بسمه تعالی

سازمان فناوری اطلاعات ایران

معاونت امنیت فضای تولید و تبادل اطلاعات

مرکز ماهر

مقاوم سازی امنیتی Cassandra

فهرست مطالب

پیشگفتار

در این مستند، مقاوم‌سازی امنیتی Cassandra نسخه ۳,۱۱ مورد بررسی قرار می‌گیرد. به این منظور، نحوه وارسی و ایمن‌سازی مقادیر و تنظیمات مربوط به تعداد زیادی از پارامترهای تاثیرگذار در عملکرد این پایگاه‌داده معرفی می‌گردند. در مورد هر پارامتر، کاربرد آن پارامتر به طور مختصر بیان می‌شود، ارزش امنیتی پارامتر ذکر می‌گردد، نحوه آگاهی از مقدار کنونی پارامتر مشخص می‌شود، و در نهایت چگونگی مقداردهی امن پارامتر نشان داده می‌شود.

بررسی پارامترهای مربوط به مقاوم‌سازی Cassandra در شش فصل متمایز صورت می‌گیرد. . در فصل اول، نیازمندی‌های مرتبط با امن‌سازی محیط اجرا ارائه می‌شود. فصل دوم به تشریح پارامترهای نصب و پیکربندی Cassandra می‌پردازد. فصل سوم به معرفی محدودیت‌هایی اختصاص دارد که باید بر روی فرآیند اتصال و ورود کاربران به Cassandra اعمال گردد. در فصل چهارم، پارامترهای کنترل دسترسی و مجازشماری مورد بررسی قرار می‌گیرند. پارامترهای مربوط به رویدادنگاری امن در فصل پنجم بررسی می‌شوند. در فصل ششم، تنظیمات مربوط به رمزنگاری مورد توجه قرار می‌گیرند. در پایان نیز، نحوه اجرای اسکریپت‌ها و اعمال تنظیمات مورد نیاز برای مقاوم‌سازی پایگاه‌داده بیان می‌گردد.

۱ امن‌سازی محیط اجرا

در این فصل، پیشنهاداتی برای مقاوم‌سازی محیط اجرای Cassandra ارائه می‌شود. در این راستا، پیشنهاد شده است که آخرین نسخه‌های جاوا و پایتون که با نیازمندی‌های سازمان سازگار هستند، بر روی سرور نصب شوند. علاوه بر این تاکید بر آن است که سرویس Cassandra باید توسط کاربر غیر ریشه با حداقل مجوزها اجرا گردد و فایل تنظیمات، دایرکتوری ذخیره داده و فایل‌های رویدادنگاری نیز باید مجوزها و دسترسی‌های مناسب و امنی داشته باشند.

۱-۱ نصب آخرین نسخه جاوا

پیش نیاز نصب Cassandra، نصب جاوا است. جدیدترین نسخه‌ی جاوا که با نیازمندی‌های عملیاتی سازمان منطبق است، باید نصب شود.

تهدید/توجیه امنیتی:

نصب آخرین نسخه‌ی جاوا SDK می‌تواند به کاهش احتمال وجود آسیب‌پذیری در نرم‌افزار کمک کند. نسخه نصب شده باید مطابق با نیازمندی‌های سازمان باشد. همچنین، این اطمینان باید حاصل شود که نسخه نصب شده دارای پشتیبانی بوده و به‌روزرسانی‌های منظمی برای رفع آسیب‌پذیری‌ها انجام می‌شود.

اطلاع از وضعیت فعلی:

با استفاده از دستور زیر نسخه نصب شده‌ی جاوا به دست می‌آید.

```
java -version
```

در صورتی که نسخه‌ی قدیمی/پشتیبانی نشده جاوا نصب شده باشد، با استفاده از خروجی دستور فوق می‌توان متوجه آن شد.

مقاوم‌سازی:

۱. نسخه قدیمی/پشتیبانی نشده جاوا را حذف کنید.
۲. آخرین نسخه سازگار Java JDK یا OpenJDK را از طریق یکی از لینک های زیر دانلود کنید.

https://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html#javasejdk
http://openjdk.java.net/

۳. مطابق با دستورالعمل نصب، نسخه دانلود شده را نصب کنید.

۱-۲ نصب آخرین نسخه پایتون

پیش نیاز نصب Cassandra، نصب پایتون است. جدیدترین نسخه‌ی پایتون که با نیازمندی‌های عملیاتی سازمان منطبق است، باید نصب شود.

تهدید/توجیه امنیتی:

نصب آخرین نسخه‌ی پایتون می‌تواند به کاهش احتمال وجود آسیب‌پذیری در نرم‌افزار کمک کند. نسخه نصب شده باید مطابق با نیازمندی‌های سازمان باشد. همچنین، این اطمینان باید حاصل شود که نسخه نصب شده دارای پشتیبانی بوده و به‌روزرسانی‌های منظمی برای رفع آسیب‌پذیری‌ها انجام می‌شود.

اطلاع از وضعیت فعلی:

با استفاده از دستور زیر نسخه نصب شده‌ی پایتون به دست می‌آید.

```
python -V
```

در صورتی که نسخه‌ی قدیمی/پشتیبانی نشده پایتون نصب شده باشد، با استفاده از خروجی دستور فوق می‌توان متوجه آن شد.

مقاوم‌سازی:

۱. نسخه قدیمی/پشتیبانی نشده پایتون را حذف کنید.
۲. آخرین نسخه سازگار پایتون را از سایت زیر دانلود کنید.

```
https://www.python.org/downloads/
```

۳. مطابق با دستورالعمل نصب، نسخه دانلود شده را نصب کنید.

۱-۳ اجرای سرویس Cassandra توسط کاربر غیر ریشه

اگرچه پایگاه‌داده Cassandra می‌تواند توسط کاربر ریشه اجرا شود ولی می‌بایست این اجرا توسط کاربر غیر ریشه صورت پذیرد.

تهدید/توجیه امنیتی:

یکی از بهترین روش‌ها برای کاهش در معرض حمله قرار گرفتن ســـرور، ایجاد کاربر و گروه یکتا و غیر ممتاز[1] برای برنامه کاربردی سرور است. بهترین روش این است که این اطمینان حاصل شود که فرآیندها و برنامه‌های کاربردی توسط کاربری با کمترین مجوزها و امتیازات اجرا می‌شوند.

اطلاع از وضعیت فعلی:

به سروری که Cassandra بر روی آن در حال اجرا است وارد شده و دستور زیر را اجرا کنید:

```
ps -aef | grep cassandra | grep java | cut -d' ' -f1
```

خروجی دستور فوق نشان می‌دهد که چه کسی Cassandra را اجرا می‌کند. کاربر اجراکننده نباید کاربر ریشه یا کاربری با مجوزها و امتیازات بیش از حد باشد.

مقاوم‌سازی:

برای Cassandra، گروهی ایجاد کنید:

```
sudo groupadd cassandra
```

کاربری برای اجرای Cassandra و فرآیندهای مربوط به آن ایجاد کنید:

```
sudo useradd -m -d <DIRECTORY_WHERE_CASSANDRA_INSTALLED> -s /bin/bash
-g
cassandra -u <USERID_NUMBER> cassandra
```

مقدار <DIRECTORY_WHERE_CASSANDRA_INSTALLED> با مســـیر کامل محلی که فایل‌های دودویی Cassandra در آنجا نصب شده است، جایگزین شود. همچنین مقدار <USERID_NUMBER> با عددی که در حال حاضر روی سرور وجود ندارد، جایگزین گردد.

۴ - ۱ پیکربندی فایل تنظیمات

فایل پیکربندی cassandra.yaml در واقع فایل پیکربندی اصلی در پایگاه داده Cassandra است و از این رو حفاظت از این فایل از اهمیت بسزایی برخوردار است. هنگامی که Cassandra نصب می‌شود، با توجه به روش نصـب، این فایل در مکان‌های متفاوتی قرار می‌گیرد. در صـورتی که بسـته[2] Cassandra از انباره[3] نصـب شـود،

[1] Unprivileged
[2] Package
[3] Repository

فایل در محل /etc/cassandra قرار می‌گیرد و در غیر این صورت با دانلود کد برنامه و کامپایل و نصب آن، فایل cassandra.yaml در مسیر نصب و دایرکتوری conf ذخیره می‌شود. در اینجا فرض بر آن است که بسته Cassandra از انباره دانلود و نصب شده است.

تهدید/توجیه امنیتی:

از آنجایی که در این فایل تنظیمات اصلی Cassandra وجود دارد، تنها مالک این فایل یعنی root که مدیر سیستم است، حق تغییر این فایل را دارد. پس باید دقت کرد که اولاً مالک این فایل root باشد، ثانیا مجوز نوشتن تنها به مالک داده‌شده باشد.

اطلاع از وضعیت فعلی:

پس از نصب Cassandra بر روی سیستم عامل اوبونتو، مسیر پیش‌فرض برای فایل تنظیمات اصلی Cassandra، مسیر زیر است:

```
/etc/cassandra/cassandra.yaml
```

با استفاده از دستور زیر می‌توان حقوق دسترسی مربوط به فایل تنظیمات اصلی Cassandra را مشاهده نمود:

```
ls -lh /etc/cassandra/cassandra.yaml
```

انتظار می‌رود، حقوق دسترسی، مالک و گروه کاربری مالکیت به صورت زیر باشند:

```
rw-r--r--  root root
```

مقاوم‌سازی:

دستورات زیر به ترتیب مالکیت، گروه کاربری مالکیت و حقوق دسترسی به فایل را به صورت امن مشخص می‌نمایند:

```
sudo  su
chown -R root /etc/cassandra/cassandra.yaml
chgrp -R root /etc/cassandra/cassandra.yaml
chmod 644 /etc/cassandra/cassandra.yaml
```

۵-۱ پیکربندی دایرکتوری ذخیره داده

در پایگاه داده Cassandra، می‌توان مسیر پیش فرض داده‌های اصلی را با استفاده از دستور زیر به دست آورد:

```
cat /etc/cassandra/cassandra.yaml | grep -A 1 data_file_directories:
```

خروجی دستور فوق برای نسخه انتخابی گزارش به صورت زیر می‌باشد:

```
/var/lib/cassandra/data/
```

تهدید/توجیه امنیتی:

برخلاف فایل‌های دیگر، مدیر پایگاه داده نباید مالک این فایل با شد. مالکیت فایل باید متعلق به کاربری بدون هرگونه حقوق ممتاز (مثلا کاربری با نام cassandra) باشـــد؛ زیرا این کاربر اجازه انجام هیچ عملیاتی داخل سیستم لینوکس را ندارد. همچنین علاوه بر مدیر، هیچ‌کس دیگری نیز حق تغییر یا اجرای این فایل را نباید داشته باشد.

اطلاع از وضعیت فعلی:

با استفاده از دستور زیر می‌توان حقوق دسترسی مربوط به دایرکتوری ذخیره داده را مشاهده نمود:

```
ls -lh /var/lib/cassandra/data/
```

حقوق دســـترســـی، مالک و گروه کاربری مالکیت فایل های موجود در دایرکتوری ذخیره داده به صـــورت زیر می‌باشند:

```
rw-r--r--  cassandra cassandra
```

همچنین حقوق د ستر سی، مالک و گروه کاربری مالکیت دایرکتوری‌های موجود در دایرکتوری ذخیره داده به صورت زیر می‌باشند:

```
rwxr-xr-x  cassandra cassandra
```

مقاوم‌سازی:

د ستورات زیر به ترتیب مالکیت، گروه کاربری مالکیت و حقوق د ستر سی به داده‌های ا صلی را به صورت امن تعیین می‌نماید.

```
sudo  su
chown -R cassandra /var/lib/cassandra/data
chgrp -R cassandra /var/lib/cassandra/data
find /var/lib/cassandra/data -type f -exec chmod 644 {} \;
find /var/lib/cassandra/data -type d -exec chmod 755 {} \;
```

۶-۱ پیکربندی فایل‌های رویدادنگاری

در پایگاه داده Cassandra، مسیر فایل‌های رویدادنگاری را می‌توان از فایل /etc/cassandra/logback.xml استخراج کرد. محتوای این فایل شامل تنظیماتی به صورت زیر است که مسیر فایل‌های رویدادنگاری را مشخص می‌کنند.

```
<file>${cassandra.logdir}/system.log</file>
 …
<file>${cassandra.logdir}/debug.log</file>
```

پارامتر cassandra.logdir به صورت پیش‌فرض برابر مقدار زیر است:

```
/var/log/cassandra/
```

تهدید/توجیه امنیتی:

هیچ کاربری به جز cassandra نباید حق نو شتن روی فایل‌های رویدادنگاری را دا شته با شد. رعایت این مورد امنیتی از نشت اطلاعات این فایل‌ها جلوگیری می‌کند.

اطلاع از وضعیت فعلی:

برای اطلاع از وضعیت فعلی مجوزهای مربوط به فایل‌های رویدادنگاری از دستور زیر استفاده می‌شود:

```
ls -lh /var/log/cassandra/
```

حقوق دسترسی، مالک و گروه کاربری مالکیت فایل‌های موجود در دایرکتوری فایل‌های رویدادنگاری به صورت زیر می‌باشند:

```
rw-r--r--  cassandra cassandra
```

مقاوم‌سازی:

دستورات زیر به ترتیب مالکیت، گروه کاربری مالکیت و حقوق دسترسی به فایل‌های رویدادنگاری را به صورت امن مشخص می‌نمایند:

```
sudo  su
chown -R cassandra /var/log/cassandra/
chgrp -R cassandra /var/log/cassandra/
find /var/log/cassandra -type f -exec chmod 644 {} \;
```

۱-۷ جمع‌بندی

در این فصل به تشریح پارامترهای امنیتی محیط اجرای سمپاد که به طور مستقیم بر عملکرد آن تاثیرگذار است، پرداختیم. در این راستا، نسخه‌هایی از جاوا و پایتون که بر روی سرور می‌بایست نصب شوند و همچنین پیکربندی فایل تنظیمات، دایرکتوری ذخیره داده و فایل‌های رویدادنگاری مورد بحث و بررسی قرار گرفت. مدیر سامانه به منظور بررسی پارامترهای مقاوم‌سازی و تهیه گزارش در این زمینه می‌تواند از چک لیست زیر استفاده نماید.

	عنوان	تنظیم صحیح	
		بله	خیر
۱	**ایمن سازی محیط اجرا**		
۱-۱	نصب آخرین نسخه جاوا	☐	☐
۱-۲	نصب آخرین نسخه پایتون	☐	☐
۱-۳	اجرای سرویس Cassandra توسط کاربر غیر ریشه	☐	☐
۱-۴	پیکربندی فایل تنظیمات	☐	☐
۱-۵	پیکربندی دایرکتوری ذخیره داده	☐	☐
۱-۶	پیکربندی فایل‌های رویدادنگاری	☐	☐

۲ نصب و پیکره‌بندی امن پایگاه‌داده

در این فصل، نصب آخرین نسخه‌ی Cassandra به منظور امن‌سازی اولیه‌ی پایگاه‌داده Cassandra معرفی می‌شود.

۲-۱ نصب آخرین نسخه Cassandra

نسخه‌ی نصب شده Cassandra و وصله‌ها باید آخرین موارد منطبق با نیازمندی‌های عملیاتی سازمان باشند. بسته‌های نرم‌افزاری باید از منابع معتبر و مجاز دانلود شوند. Cassandra را می‌توان از منابع معتبر زیر دانلود کرد:

- وب سایت رسمی Apache Cassandra

```
http://cassandra.apache.org/
```

- وب سایت DataStax Enterprise

```
https://www.datastax.com/
```

تهدید/توجیه امنیتی:

نصب آخرین نسخه نرم‌افزار Cassandra به همراه بسته‌های تکمیلی، احتمال سوء استفاده از آسیب‌پذیری‌های نرم‌افزار را کاهش می‌دهد. نسخه نصب شده باید مطابق با نیازمندی‌های سازمان باشد. همچنین، این اطمینان باید حاصل شود که نسخه نصب شده دارای پشتیبانی بوده و به‌روزرسانی‌های منظمی برای رفع آسیب‌پذیری‌ها انجام می‌شود.

اطلاع از وضعیت فعلی:

با استفاده از دستور زیر نسخه نصب شده‌ی Cassandra به دست می‌آید.

```
cassandra -v
```

در صورتی که نسخه‌ی قدیمی/پشتیبانی نشده Cassandra نصب شده باشد، با استفاده از خروجی دستور فوق می‌توان متوجه آن شد.

مقاوم‌سازی:

به‌روزرسانی نرم‌افزار Cassandra موجود در هر گره[4] از خوشه[5] با گام‌های زیر انجام می‌شود:

۱. استفاده از دستور nodetool برای انتقال داده موجود در memtables به SSTables

```
nodetool drain -h <hostname>
```

۲. متوقف کردن سرویس‌های Cassandra

```
service cassandra stop
```

۳. تهیه نسخه پشتیبان از مجموعه داده و تمامی فایل‌های پیکربندی Cassandra

۴. دانلود/به‌روزرسانی جاوا در صورت نیاز

۵. دانلود/به‌روزرسانی پایتون در صورت نیاز

۶. دانلود فایل‌های دودویی آخرین نسخه Cassandra

۷. نصب نسخه جدید Cassandra

۸. پیکربندی نسخه جدید Cassandra و به کارگیری تنظیمات پیشین موجود در فایل‌های پیکربندی همچون cassandra.yml و cassandrea-env.sh

۹. راه‌اندازی سرویس‌های Cassandra

```
service cassandra start
```

۱۰. بررسی فایل‌های رویدادنگاری و رویدادهای خطا یا هشدار. به صورت پیش‌فرض مسیر فایل‌های رویدادنگاری، مسیر /var/log/cassandra/ است.

۱۱. استفاده از nodetool برای ارتقاء SSTables

```
nodetool upgradesstables
```

۱۲. استفاده از دستور nodetool برای بررسی وضعیت خوشه

```
nodetool -h <hostname> status
```

[4] Node
[5] Cluster

۲-۲ جمع‌بندی

در این فصل به تشریح به‌روزرسانی Cassandra به عنوان یکی از مهم‌ترین تنظیمات مربوط به پیکربندی سمپاد قبل از بکارگیری عملیاتی آن پرداختیم. مدیر سامانه به منظور بررسی پارامترهای مقاوم سازی و تهیه گزارش می‌تواند از چک لیست زیر استفاده نماید.

	عنوان	تنظیم صحیح	
		بله	خیر
۲	**پیکربندی امن پایگاه‌داده**		
۲-۱	نصب آخرین نسخه Cassandra	☐	☐

۳ امن‌سازی اتصال به پایگاه‌داده

در این فصل پیشنهاداتی برای مکانیزم‌های احراز اصالت کاربران و بررسی مجوزها در پایگاه‌داده Cassandra ارائه شده است.

۳-۱ فعال‌سازی احراز اصالت در پایگاه‌داده Cassandra

قابلیت احراز اصالت را می‌توان از طریق تنظیم authenticator در فایل پیکربندی cassandra.yaml به پایگاه‌داده Cassandra اضافه کرد. تنظیم authenticator می‌تواند یکی از دو مقدار AllowAllAuthenticator و PasswordAuthenticator را دریافت کند. مقدار پیش‌فرض یعنی AllowAllAuthenticator احراز اصالت را غیرفعال می‌کند. مقدار PasswordAuthenticator احراز اصالت از طریق نام کاربری و رمز عبور را فعال کرده و سبب می‌شود که رمزهای عبور در جدول سیستمی به صورت رمز شده ذخیره شوند.

تهدید/توجیه امنیتی:

احراز اصالت یک شرط لازم برای زیرساخت مجوزها در Cassandra است، بنابراین در صورتی که احراز اصالت غیرفعال باشد، زیرسیستم مجوزها نیز غیرفعال خواهد بود. در صورت عدم احراز اصالت کلاینت‌ها، کاربران و/یا سرورها، دسترسی‌های غیرمجاز به پایگاه‌داده Cassandra امکان‌پذیر خواهد شد و ردیابی فعالیت‌ها غیرممکن می‌شود. پیش از آنکه کسی به سرور Cassandra دسترسی پیدا کند، باید مکانیزم احراز اصالت پیاده سازی شود.

اطلاع از وضعیت فعلی:

با استفاده از دستور زیر می‌توان بررسی کرد که آیا احراز اصالت روی سرور Cassandra فعال است یا خیر. فایل‌های پیکربندی Cassandra را می‌توان در مسیر /etc/cassandra یافت.

```
cat cassandra.yaml | grep -in "authenticator:"
```

انتظار می‌رود خروجی دستور فوق PasswordAuthenticator باشد.

مقاوم‌سازی:

به منظور فعال‌سازی مکانیزم احراز اصالت گام‌های زیر باید طی شوند:

۱. متوقف‌سازی پایگاه‌داده Cassandra

```
service cassandra stop
```

۲. تغییر فایل cassandra.yaml و افزودن/تغییر پارامتر authenticator با مقدار PasswordAuthenticator: در صورتی که پارامتر authenticator با مقداری به غیر از مقدار

PasswordAuthenticator تنظیم شده باشد، با دستور زیر می توان مقدار این پارامتر را به مقدار PasswordAuthenticator تغییر داد:

```
sudo sed –i
"s/authenticator\s*:\s*AllowAllAuthenticator/authenticator:
PasswordAuthenticator/" "/etc/cassandra/cassandra.yaml"
```

در صورتی که پارامتر authenticator در فایل پیکربندی تنظیم نشده باشد، با دستور زیر این پارامتر با مقدار مناسب تنظیم می‌شود:

```
sudo sed –i '$ a authenticator: PasswordAuthenticator'
"/etc/cassandra/cassandra.yaml"
```

۳. اجرا و راه‌اندازی پایگاه‌داده Cassandra:

```
service cassandra start
```

۳-۲ فعال‌سازی بررسی مجوزها در پایگاه‌داده Cassandra

قابلیت بررسی مجوزها[6] را می‌توان از طریق تنظیم authorizer در فایل پیکربندی cassandra.yaml به پایگاه‌داده Cassandra اضافه کرد. تنظیم authorizer می‌تواند یکی از دو مقدار AllowAllAuthorizer و CassandraAuthorizer را دریافت کند. مقدار پیش‌فرض یعنی AllowAllAuthorizer هیچ بررسی انجام نداده و به تمامی نقش‌ها تمامی مجوزها را اعطا می‌کند. مقدار CassandraAuthorizer مدیریت کامل مجوزها را اجرا کرده و داده‌های مرتبط را در جداول سیستمی Cassandra ذخیره می‌کند. در صورتی که برای تنظیم authenticator مقدار AllowAllAuthenticator تنظیم شده باشد، مقدار تنظیم authorizer باید برابر AllowAllAuthorizer باشد.

تهدید/توجیه امنیتی:

در صورتی که مجوزهای نقش‌ها بررسی شوند می‌توان مطمئن بود که تنها دسترسی‌های مجاز به جداول پایگاه‌داده Cassandra امکان‌پذیر است. بررسی مجوزهای نقش‌ها لازمه‌ی پیاده‌سازی قانون حداقل مجوزها است. پیش از آنکه کسی به پایگاه‌داده Cassandra دسترسی پیدا کند باید مکانیزم بررسی مجوزها پیاده‌سازی شود.

---

[6] Authorization

اطلاع از وضعیت فعلی:

با استفاده از دستور زیر می‌توان بررسی کرد که آیا بررسی مجوزها روی سرور Cassandra فعال است یا خیر. فایل‌های پیکربندی Cassandra را می‌توان در مسیر /etc/cassandra یافت.

```
cat cassandra.yaml | grep -in "authorizer:"
```

انتظار می‌رود خروجی دستور فوق CassandraAuthorizer باشد.

مقاوم‌سازی:

به منظور فعال‌سازی مکانیزم بررسی مجوزها گام‌های زیر باید طی شوند:

۱. متوقف‌سازی پایگاه‌داده Cassandra

```
service cassandra stop
```

۲. تغییر فایل cassandra.yaml و افزودن/تغییر تنظیم authorizer با مقدار CassandraAuthorizer: در صورتی که پارامتر authorizer با مقداری به غیر از مقدار CassandraAuthorizer تنظیم شده باشد، با دستور زیر می توان مقدار این پارامتر را به مقدار CassandraAuthorizer تغییر داد:

```
sudo sed -i "s/authorizer\s*:\s*AllowAllAuthorizer/authorizer:
CassandraAuthorizer/" "/etc/cassandra/cassandra.yaml"
```

در صورتی که پارامتر authorizer در فایل پیکربندی تنظیم نشده باشد، با دستور زیر این پارامتر با مقدار مناسب تنظیم می‌شود:

```
sudo sed -i '$ a authorizer: CassandraAuthorizer'
"/etc/cassandra/cassandra.yaml"
```

۳. اجرا و راه‌اندازی پایگاه‌داده Cassandra

```
service cassandra start
```

۳-۳ جمع‌بندی

در این فصل به تشریح تنظیمات مربوط به امن‌سازی اتصال به سمپاد پرداختیم. در این راستا، برخی از تنظیمات مرتبط مورد بحث و بررسی قرار گرفتند. مدیر سامانه به منظور بررسی پارامترهای مقاوم‌سازی و تهیه گزارش در این زمینه می‌تواند از چک لیست زیر استفاده نماید.

	عنوان	تنظیم صحیح	
		بله	خیر
۳	**امن‌سازی اتصال به پایگاه‌داده**		
۳-۱	فعال‌سازی احراز اصالت در پایگاه‌داده Cassandra	☐	☐
۳-۲	فعال‌سازی بررسی مجوزها در پایگاه‌داده Cassandra	☐	☐

۴ کنترل دسترسی و مجازشماری

این فصل شامل پیشنهاداتی به منظور محدودسازی دسترسی به پایگاه‌داده Cassandra و خط‌مشی‌های رمز عبور است.

۴-۱ نقش‌های کاربر فوق‌العاده

روال نصب پیش‌فرض برای پایگاه‌داده Cassandra یک نقش کاربر فوق‌العاده[7] به نام cassandra ایجاد می‌کند. این موضوع نشان‌دهنده‌ی ضرورت ایجاد یک نقش جداگانه به عنوان نقش کاربر فوق‌العاده است.

تهدید/توجیه امنیتی:

مجوزهای کاربر فوق‌العاده برای ایجاد، حذف و مدیریت مجوزهای سایر کاربران به کار می‌رود. با توجه به آنکه نقش cassandra شناخته شده است، نباید یک کاربر فوق‌العاده باشد یا کاربری باشد که وظایف مدیریتی را انجام می‌دهد.

اطلاع از وضعیت فعلی:

در صورتی که خروجی دستور زیر cassandra یا هر نقش تأیید نشده‌ای باشد، پیکربندی مناسبی بر روی سرور انجام نشده است.

```
SELECT role FROM system_auth.roles WHERE is_superuser = True;
```

مقاوم‌سازی:

مراحل زیر برای اعمال تنظیمات مناسب باید طی شوند.

۱. اجرای دستورات زیر

```
create role '<NEW_ROLE_HERE>' with password='<NEW_PASSWORD_HERE>' and
login=TRUE and superuser=TRUE ;
grant all permissions on all keyspaces to <NEW_ROLE_HERE>;
```

مقدار *<NEW_ROLE_HERE>* با نقش مناسب و مقدار *<NEW_PASSWORD_HERE>* با رمز عبور جایگزین شود.

[7] Superuser

۲. بررسی شود نقش جدید به درستی عمل می‌کند.
۳. حذف نقش کاربر فوق‌العاده از حساب کاربری cassandra با اجرای دستور زیر:

```
UPDATE system_auth.roles SET is_superuser=false WHERE role='cassandra'
```

۴-۲ تغییر رمز عبور پیش‌فرض نقش Cassandra

نقش cassandra رمز عبور پیش‌فرضی (رمز عبور پیش‌فرض cassandra است) دارد که باید تغییر داده شود.

تهدید/توجیه امنیتی:

در صورت عدم تغییر رمز عبور پیش‌فرض مربوط به نقش cassandra، پایگاه‌داده می‌تواند در معرض ریسک دسترسی غیرمجاز قرار گیرد.

اطلاع از وضعیت فعلی:

در صورتی که اتصال از طریق دستور زیر موفقیت‌آمیز باشد، نشان می‌دهد که نقش cassandra، رمز عبور پیش‌فرض دارد. توجه به این نکته حائز اهمیت است که پارامتر authenticator در فایل پیکربندی cassandra.yaml حتما باید برابر PasswordAuthenticator باشد تا در اتصال از طریق دستور زیر نام کاربری و رمز عبور نیز دخیل شوند، در غیر این صورت احراز اصالت انجام نمی‌شود و اتصال صورت می‌گیرد.

```
cqlsh -u cassandra -p cassandra
```

مقاوم‌سازی:

با استفاده از دستور زیر می‌توان رمز عبور نقش Cassandra را تغییر داد.

```
alter role 'cassandra' with password='<NEWPASSWORD_HERE>';
```

مقدار <NEWPASSWORD_HERE> باید با رمز عبور مناسب جایگزین شود.

۴-۳ استفاده از حساب سرویس اختصاصی و غیرممتاز

استفاده از کاربر اختصاصی برای یک سرویس امکان محدودسازی سرویس را فراهم می‌آورد.

تهدید/توجیه امنیتی:

استفاده از حساب سرویس غیرممتاز برای Cassandra مخاطرات ناشی از آسیب‌پذیری‌های مربوط به Cassandra را کاهش می‌دهد. حساب محدود شده، توانایی دسترسی به منابع نامرتبط با Cassandra همچون پیکربندی‌های سیستم عامل را ندارد.

اطلاع از وضعیت فعلی:

به منظور بررسی حساب سرویس Cassandra از دستور زیر استفاده می‌شود.

```
ps -ef | egrep "^cassandra.*$"
```

در صورتی که دستور فوق هیچ خروجی نداشته باشد، حساب سرویس اختصاصی برای Cassandra وجود ندارد. در اینجا فرض شده است که حساب سرویس اختصاصی برای پایگاه‌داده، cassandra است.

مقاوم‌سازی:

یک کاربر برای اجرای Cassandra و فرآیندهای مرتبط با آن باید ایجاد شود. این حساب کاربری نباید مجوزهای مدیریتی در سیستم داشته باشد. بدین منظور می‌توان مراحل زیر را دنبال کرد:

۱. ایجاد گروه و نام کاربری

```
sudo groupadd Cassandra
sudo useradd -m -d /var/lib/Cassandra/ -s /bin/bash -g Cassandra Cassandra
```

۲. تنظیم مالک دایرکتوری‌های مربوط به پایگاه داده Cassandra

```
sudo chown -R cassandra:cassandra /var/lib/cassandra/
sudo chown -R cassandra:cassandra /var/run/cassandra/
```

۳. در صورتی که سرویس Cassandra تعریف شده باشد، نام کاربری موجود در فایل /etc/init.d/Cassandra باید به دستور زیر، با مقدار مناسب تنظیم شود:

```
sudo sed -i "s/start-stop-daemon -S -c [[:alnum:]]*/start-stop-daemon -S -c cassandra/" /etc/init.d/cassandra
```

۴-۴ جمع‌بندی

در این فصل موارد مربوط به کنترل دسترسی و مجازشماری مورد بحث و بررسی قرار گرفت. مدیر سامانه به منظور بررسی پارامترهای مقاوم‌سازی و تهیه گزارش در این زمینه می‌تواند از چک لیست زیر استفاده نماید.

	عنوان	تنظیم صحیح	
		بله	خیر
۴	**کنترل دسترسی و مجازشماری**		
۴-۱	نقش‌های کاربر فوق‌العاده	☐	☐

۴-۲	تغییر رمز عبور پیش‌فرض نقش cassandra	☐	☐
۴-۳	استفاده از حساب سرویس اختصاصی و غیرممتاز	☐	☐

۵ تنظیمات و خط‌مشی‌های رویدادنگاری/ممیزی

در این فصل به تشریح مکانیزم‌های رویدادنگاری و ممیزی می‌پردازیم.

۵-۱ فعال‌سازی رویدادنگاری

Apache Cassandra از Logback برای رویدادنگاری استفاده می‌کند. می‌توان سطح رویدادنگاری را با دستور nodetool setlogginglevel تغییر داد ولی هنگامی که Cassandra مجدداً راه‌اندازی شود، سطح رویدادنگاری تنظیم شده در فایل logback.xml دو مرتبه اعمال می‌شود. سطوح رویدادنگاری قابل اعمال شامل موارد زیر هستند:

- OFF
- TRACE
- DEBUG
- INFO (مقدار پیش‌فرض)
- WARN
- ERROR

تهدید/توجیه امنیتی:

در صورتی که رویدادنگاری فعال نباشد، موارد امنیتی و حوادث رخ داده شده ممکن است تشخیص داده نشوند.

اطلاع از وضعیت فعلی:

به منظور بررسی تنظیمات مربوط به رویدادنگاری دستور زیر اجرا شود:

```
nodetool getloggginglevels
```

انتظار می‌رود خروجی دستور فوق OFF نباشد.

مقاوم‌سازی:

برای تنظیم سطح رویدادنگاری مراحل زیر باید طی شوند:

۱. در صورت وجود فایل logback-test.xml، این فایل ویرایش شود، در غیر این صورت فایل logback.xml ویرایش شود. به عنوان نمونه در صورتی که رویدادنگاری غیرفعال باشد، با دستور زیر می‌توان سطح رویدادنگاری را به INFO تغییر داد:

```
sudo sed -i 's/<root level="OFF">/<root level="INFO">/'
"/etc/cassandra/logback.xml"
```

قسمتی از محتوای فایل logback.xml در ادامه آورده شده است:

```
<configuration scan="true">
<appender name="STDOUT"
class="ch.qos.logback.core.ConsoleAppender">
<filter class="ch.qos.logback.classic.filter.ThresholdFilter">
<level>INFO</level>
</filter>
<encoder>
<pattern>%-5level [%thread] %date{ISO8601} %F:%L -
%msg%n</pattern>
</encoder>
</appender>
<root level="INFO">
<appender-ref ref="STDOUT" />
</root>
<logger name="org.cisecurity.workbench" level="WARN"/>
</configuration>
```

۲. راه‌اندازی مجدد Apache Cassandra

```
service cassandra restart
```

۵-۲ جمع‌بندی

در این فصل به تشریح پیکربندی مناسب رویدادنگاری پرداختیم. مدیر سامانه به منظور بررسی پارامترهای مقاوم‌سازی و تهیه گزارش در این زمینه می‌تواند از چک لیست زیر استفاده نماید.

عنوان		تنظیم صحیح	
		بله	خیر
۵	**تنظیمات رویدادنگاری**		
۵-۱	فعال‌سازی رویدادنگاری	☐	☐

۶ تنظیمات رمزنگاری

این فصل شامل پیشنهاداتی مرتبط با جنبه‌های رمزنگاری Cassandra است.

۶ – ۱ رمزنگاری بین گره‌ها

Cassandra امکان رمزنگاری داده‌های در حال انتقال میان گره‌های درون خوشه را فراهم می‌کند. به صورت پیش‌فرض، رمزنگاری میان گره‌ها غیرفعال است.

تهدید/توجیه امنیتی:

می‌توان از سرقت داده‌های در حال انتقال میان گره‌ها از طریق رمزنگاری جلوگیری کرد.

اطلاع از وضعیت فعلی:

برای بررسی وضعیت رمزنگاری بین گره‌ها، دستور زیر اجرا شود:

```
cat cassandra.yaml | grep -in "internode_encryption:"
```

در صورتی که خروجی دستور فوق none باشد یعنی رمزنگاری میان گره‌ها فعال نیست. مقادیر قابل قبول برای تنظیم internode_encryption شامل all، dc یا rack است.

مقاوم‌سازی:

پیش از آنکه کسی به سرور Cassandra دسترسی داشته باشد، رمزنگاری میان گره‌ها باید پیاده‌سازی شود. به منظور فعال‌سازی رمزنگاری میان گره‌ها مراحل زیر باید طی شوند:

۱. متوقف کردن پایگاه‌داده Cassandra

```
service cassandra stop
```

۲. keystore و truststore را ساخته و تنظیمات مربوط به آن‌ها را در فایل پیکربندی وارد نمایید. در صورتی که پیش از این تنظیمات مربوط به keystore و truststore در فایل پیکربندی cassandra.yaml وارد نشده باشد، با دستورات زیر می‌توان آن‌ها را تنظیم کرد:

```
sudo sed -i -e "/^[[:blank:]]*server_encryption_options:.*/a \
keystore: <keystore_path>" "/etc/cassandra/cassandra.yaml"


sudo sed -i -e "/^[[:blank:]]*server_encryption_options:.*/a \
keystore_password: <keystore_password>"
"/etc/cassandra/cassandra.yaml"
```

```
sudo sed -i -e "/^[[:blank:]]*server_encryption_options:.*/a \
truststore: <truststore_path>" "/etc/cassandra/cassandra.yaml"

sudo sed -i -e "/^[[:blank:]]*server_encryption_options:.*/a \
truststore_password: <truststore_password>"
"/etc/cassandra/cassandra.yaml"
```

همچنین در صورتی که پیش از این مقادیر مربوط به پارامترهای keystore، keystore_password، truststore و truststore_password تنظیم شده باشند، با باز کردن فایل cassandra.yaml می‌توان مقادیر پارامترها را با مقادیر دلخواه جایگزین کرد.

۳. تغییر فایل پیکربندی cassandra.yaml و افزودن/تغییر تنظیم internode_encyption با مقدار all:

در صورتی که پارامتر internode_encyption با مقداری به غیر از مقدار all تنظیم شده باشد، با دستور زیر می توان مقدار این پارامتر را به مقدار all تغییر داد:

```
sudo sed -i "s/internode_encryption\s*:\s*none/internode_encryption:
all/" "/etc/cassandra/cassandra.yaml"
```

در صورتی که پارامتر internode_encyption در فایل پیکربندی تنظیم نشده باشد، با دستور زیر این پارامتر با مقدار مناسب تنظیم می‌شود:

```
sudo sed -i -e "/^server_encryption_options:.*/a \
internode_encryption: all" "/etc/cassandra/cassandra.yaml"
```

۴. راه‌اندازی پایگاه‌داده Cassandra

```
service cassandra start
```

۲ – ۶ رمزنگاری کلاینت

Cassandra امکان رمزنگاری داده‌های در حال انتقال میان گره‌های درون خوشـــه و کلاینت را فراهم می‌کند. به صورت پیش‌فرض، رمزنگاری کلاینت غیرفعال است.

تهدید/توجیه امنیتی:

می‌توان از سرقت داده‌های در حال انتقال میان کلاینت و گره موجود در خو شه از طریق رمزنگاری جلوگیری کرد.

اطلاع از وضعیت فعلی:

وارد فایل Cassandra.yaml شده و به دنبال قسمت client_encryption_options بگردید. مقادیر دو تنظیم enabled: و optional: را بررسی کنید.

```
enabled: true
optional: false
```

در صورتی که مقدار هیچ یک از دو تنظیم فوق true نبا شد، تمامی ارتباطات کلاینت رمز نشده خواهند بود. در صورتی که enabled برابر true و optional برابر false باشد، تمامی ارتباطات کلاینت باید رمز شده باشند. در صورتی که enabled برابر false و optional برابر true باشد، enabled اعمال شده و تمامی ارتباطات کلاینت رمز نشده خواهند بود. در صورتی که هر دو تنظیم برابر true باشند، ارتباطات رمز شده و رمز نشده، هر دو مجاز خواهند بود. لازم به ذکر است که به صورت پیش‌فرض مقدار هر دو تنظیم برابر false است.

مقاوم‌سازی:

پیش از آنکه کسی به سرور Cassandra دسترسی داشته باشد، رمزنگاری کلاینت باید پیاده سازی شود. به منظور فعال‌سازی رمزنگاری کلاینت مراحل زیر باید طی شوند:

۱. متوقف کردن پایگاه‌داده Cassandra

```
service cassandra stop
```

۲. keystore را ساخته و تنظیمات مربوط به آن را در فایل پیکربندی وارد نمایید. در صورتی که پیش از این تنظیمات مربوط به keystore در فایل پیکربندی cassandra.yaml وارد نشده باشد، با دستورات زیر می‌توان آن‌ها را تنظیم کرد:

```
sudo sed -i -e "/^[[:blank:]]*server_encryption_options:.*/a \ keystore: <keystore_path>" "/etc/cassandra/cassandra.yaml"
sudo sed -i -e "/^[[:blank:]]*server_encryption_options:.*/a \ keystore_password: <keystore_password>" "/etc/cassandra/cassandra.yaml"
```

همچنین در صورتی که پیش از این مقادیر مربوط به پارامترهای keystore و keystore_password تنظیم شده باشند، با باز کردن فایل cassandra.yaml می‌توان مقادیر پارامترها را با مقادیر دلخواه جایگزین کرد.

۳. تغییر فایل پیکربندی cassandra.yaml و افزودن/تغییر موارد زیر در قسمت client_encryption_options:

```
set enabled: true
set optional: false
```

با تنظیم دو مقدار فوق، تمامی ارتباطات میان کلاینت و گره روی خوشه باید رمزگذاری شوند.

۴. راه‌اندازی پایگاه‌داده Cassandra

```
service cassandra start
```

۶-۳ جمع‌بندی

در این فصـــل به تشـــریح برخی از مهم‌ترین پارامترهای امنیتی مربوط به پیکربندی تنظیمات رمزنگاری پرداختیم. مدیر سامانه به منظور بررسی پارامترهای مقاوم‌سازی و تهیه گزارش در این زمینه می‌تواند از چک لیست زیر استفاده نماید.

	عنوان	تنظیم صحیح	
		بله	خیر
۶	**تنظیمات رویدادنگاری**		
۶-۱	رمزنگاری بین گره‌ها	☐	☐
۶-۲	رمزنگاری کلاینت	☐	☐

۷ راهنمای اعمال مقاوم‌سازی

این پروژه دارای سه فایل اجرایی است که در ادامه به بررسی هر یک می‌پردازیم.

۷-۱ فایل start.sh

این فایل، تنها فایلی است که کاربر باید آن را اجرا کند. برای آنکه فرآیند تست پایگاه‌داده و امن‌سازی آن صورت گیرد، ابتدا لازم است از وجود Cassandra روی سیستم اطمینان حاصل گردد. پس از آن، کاربر می‌تواند نام کاربری و رمز عبور مورد نظر خود را برای اتصال به پایگاه‌داده وارد کند. همچنین، آدرس میزبان نیز از کاربر دریافت می شود که مقدار پیش‌فرض آن locahost ا ست. توجه به این نکته حائز اهمیت ا ست که در ا سکریپت‌ها فرض بر آن ا ست که اجرای سرور Cassandra از طریق سرویس Cassandra انجام می شود. پس از اجرای start.sh و دریافت اطلاعات مورد نظر از کاربر، حال اسکریپت script.sh اجرا می‌شود. در اثر اجرای این فایل، دو پوشه حاوی نتایج آزمایش و نتایج مورد انتظار ایجاد می‌شوند. در این اسکریپت قصد داریم این دو پوشه را با هم مقایسه کنیم تا مغایرت‌های سیستم با موارد امنیتی مورد انتظار مشخص شوند. نتایج این آزمایش در فایل first_test_result ثبت می شود. نمونه‌ای از خروجی این برنامه در شکل (۱) نشان داده شده است.

```
      "Cassandra RESULT"                                    "EXPECTED RESULT"

<------1-1-version of Java ------>                          <------1-1-version of Java ------>
openjdk version "1.8.0_144"                               | The latest version of Java 8, either the Oracle Java Standard
OpenJDK Runtime Environment (build 1.8.0_144-8u144-b01-2-b01) <
OpenJDK 64-Bit Server VM (build 25.144-b01, mixed mode)       <


<------1-2-version of Python ------>                        <------1-2-version of Python ------>
Python 2.7.14                                             | For using cqlsh, the latest version of Python 2.7.


<------1-3-Cassandra service is run as a non-root user -|-----  <------1-3-Cassandra service is run as a non-root user ------
cassandra                                                 | non-root user


<------2-1-Version of Cassandra ------>                     <------2-1-Version of Cassandra ------>
3.11.4                                                     3.11.4


<------3-1-Authentication for Cassandra databases ------>     <------3-1-Authentication for Cassandra databases ------>
AllowAllAuthenticator                                     | PasswordAuthenticator


<------3-2-Authorization for Cassandra databases ------>     <------3-2-Authorization for Cassandra databases ------>
AllowAllAuthorizer                                        | CassandraAuthorizer
```

شکل ۱: محتوای فایل first_test_result

ستون سمت چپ نشان دهنده تنظیمات فعلی و ستون سمت راست نشان دهنده تنظیمات مورد انتظار است. پس از اجرای این اسکریپت، در صورت تمایل کاربر، اسکریپت repair اجرا می‌شود. سپس هر مورد امنیتی که

در آن نتیجه مورد انتظار و نتیجه حاصل از تست مغایر باشند، با موافقت کاربر امن سازی شده و در آخر نیز تست دوباره‌ای روی سیستم انجام می‌شود. نتیجه تست دوم در فایل second_test_result ذخیره خواهد شد.

```
     "Cassandra RESULT"                                    "EXPECTED RESULT"

<------1-1-version of Java ------>                         <------1-1-version of Java ------>
openjdk version "1.8.0_144"                              | The latest version of Java 8, either the Oracle Java Standard
OpenJDK Runtime Environment (build 1.8.0_144-8u144-b01-2-b01) <
OpenJDK 64-Bit Server VM (build 25.144-b01, mixed mode)       <


<------1-2-version of Python ------>                       <------1-2-version of Python ------>
Python 2.7.14                                            | For using cqlsh, the latest version of Python 2.7.


<------1-3-Cassandra service is run as a non-root user ------  <------1-3-Cassandra service is run as a non-root user ------
cassandra                                                | non-root user


<------2-1-Version of Cassandra ------>                    <------2-1-Version of Cassandra ------>
3.11.4                                                    3.11.4


<------3-1-Authentication for Cassandra databases ------>      <------3-1-Authentication for Cassandra databases ------>
PasswordAuthenticator                                        PasswordAuthenticator


<------3-2-Authorization for Cassandra databases ------>       <------3-2-Authorization for Cassandra databases ------>
CassandraAuthorizer                                          CassandraAuthorizer
```

شکل ۲: محتوای فایل second_test_result

۷-۲ فایل script.sh

در این فایل دو متغیر با نام‌های result_path و expected_path تعریف شده است. این دو متغیر به پوشه‌هایی اشاره می‌کنند که در آن‌ها به ترتیب نتایج هر آزمایش و نتیجه مورد انتظار آن آزمایش، ساخته می‌شود. در اینجا لازم است بنا به نیازمندی سیستم و نکات گفته شده حین توضیح هر مورد، نتایج مورد انتظار برای هر مورد امنیتی تنظیم شود. این کد توسط برنامه start.sh اجرا می‌شود.

۷-۳ فایل repair.sh

فایل آخر مربوط به تغییر تنظیمات سیستم می شود. در صورتی که تنظیمات به درستی انجام شده باشد انتظار می‌رود که مورد امنیتی در ستون دوم وجود نداشته باشد و تنها چند پیشنهاد برای امنیت بیشتر در آن باقی بماند. در شکل (۲) می‌توان نمونه‌ای از فایل second_test-result را پس از اعمال تغییرات مشاهده کرد.

۸ جمع‌بندی

در این مستند به بررسی موارد امنیتی مربوط به مقاوم‌سازی پایگاه‌داده‌ی Cassandra پرداخته شد. تنظیمات مربوط به مقاوم سازی Cassandra در شش فصل مختلف دسته‌بندی شدند. در فصل اول، امن سازی محیط اجرا، فصل دوم نصب و پیکربندی امن پایگاه‌داده، فصل سوم امن سازی اتصال به پایگاه‌داده، فصل چهارم تنظیمات کنترل دسترسی و مجازشماری، فصل پنجم تنظیمات رویدادنگاری و فصل ششم تنظیمات رمزنگاری بررسی شدند. در مورد هر پارامتر یا تنظیم امنیتی، کاربرد، ارزش امنیتی و نحوه آگاهی از مقدار کنونی آن پارامتر و چگونگی مقداردهی امن آن توضیحاتی داده شد. در پایان نیز نحوه اجرای اسکریپت‌ها و خروجی‌های سیستم بیان شدند. خلاصه‌ای از گزارش ارائه شده، به صورت یک چک لیست در ادامه آورده شده است.

	عنوان	تنظیم صحیح	
		بله	خیر
۱	**ایمن‌سازی محیط اجرا**		
۱-۱	نصب آخرین نسخه جاوا	☐	☐
۱-۲	نصب آخرین نسخه پایتون	☐	☐
۱-۳	اجرای سرویس Cassandra توسط کاربر غیر ریشه	☐	☐
۱-۴	پیکربندی فایل تنظیمات	☐	☐
۱-۵	پیکربندی دایرکتوری ذخیره داده	☐	☐
۱-۶	پیکربندی فایل‌های رویدادنگاری	☐	☐
۲	**پیکربندی امن پایگاه‌داده**		
۲-۱	نصب آخرین نسخه Cassandra	☐	☐
۳	**امن‌سازی اتصال به پایگاه‌داده**		
۳-۱	فعال‌سازی احراز اصالت در پایگاه‌داده Cassandra	☐	☐
۳-۲	فعال‌سازی بررسی مجوزها در پایگاه‌داده Cassandra	☐	☐
۴	**کنترل دسترسی و مجازشماری**		
۴-۱	نقش‌های کاربر فوق‌العاده	☐	☐

۴-۲	تغییر رمز عبور پیش‌فرض نقش cassandra	☐	☐
۴-۳	استفاده از حساب سرویس اختصاصی و غیرممتاز	☐	☐
۵	**تنظیمات رویدادنگاری**		
۵-۱	فعال‌سازی رویدادنگاری	☐	☐
۶	**تنظیمات رمزنگاری**		
۶-۱	رمزنگاری بین گره‌ها	☐	☐
۶-۲	رمزنگاری کلاینت	☐	☐

۹ مراجع

[1] https://www.cisecurity.org/