpfs size is taken from the system memory, which is limited to 2GB with the current server configuration.

CB-02-017 No Linux Security Module was identified *(Medium)*

As a security-in-depth feature and hindrance of attackers' work, one should consider using a Linux Security Module, such as SELinux[11]. With a proper configuration, an attacker might still have problems gaining complete root access despite having control over a process running as root. The configuration can be time consuming and the additional effort should be weighed against the additional security gain. A beginner's guide for Debian can be found in the *Debian Administrator's Handbook*[12].

CB-02-018 PHP open_basedir to protect multiple instances from another *(Medium)*

It is recommended to isolate multiple CaseBox silos that reside on the same server from each other. This would help in making sure that a compromise on one Silo does not necessarily affect the other silos hosted on that same machine. It is recommended to deploy a per-virtual host configuration directive that sets the PHP *open_basedir*[13] setting to the targeted silo's directory. In doing so, one ensures that an attacker can only access data from the compromised silo, remaining unable to leave the directory and access other silos[14].

It should be also considered to use per-application *php.ini* files to be able to securely host multiple instances of CaseBox on the same machine without risking an attacker snooping data from one compromised silo to another[15]. This protection might however be insufficient if the attacker already gained control over one of the co-hosted CaseBox instances and further architectural measurements should be undertaken to fully cover this rather specific threat model.

[11] http://en.wikipedia.org/wiki/Security-Enhanced_Linux
[12] http://debian-handbook.info/browse/stable/sect.selinux.html
[13] http://php.net/manual/en/ini.core.php#ini.open-basedir
[14] http://www.acunetix.com/websitesecurity/php-security-3/
[15] http://www.howtoforge.com/how-to-specify-a-custom-php.ini-for-a-website-apache2-with-mod_php

Conclusion

This report described findings of a follow-up project, complementing and expanding a full source-code audit against the CaseBox software that took place in late June and early July 2014. The here-described test was carried out against a hardened server system including SELinux and firewall setup. The goal was to use an unauthenticated user - or a user with a small set of privileges - and attempt to access to the login-protected areas of the application.

A crucial task was to examine whether it was still possible to escalate privileges on the application, server and user-agents, as well as clients of other people working on the same CaseBox instance. In addition, the absence of HTTP Security headers on the entire installation was noted and discussed. While cookies were properly flagged as secure and HTTPonly[16], an attacker could still abuse the lack of HTTP Security headers to perform click-jacking attacks[17] and cause UI-Redressing hazard. Any CaseBox instance should set the aforementioned headers in the application logic and not rely on the server to take care of this setting. The lack of security headers was mentioned in the first report under the CB-01-030 issue description. When the penetration test was completed, an investigation was carried out against the configuration of the hosting server by giving Cure53 shell access with root privileges. While those efforts helped to uncover additional twelve issues, not a single one among them could be considered critical in terms of the degree of severity. A full range of issues and resulting recommendations were outlined in this Report.

It needs to be pointed out that during the investigation phase dedicated to the pre-installed server, the question was raised about feasibility and security-considerations of running multiple instances of CaseBox in parallel. Capacity to adequately isolate a single instance (called “silo”) was questioned and challenged. While it is possible to install basic protection mechanisms by properly configuring PHP runtime[18],[19] and VirtualHost setup[20], a perfect isolation from various CaseBox silos residing on the same machine is a complex task. It was deemed not recommended to undertake this endeavor without additional design and implementation considerations. Particularly a scenario when an attacker simultaneously fully controls a CaseBox silo on the one hand, but must not be able to compromise other silos, on the other hand, constitutes a thread model that was not considered. As such, it requires further investigation.

Ultimately CaseBox got significantly stronger since the first penetration test and SCA, yet there is still a lot of work to be done. It is especially visible in the need to harden the system sufficiently against advanced attacks. The Flash XSS is still present, CSRF is still an unresolved issue, and information disclosure remains possible, leading to privilege escalations and general expansion of the attack surface. At the same time, the two tests

[16] http://en.wikipedia.org/wiki/HTTP_cookie#Secure_cookie
[17] http://en.wikipedia.org/wiki/Clickjacking
[18] https://www.owasp.org/index.php/PHP_Configuration_Cheat_Sheet
[19] https://github.com/sektioneins/pcc
[20] http://httpd.apache.org/docs/2.2/vhosts/examples.html

point out the directionality of the desirable changes by drawing attention to the major weaknesses of CaseBox. The reports also specify the fixes and mitigation approaches that match and appease the risks present under the given threat model.

We extend our gratitude to the consultants from the Open Society Foundations Information Program, Tom Longley and Sam Smith, for suggesting the audit in the first place and facilitating a smooth liaising between the teams throughout this process. Cure53 would like to equally thank the entire CaseBox Team for their support and assistance during this assignment.