بسمه تعالی

مرکز مدیریت راهبردی افتا

عنوان

امن‌سازی پایگاه داده

گروه زیر ساخت امن

تیرماه ۱۳۹۳

فهرست مطالب

مقدمه

داده‌های موجود در یک سازمان، سرمایه اصلی آن سازمان به حساب می‌آیند، بنابراین کلیه منابع انسانی، تجهیزات و منابع مادی سعی در نگه‌داری، حفظ، انتقال امن، سطح دسترسی مطلوب و در نهایت کاربری مفید آن‌ها را دارند.

به طور کلی سطح وسیعی از اطلاعات شامل داده‌های آموزشی، مدیریتی، پژوهشی، مالی، اداری و پرسنلی در شبکه‌های مختلف کشور، در مجموعه کلی داده‌ها جای می‌گیرند. از طرفی رشد روزافزون داده‌ها باعث می‌شود حجم و نوع داده‌های اندوخته شده و حساسیت آن‌ها تغییر نمایند. موارد فوق سبب می‌شوند تا صاحبان و مدیران شبکه در خصوص تهاجم و استفاده غیرمجاز بعضی کاربران (خرابکاران) چاره‌اندیشی نمایند.

حملاتی که بر علیه داده‌ها صورت می‌پذیرد را می‌توان در دسته های کلی زیر بیان کرد:

- تهاجماتی که باعث افشا شدن اطلاعات حساس می‌شوند.
- تهاجماتی که باعث تغییرات ناخواسته در اطلاعات ذخیره‌شده می‌گردند.
- تهاجماتی که باعث تغییرات عمدی در اطلاعات می‌شوند.
- تهاجماتی که باعث جلوگیری از دسترسی افراد مجاز به اطلاعات می‌شوند.
- تهاجماتی که باعث ایجاد دسترسی افراد غیرمجاز به اطلاعات می‌گردند.

پایگاه داده در هر سازمان شامل داده‌های بسیار مهمی از قبیل سوابق مشتری و دیگر اطلاعات محرمانه تجارت سازمان است و بنابراین از این جهت بسیار مورد توجه نفوذگرها می‌باشد. اما چرا پایگاه داده‌ها آسیب پذیر هستند؟ یک دلیل این است که سازمان‌ها از دارایی‌هایشان به اندازه کافی محافظت نمی‌کنند. در سال ۲۰۱۱ کمتر از ۵ درصد از ۲۷ میلیارد دلاری که صرف محصولات امنیتی شده است، به طور مستقیم به امنیت پایگاه‌های داده اختصاص یافته است. زمانی که هکرها و یا افراد درون سازمانی مخرب به اطلاعات حساس دست پیدا می‌کنند، می‌توانند به سرعت اطلاعات با ارزش زیادی بدست آورند. همچنین می‌توانند خسارت زیادی به بار آورده و شرایط تجارت مورد نظر را تحت تاثیر قرار دهند.

با این حال، با توجه به گزارش (OTA) در سال ۲۰۱۳ خبر خوب این است که می‌توان از رخ دادن بسیاری از حوادث (بیش از ۹۷٪) با اجرای مراحل ساده و پیروی از کنترل‌های داخلی جلوگیری کرد. البته باید به این نکته توجه داشت که با پیشرفت علوم رایانه‌ای، همچنین به وجود آمدن ابزارهای جدید Hack و Crack و همچنین وجود صدها مشکل

ناخواسته در طراحی نرم‌افزارهای مختلف و روال‌های امنیتی سازمان‌ها، همیشه خطر حمله و دسترسی افراد غیرمجاز وجود دارد. حتی قوی‌ترین ابزارآلات مدیریتی بانک‌های اطلاعاتی موجود در دنیا در معرض خطر افراد غیرمجاز و سودجو قرار دارند.

قبل از بیان روش‌ها و موارد امن‌سازی پایگاه داده با توجه به موارد ذکر شده در پاسخ این سؤال که اصولاً چرا باید به امنیت پایگاه داده اهمیت دهیم؟ به صورت خلاصه در ادامه می‌پردازیم:

۱- برخلاف گذشته که دسترسی به پایگاه داده بسیار کم بود اکنون دسترسی به پایگاه داده به طور مستقیم به صورت خیلی روان‌تر- و البته ناامن‌تر- تحت محیط برنامه‌های کاربردی وب[۱] مهیا شده است.

۲- پایگاه‌های داده میزبان اطلاعات حساسی هستند که این مسأله باعث می‌شود اهداف ارزشمندی باشند.

۳- به خاطر اینکه پایگاه‌های داده ویژگی‌های زیادی دارند که هرکدام تنظیمات خاص خود را دارند خطر تنظیم نادرست و سوءاستفاده در آن‌ها وجود دارد.

۴- قوانین و استانداردهای امنیتی بیشتری نسبت به قبل درباره آن‌ها وضع شده است.

۵- به دلیل پیشرفت تجارت و کسب و کار الکترونیک

۶- وجود راه‌های متنوع و جدید برای استفاده از پایگاه داده

۷- افزایش آگاهی در میان جامعه هکرها

وقتی که شما محیط پایگاه داده‌تان را امن‌سازی می‌کنید شما آسیب‌پذیری‌هایی را که حاصل از سهل‌انگاری در تنظیمات می‌باشند از بین می‌برید و حتی می‌توانید آسیب‌پذیری هایی را که حاصل از باگ[۲]های سازنده پایگاه داده است جبران کنید، گرچه شما نمی‌توانید آن باگ‌ها را اصلاح کنید اما می‌توانید محیط را طوری شکل دهید که این باگ‌ها نتوانند مورد استفاده قرار گیرند. امن‌سازی پایگاه داده در حوزه‌های مختلف مورد بررسی قرار می‌گیرد که هرکدام پایگاه داده را از جنبه‌ای متفاوت مورد مطالعه قرار می‌دهند. در ادامه ابتدا تعاریفی از تهدیدات عمده در این حوزه

---

[۱] Web Application

[۲] Bug

خواهیم داشت و سپس قراردادهایی برای بکار گیری ترجمه‌های فارسی و اختصارات و اصطلاحات خواهیم داشت و بعد از آن در بخش‌های آینده با نگاه به ابعاد مختلف امنیت پایگاه داده هر کدام را به ترتیب توضیح خواهیم داد و توصیه‌های در هر بخش خواهیم داشت. مواردی که گفته خواهد شد به صورت عمومی برای پایگاه داده‌های مختلف می‌باشند بنابراین در انتها نیز به طور خاص چک لیست‌هایی برای پایگاه داده‌های خاص اعم از MySQL و SQL Server ارائه شده است.

قبل از مطالعه و اجرای این سند، نکات زیر باید مورد توجه قرار گیرد:

- از آن‌جا که ممکن است تنظیمات امن‌سازی، کارکردهای سیستم را مختل یا غیرفعال کند، لازم است قبل از اجرای تنظیمات، یک نسخه پشتیبان از پیکربندی سیستم تهیه شود.
- برای اجرای الزامات تعیین شده در این سند، اولویت با خط‌مشی‌های سازمان است. به عبارت دیگر اگر برخی از الزامات بیان شده در این سند، با خط‌مشی‌های سازمان تداخل یا تضاد داشت، اولویت با خط‌مشی‌های سازمان است.
- در تهیه این سند، سعی شده است که حداکثر الزامات مرتبط با حوزه‌ی سند، پوشش داده شود. اما این بدان معنا نیست که پس از اجرای این الزامات، سیستم به صورت صد در صد امن خواهد بود. الزامات بیان شده در این سند، حداقل انتظارات برای امن‌سازی در حوزه‌ی تعریف شده در این مستند است.

تهدیدات

در این بخش به بیان ۱۰ تهدید مهم در حوزه امنیت پایگاه داده می‌پردازیم. بدیهی است که امکان وجود تهدیدات دیگر نیز در این حوزه وجود دارد و هم ممکن است تهدیدات جدیدی در آینده به آن‌ها اضافه شوند. اما ما در اینجا به تهدیدات مهم به صورت خلاصه اشاره می‌کنیم:

۱- دسترسی بلااستفاده و بیش از اندازه: وقتی که به یک فرد دسترسی فراتر از نیازهای کاری او در پایگاه داده اعطا

می‌گردد این دسترسی می‌تواند باعث سوء استفاده گردد.

۲- سوء استفاده از حق دسترسی: کاربران از حق دسترسی‌های قانونی برای اهداف غیر مجاز استفاده می‌کنند.

۳- SQL Injection : یک حمله موفق SQL Injection به فرد دسترسی نامحدود به پایگاه داده را می‌دهد. SQL Injection مستلزم این است که دستورات پایگاه داده‌ای مخرب یا غیر مجازی را داخل قسمتی که آسیب‌پذیری وجود دارد وارد کنند (تزریق گردد) .

۴- بدافزار

۵- حسابرسی[۳] ضعیف: عدم توانایی در جمع‌آوری جزئیات اطلاعات حسابرسی پایگاه داده، بررسی و پایش امنیت آن را با خطر مواجه می‌کند.

۶- در معرض قرار گرفتن رسانه‌های ذخیره‌سازی: به عنوان نمونه رسانه‌های ذخیره‌سازی نسخه پشتیبان اغلب کاملاً در برابر حملات بی حفاظ می‌باشند.

۷- استفاده از پایگاه داده‌های آسیب‌پذیر و نادرست تنظیم شده

۸- عدم مدیریت داده‌های حساس: بسیاری از شرکت‌ها داده‌های حساس را به همان صورت داده‌های معمولی نگه‌داری می‌کنند.

۹- DOS[۴]

۱۰- محدودیت در وجود خبرگان امنیتی و آموزش

دامنه

این مستند به بررسی موارد امنیتی بدون پیش‌فرض استفاده از پایگاه داده‌ای خاص و ارائه توصیه در این حوزه می‌پردازد و در انتها نیز چک لیست‌هایی برای دو محصول MySQL و SQL Server ارائه گردیده است.

---

[۳] Auditing

[۴] Denial Of Service

اصطلاحات و اختصارات

۱. حسابرسی: پایش کردن و ضبط اعمال کاربران پایگاه داده.

۲. پایگاه داده به اختصار DB نوشته می‌شود.

۳. روش‌های پذیرفته شده (best practices) : در اصطلاح به مجموعه‌ای از روش‌ها و توصیه‌ها اطلاق می‌شود که تجربه نشان داده است در صورت رعایت آن‌ها به بهترین نتایج دست می یابیم. این روش‌ها و توصیه‌ها مورد پذیرش اهل فن قرار گرفته است.

۴. محیط تولید : محیطی که داده و DB در آن برای انجام کار واقعی حاضر هستند.

۵. False-positive : موردی که به غلط حمله، حرکت مشکوک، هشدار و غیره شناخته شده است.

۶. DDL : زبانی برای تعریف داده ساختارها و بخصوص شماهای DB می‌باشد.

۷. مخازن داده : کامپیوتر یا هر رسانه‌ای دیگر که توانایی دریافت داده را دارد.

۸. منابع داده : مبدا تولید داده

۹. رویه‌های خارجی : توابعی هستند که در زبانی دیگر مانند C نوشته شده‌اند و در پایگاه داده قابلیت فراخوانی دارند.

۱۰. صاحب داده : کسی که اجازه تولید، تغییر، فشرده‌سازی، بهره‌برداری، فروش و حتی حذف داده را دارد و مرجعی برای دادن دسترسی به داده برای دیگران نیز می‌باشد.

در ادامه به امنیت DB از جنبه‌های مختلف می‌پردازیم:

موارد امنیتی با نگاه به DB به عنوان یک سرور تحت شبکه

یک DB در درجه نخست یک مهیا کننده سرویس است و منتظر درخواست‌ها از طرف کاربر می‌ماند تا پس از دریافت درخواست‌ها، آن‌ها را پاسخ دهد. در اغلب موارد DB به صورت یک سرور تحت شبکه استفاده می‌گردد و به عنوان یک سرویس تحت شبکه شناخته می‌شود. بنابراین می‌توان به DB از این جهت مانند یک گره[۵] روی شبکه نگاه کرد. مانند همه گره‌های شبکه DB نیز ممکن است در برابر حملات تحت شبکه آسیب‌پذیر باشد. در این بخش به امن‌سازی DB به عنوان یک سرور می‌پردازیم.

تکنیک‌های اصلی حول چند مفهوم ساده می‌چرخند. با دید سطح بالا باید کارهای زیر انجام شوند:

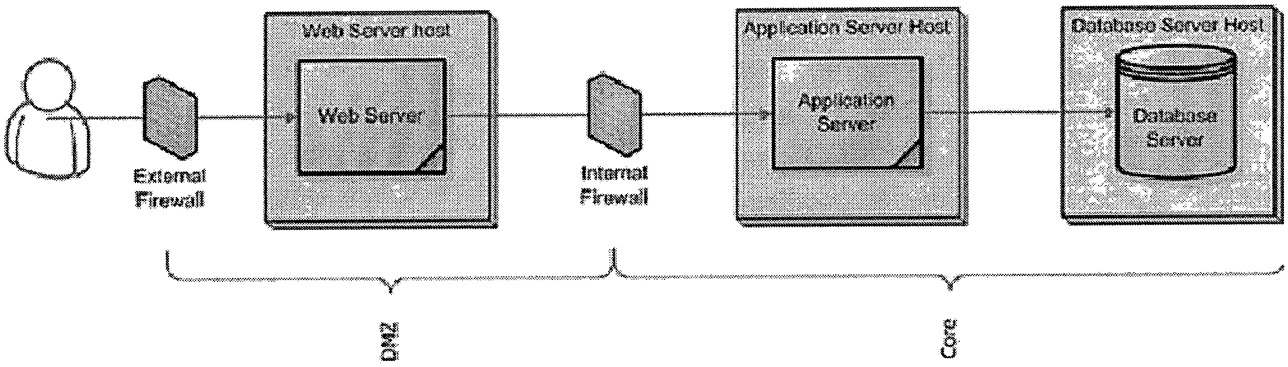
- فهمیدن و کنترل طریقه دسترسی به DB
- آنچه احتیاج ندارید را حذف کنید
- آنچه نیاز دارید را امن کنید
- به صورت پیوسته هرگونه تغییر در طریقه دسترسی به DB را تحت شبکه پایش کنید.

توصیه‌ها:

۱- DB را در هسته معماری شبکه‌تان قرار دهید (دسترسی از بیرون غیرممکن یا محدود باشد).

---

[۵] Node



۲- DB را هرگز در معرض شبکه‌های عمومی مانند اینترنت قرار ندهید.

۳- نقشه دسترسی به محیط پایگاه داده‌تان را داشته باشید. یعنی دید کاملی روی گره‌های شبکه که به DB دسترسی دارند، داشته باشید. برای این منظور یک نقشه تهیه کرده و در آن نوع دسترسی و سطح آن را نیز برای هر گره مشخص کنید. در مواردی که دسترسی‌ها زیاد است می‌توانید به جای دیاگرام از جدول برای نشان دادن دسترسی‌ها استفاده کنید. مانند شکل زیر:

Networked Access

Client IP	Server IP	Server Type	DB User Name	Timestamp
200.100.2.1	192.168.2.33	SYBASE	jason	2004-09-29 10:49:38
200.200.2.1	192.168.2.34	SYBASE	jason	2004-09-29 10:49:38
200.100.1.1	192.168.2.33	SYBASE	jason	2004-09-29 10:49:38
200.200.1.1	192.168.2.34	SYBASE	dba	2004-09-29 10:52:27
200.200.1.2	192.168.2.33	SYBASE	barbara	2004-09-29 10:59:32
200.100.1.2	192.168.2.34	SYBASE	barbara	2004-09-29 10:59:32
200.100.2.1	192.168.2.123	MSSQL	sa	2004-10-01 09:42:52
200.200.1.1	192.168.2.123	MSSQL	sa	2004-10-01 09:42:52
200.200.1.2	192.168.2.123	MSSQL	barbara	2004-10-01 09:42:52
200.200.2.1	192.168.2.123	MSSQL	mike	2004-10-01 09:42:52
200.100.1.1	192.168.2.123	MSSQL	sa	2004-10-01 09:42:52
200.100.1.2	192.168.2.123	MSSQL	dba	2004-10-01 09:42:52
200.100.2.1	192.168.2.136	SYBASE	jason	2004-10-01 09:42:52
200.200.1.1	192.168.2.203	DB2	system	2004-10-01 09:42:52
200.200.1.2	192.168.2.136	SYBASE	barbara	2004-10-01 09:42:52
200.200.2.1	192.168.2.203	DB2	system	2004-10-01 09:42:52
200.100.1.1	192.168.2.136	SYBASE	jason	2004-10-01 09:42:52
200.100.1.2	192.168.2.203	DB2	system	2004-10-01 09:42:52
200.100.2.1	192.168.2.65	DB2	system	2004-10-01 09:42:52
200.200.1.1	192.168.2.64	ORACLE	scott	2004-10-01 09:42:52
200.200.1.2	192.168.2.65	DB2	system	2004-10-01 09:42:52
200.200.2.1	192.168.2.64	ORACLE	scott	2004-10-01 09:42:52
200.100.1.1	192.168.2.65	DB2	system	2004-10-01 09:42:52
200.100.1.2	192.168.2.64	ORACLE	dba	2004-10-01 09:42:52

۴- دسترسی به DB را با ابزارهای مناسب دنبال کنید. به دلایل زیر:

- دانستن ابزارهای متصل شده به DB و نسخه‌های آن‌ها به شما این امکان را می‌دهد که نقاط آسیب‌پذیر را بدست آورید.
- دانستن ابزارهای متصل شده به DB و نسخه‌های آن‌ها به شما این امکان را می‌دهد که آن‌ها را با قوانین حاکمیتی IT مطابقت دهید.
- مقایسه مجموعه ابزارها با موقعیت‌شان در شبکه می‌تواند باعث هشدارهایی برای تغییرات مشکوک باشد.
- دسته بندی دسترسی‌ها این امکان را به شما می‌دهد تا مطمئن شوید که سازمان و فرآیندهای کاربردی آن به قوانین پایبند هستند.

۵- کتابخانه‌های غیر لازم شبکه‌ای را حذف کنید.

۶- از پویشگرهای پورت جهت شناسایی سرویس‌ها و پورت‌های اضافی استفاده کنید و سپس آن‌ها را خاموش کنید. برای اینکار می‌توانید از ابزاری مانند netstat در ویندوز یا nmap استفاده کنید.

۷- سرویس‌ها را در مقابل حملات تحت شبکه شناخته شده امن‌سازی کنید. با اعمالی از قبیل patch کردن

۸- از فایروال[۶] استفاده کنید. فایروال‌های SQL (فایروال‌های DB) علاوه بر کار فایروال‌های معمولی که تنها روی فیلتر کردن براساس سرآیند بسته‌های TCP/IP می‌توانند کمک کنند می‌توانند روی دستورات SQL کاربران DB، انواع برنامه‌های کاربردی و اشیاء[۷] DB نیز کار انجام دهند.

۹- سرور DB باید پشت فایروالی باشد که تنها به موارد مجاز تنظیم شده اجازه دسترسی دهد.

۱۰- فایروال سرور DB تنها برای برنامه کاربردی یا سرور خاصی باز باشد و قوانین فایروال اجازه دسترسی مستقیم به

---

[۶] Firewall

[۷] Objects

کلاینت[۸] را ندهد. اگر محیط توسعه نمی‌تواند این نیازمندی را برآورده کند داده حساس روی سرور DB توسعه ذخیره نشود و داده جعلی برای توسعه ساخته شود.

۱۱- رویه‌های کنترلی تغییر قوانین فایروال مشخص باشند و تغییرات قوانین به اطلاع مدیران سیستم (SA ها) و مدیران پایگاه داده (DBA ها) برسند.

۱۲- قوانین فایروال برای سرورهای DB به طور منظم توسط DBA ها و SA ها برقرار و بازنگری شوند.

۱۳- به طور منظم مقاومت سیستم و قوانین فایروال، از طریق پویش کردن شبکه تست شود.

موارد امنیتی احراز هویت و امنیت گذرواژه

مهمترین مواردی که باید در احراز هویت مورد توجه قرار گیرند، عبارتند از:

۱- روش‌های احراز هویت DB خود را شناخته و با در نظر گرفتن راهنمایی‌های این بخش از قوی‌ترین استفاده کنید.

۲- هرگز به احراز هویت داخل کلاینت‌هایی که به DB وصل می‌شوند اطمینان نکنید و برای DB حتماً یک گذرواژه تعریف کنید تا پس از اتصال کلاینت‌ها آن‌ها احراز هویت شوند. یعنی DB خود را به این تصور که کلاینت ها احراز هویت دارند، بی‌حفاظ رها نکنید.

۳- در مواردی که امکان احراز هویت روی کلاینت یا سرور وجود دارد احراز هویت روی سرور را انتخاب کنید. سپردن احراز هویت اصلی روی سرور به دلیل اینکه نظارت روی آن به مراتب بیشتر از کلاینت است، امری بدیهی است.

۴- در مواردی که امکان سپردن احراز هویت به سیستم عامل وجود دارد از احراز هویت سیستم عامل سرور استفاده کنید. مثلاً در DB2 UDB پس از نصب به صورت پیش‌فرض برای مدیریت DB یک کاربر ایجاد می‌کند که با ورود به آن می‌توان کارهای مدیریت DB را انجام داد و یا در Microsoft SQL Server دو شکل برای احراز هویت وجود دارد: احراز هویت توسط ویندوز و احراز هویت با هم آمیخته (احراز هویت ویندور و DB). اولی به صورت پیش‌فرض می‌باشد

---

[۸] Client

و توسط مایکروسافت نیز توصیه شده است.

۵- بفهمید چه کسانی امتیازهای مدیریت سیستم را می‌گیرند. مثلاً در ویندوز تمامی کاربرانی که جزء گروه مدیریت ویندوز هستند، از امتیازهای مدیریت سیستم برخوردارند.

۶- گذرواژه‌های قوی انتخاب کنید. یعنی به صورت پیش‌فرض رها نشوند، به راحتی قابل حدس زدن نباشند و به راحتی قابل کرک[۹] نباشند.

۷- گذرواژه‌ها با مقدار پیش‌فرض یا خالی رها نشوند. مثلاً گذرواژه SA در Microsoft SQL Server حتماً باید از مقدار پیش‌فرض تغییر کند.

۸- همواره ورودهای ناموفق به DB را مانیتور کنید. توجه کنید که در حالت عادی ورودهای ناموفق تنها باید متعلق به کاربران باشند زیرا برنامه‌های کاربردی مجاز هرگز ورود ناموفق ندارند زیرا نام کاربری و گذرواژه در آن‌ها به صورت کد شده، استفاده می‌شوند.

۹- جهت مبارزه با تلاش‌های هکرها یا افراد بدون حساب کاربری، با تعداد مشخصی تلاش برای ورود یا مدت مشخصی، حساب کاربری مربوطه را غیرفعال یا برای مدتی قفل کنید. اگر DB شما از این خصوصیت پشتیبانی نمی‌کند می‌توانید از ابزارهای بیرونی مانند فایروال DB استفاده کنید.

۱۰- با توجه به اینکه مورد ۹ ممکن است باعث طراحی نوعی حمله DOS توسط هکرها و از دسترس خارج کردن حساب‌های کاربری از کاربران مجاز باشد بهتر است بلوکه کردن یا ممانعت به جای اینکه روی حساب کاربری اعمال شود، روی اتصال[۱۰] انجام شود. مثلاً IP کاربر با تعداد بار تلاش ناموفق مشخص، اجازه دسترسی نداشته باشد. این کار را می‌توانید با فایروال های DB انجام دهید.

۱۱- چارچوب‌های مدیریت گذرواژه خوب را در قالب پروفایل‌های گذرواژه درست کنید و به اجرا درآورید. مثلاً اوراکل از پروفایل‌هایی برای قفل کردن حساب کاربری به مانند زیر استفاده می کند.

---

[۹] Crack

[۱۰] Connection

- PASSWORD_LIFE_TIME : تعداد روزهایی که گذرواژه برای احراز هویت استفاده می‌شود.
- PASSWORD_REUSE_TIME . : حداکثر تعداد روزهای مجاز، قبل از اینکه یک گذرواژه دوباره استفاده شود.
- و ...

۱۲- برای تمامی اجزای DB از گذرواژه استفاده کنید. البته ممکن است اجزایی وجود داشته باشند که نیاز به گذرواژه نداشته باشند مانند سرورهای توکار[۱۱]. توصیه ما این است که ساختار DB خود را مرور کنید و اجزای آن را به دقت بشناسید.

۱۳- درهای پشتی[۱۲] را درمکانیزم احراز هویت شناسایی کنید و با توجه به محیط کاری امن‌سازی کنید.

۱۴- حساب‌های کاربری با قابلیت مدیریت سیستم تا جایی که می‌شود برای کمترین افراد محیا شوند و تنها زمانی که واقعاً نیاز است این دسترسی به فردی داده شود.

۱۵- تمامی توسعه دهندگان، فروشندگان، SA ها، DBA ها و پیمانکاران توافق نامه عدم افشا اسرار را امضا کنند.

۱۶- سابقه توسعه دهندگان، SA ها، DBA ها و پیمانکاران براساس سیاست‌های سازمان باید بررسی شود.

۱۷- حساب کاربری استفاده شده توسط DBA برای ورود به ماشین DB جهت انجام وظایف مدیریتی باید حساب منحصر بفرد باشد و جزء یک گروه به اشتراک نباشد.

۱۸- داشتن یک گروه حساب کاربری برای انجام کارهای نگه‌داری و مانیتورینگ، از قبیل گرفتن نسخه پشتیبان بلا اشکال است. این گروه نباید برای کارهای روزمره و تعاملی استفاده گردد.

۱۹- کاربران باید حداقل دسترسی مورد نیاز برای انجام کار روی DB داشته باشند. دسترسی‌ها از طریق تعریف نقش‌ها و گروه‌ها مدیریت شوند.

۲۰- گذرواژه‌هایی که در DB ذخیره می‌گردند یا در شبکه ردوبدل می‌شوند رمز گردند.

---

[۱۱] Embedded

[۱۲] Back-door

۲۱- برنامه‌های کاربردی باید منحصرا دارای گذرواژه و دسترسی مشخص باشند. این اطلاعات نباید قابل دسترسی باشد.

موارد امنیت فیزیکی DB

۱- مکانی که میزبان ماشین DB است باید امن شده، قفل شده و محیطی مانیتور شده باشد تا از دسترسی غیر مجاز و دستبرد جلوگیری شود.

۲- توصیه می شود سرورهای وب و برنامه‌های کاربردی در همان جایی نباشند که سرور DB در آن قرار دارند.

موارد امنیت برنامه کاربردی

قسمت اعظمی از امن‌سازی برنامه‌های کاربردی، امن‌سازی داده برنامه کاربردی می‌باشد و به دلیل اینکه اغلب داده برنامه‌های کاربردی روی DB ها ذخیره می‌شوند بنابراین امن‌سازی داده برنامه‌های کاربردی، امن‌سازی دسترسی به DB است. علاوه بر آن استفاده کنندگان اولیه داده برنامه‌های کاربردی هستند، بنابراین پرداختن به امنیت DB بدون داشتن فهم از اینکه برنامه‌های کاربردی و آسیب‌پذیری‌هایشان چگونه می‌توانند بر امنیت DB تأثیر بگذارند کامل نمی شود. در این بخش نگاه ما به DB به عنوان نقش مرکزی در ساختار برنامه کاربردی می‌باشد با این فرق که DB می‌تواند در برابر سوء استفاده از برنامه کاربردی توسط حمله کننده از خود دفاع کند.

توصیه ها:

۱- مکان و چگونگی نگه‌داری نام‌های کاربری و گذرواژه‌ها را مرور کنید. نام کاربری و گذرواژه باید جایی امن ذخیره شوند. هیچ‌گاه نام کاربری و گذرواژه‌ها را به صورت غیر رمزشده در فایل‌های تنظیمات نگه‌ندارید.

۲- از استفاده از اسکریپت[۱۳]های اتصال خودکار به DB که در آن نام کاربری و گذرواژه نوشته شده است، خودداری

---

[۱۳] Script

کنید. بسیاری از توسعه دهندگان به دلایلی از جمله راحتی کار، اسکریپتی تهیه و آن را در پوشه مخصوص خود گذاشته و اجرا می‌کنند. این اسکریپت‌ها به سادگی قابل دسترسی هستند. برای رهایی از این مشکل می‌توانید از سیستم‌های احراز هویت متمرکز مانند LDAP استفاده کنید یا احراز هویت را به سیستم عامل واگذار کنید.

۳- از دستوراتی که در آن‌ها نام کاربری و گذرواژه وجود دارد و در صورت لیست کردن پروسه‌ها توسط شخصی دیگر آن پارامترها نمایان می‌شوند، اجتناب کنید. در بسیاری از DB ها مانند اوراکل، MySQL، Sybase و غیره دستور اتصال به DB با مخفی کردن گذرواژه نمایان می‌شود. مثلا اگر پروسه‌ها را لیست کنیم دستور MySQL به صورت زیر نمایان می‌شود:

```
mysql -uroot -pXXXXX DB
```

۵- کد برنامه کاربردیتان را مخفی کنید. در غیر این صورت هکرها می‌توانند اطلاعات ارزشمندی را درباره نحوه اتصال به DB پیدا کنند. شبه کدهایی که مثلاً در .NET یا جاوا نوشته می‌شوند به شکلی تبدیل کنید که مهندسی معکوس روی آن سخت باشد.

۶- DB خود را در برابر SQL injection امن سازید. راه حل به طور کلی به سه راهکار اصلی تقسیم می‌شود:

- محدود کردن آسیب‌پذیری‌های برنامه کاربردی
- کشف آسیب‌پذیری‌های SQL injection و رفع آن‌ها
- فیلتر کردن دستورات SQL که از برنامه کاربردی می‌آید.

توجه: در میان DB های مختلف SQL Server نسبت در برابر این نوع حملات شکننده‌تر است.

۷- DB خود را در برابر آسیب‌پذیری‌ها وصله[۱۴] کنید و از روش‌های پذیرفته شده استفاده کنید.

۸- همانطور که در بخش‌های قبل گفته شد ساختار امنیتی دارای لایه‌هایی است که در میان آن‌ها لایه DB با استفاده از یک فایروال در درون ساختار محافظت می‌شود. هرگز لایه برنامه کاربردی را از ساختار امنیتی حذف نکنید و به هیچ

---

[۱۴] Patch

وجه همه کارها (از قبیل پاسخگویی به کاربران) را روی خود DB انجام ندهید. ممکن است لایه برنامه کاربردی دارای معایب و باگ‌هایی باشد اما این کار معایب را از بین نمی‌برد بلکه همان معایب این بار روی DB مستقیماً اجرا می‌شوند.

۱۰- تمام سرورها، برنامه‌های کاربردی و ابزارها که به DB دسترسی دارند باید شناخته و مستند شوند.

۱۱- فایل‌های تنظیمات و کد منبع[۱۵] باید قفل شوند و فقط از طریق حساب‌های کاربری سیستم عامل قابل دسترسی باشند (با توجه به نوع دسترسی هر حساب).

امن‌سازی ارتباط DB به DB

ارتباط DB به DB چالش‌هایی را برای امنیت مطلوب بوجود می‌آورد. گرچه ممکن است گاهی اوقات این محیط داده توزیع شده برای ارتباط هر DB به DB دیگر مانند اتصال یک کلاینت به آن DB ساده سازی شود، اما ماهیت این ارتباط یعنی ارتباط DB به DB، با ماهیت ارتباط کلاینت با DB فرق می‌کند. در این بخش به بیان موارد امنیتی درباره این ارتباط می‌پردازیم.

توصیه‌ها:

۱- ارتباطات خروجی[۱۶] را محدود و مانیتور کنید. به عنوان نمونه کرمی به نام SQL Slammer اقدام به اشباع شبکه از بسته‌ها می‌کند که می‌توان با فیلتر کردن ترافیک خروجی از آن جلوگیری کرد.

۲- لینک‌های DB را امن کنید و مواظب ارتقای دسترسی از طریق لینک باشید. مورد امنیتی که در اینجا وجود دارد این است که اگر DB، الف با یک لینک به DB، ب وصل باشد و یک کلاینت درخواستی را به الف بدهد که در الف وجود نداشته باشد و در ب باشد، الف پس از احراز هویت و دسترسی توسط ب می‌تواند اطلاعات را دریافت نماید. درواقع اینجا دسترسی لینک مورد سنجش قرار می‌گیرد نه دسترسی کلاینت، که این خود می‌تواند تهدید امنیتی باشد. برای حل این مشکل باید یک نگاشت متناسب بین کاربران DB الف با ب برقرار گردد. یعنی مثلاً با استفاده از یک رویه

---

[۱۵] Source code

[۱۶] Outbound communication

بتوان سطح دسترسی کلاینت متصل به الف را تحت یک قرارداد به ب اعلام کرد.

۳- از نام کاربری و گذرواژه‌های لینک محافظت کنید. اولین قدم در اجرای امنیت لینک‌ها اطمینان از دسترسی مجاز به آن‌هاست. دسترسی به اطلاعات لینک‌ها که در DB نگه‌داری می‌شوند، باید امن شود و مانیتور گردد. مثلاً در اوراکل اطلاعات مربوط به لینک در SYS.LINK$ نگه‌داری می‌شود که گذرواژه و نام کاربری به صورت متن ساده ذخیره می‌شوند. بنابراین باید دسترسی به این جدول محافظت و مانیتور شود.

۴- استفاده از لینک‌های DB را مانیتور کنید. در دو حوزه مانیتورینگ باید انجام شود، اولی در دسترسی به تعریف لینک‌ها و دومی استفاده از آن‌ها. در مورد اول همواره باید ایجاد لینک‌های DB، تغییر آن‌ها و دسترسی به اطلاعات لینک مانیتور شود و در مورد دوم باید استفاده از لینک‌های DB را مانیتور کنید.

۵- مکانیزم‌های رونوشت برداری[۱۷] را امن کنید. این مورد در دو وجه بررسی می‌شود اولی امن‌سازی مکانیزم‌های رونوشت برداری (کی، کجا، چه چیزی رونوشت برداری شود)، اینکه هکرها نتوانند خللی در عملیات رونوشت برداری ایجاد کنند یا نتوانند یک کار رونوشت برداری جدید تعریف کنند و وجه دوم امن‌سازی ارتباطات و فایل‌هایی است که توسط مکانیزم‌های رونوشت برداری استفاده می‌شوند.

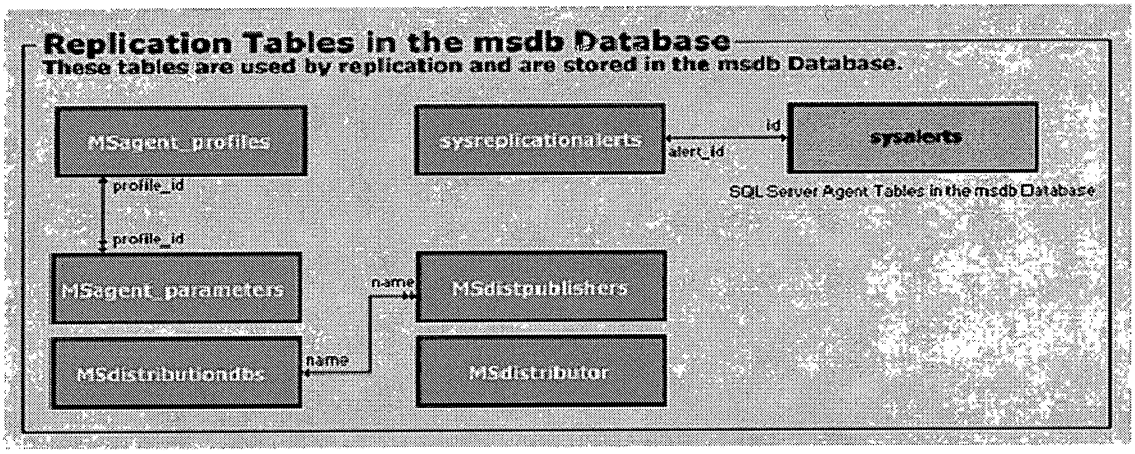
۶- در DB هایی که کار رونوشت برداری با استفاده از فایل‌های بیرونی انجام می‌شود حتماً فولدرها حاوی فایل‌ها را از نظر دسترسی امن‌سازی کنید به طوری که هیچ‌کس غیر از عامل رونوشت برداری، به آن دسترسی نداشته باشد. همچنین می‌توانید از رمزنگاری فایل‌ها جهت امن‌سازی دسترسی به آن‌ها استفاده نمایید.

۷- اتصالات و کاربران دخیل در رونوشت برداری را مانیتور کنید و امن سازید.

۸- دستوراتی که رونوشت برداری را تحت تأثیر قرار می‌دهند، مانیتور کنید. جداول مربوط به رونوشت برداری، رویه‌ها و دستوراتی که مربوط به رونوشت برداری هستند باید مانیتور شوند. مثلاً در SQL Server جداول زیر مربوط به رونوشت برداری هستند.

---

[۱۷] Replication



۸- دیگر موارد بالقوه نشت اطلاعات را مانیتور کنید. مثلاً SQL Server به شما اجازه نگه‌داری اطلاعات نشر[1]را داخل Active Directory می‌دهد و این یعنی هرگونه نشتی در Active Directory می‌تواند محیط رونوشت برداری شما را نمایان کند.

۱۰- تمامی مخازن و منابع داده را شناخته و در قالب یک نقشه ترسیم کنید و آن‌ها را امن سازید. تمام جریان داده را در محیط خودتان ترسیم کنید و مرور کنید که چگونه داده‌ها ذخیره می‌شوند؟ چه ID های کاربری استفاده می‌شوند؟ و مانیتورینگ چگونه بکار گرفته می‌شود؟

۱۱- log shipping را مانیتور کنید و امن گردانید. برای این منظور باید تمامی موارد گفته شده برای رونوشت برداری پیاده شود.

موارد امنیتی محافظت از DB در برابر تروجان ها

تروجان‌ها تهدیداتی هستند که به عنوان یکی از شکل‌های اصلی حمله شهرت زیادی کسب کرده‌اند. در اینجا منظور ما از تروجان فقط حملات و نوعی خرابکاری عمدی نیست بلکه تروجان‌ها می‌توانند حاصل اشتباهات و تنظیمات بد در کنترل باشند. یک توسعه دهنده می‌تواند به اشتباه یک باگ را در سیستم ایجاد کند یا با لاگ[2]های بیش از حد سرور را

---

[1] Publication
[2] Log

را از کار بیاندازد.

توصیه‌ها:

۱- برای جلوگیری از تروجان‌ها نیاز است که رویه‌های ذخیره شده مانیتور شوند. با توجه به حجم و تعداد بالای این رویه‌ها باید یک الگو از عملکرد طبیعی از آغاز تا پایان تهیه شود و هر رویه‌ای برخلاف آن عمل کرد عملیات‌های مختلف از قبیل لاگ انداختن یا اعلام خطر انجام شود. برای اینکار می‌توانید از فایروال‌های با قابلیت تعریف یک پایه[۱] و الگو برای روال عادی استفاده کنید. توجه داشته باشید که الگوها برای گروه‌های کاربری و اقسام رویه‌ها با هم متفاوت هستند.

۲- تولید و تغییر رویه‌ها و تریگر[۲]ها را کنترل کنید. برای این منظور باید اجرای رویه‌ها تولید یا تغییر آن‌ها دنبال شود و عکس‌العمل‌هایی همچون لاگ انداختن، دادن هشدار یا جلوگیری توسط فایروال انجام شود. دومین روش مقابله مانیتور کردن درخواست‌های تولید یا تغییرات اشیاء بر اساس دستورات SQL و همچنین تعریف یک قانون برای تأیید درخواست‌ها است. سرانجام نیز می‌توانید یک گروه از دستورات و رویه‌های سیستمی را که شما فکر می‌کنید ریسک دارند و ممکن است برای تزریق تروجان استفاده شوند تعریف کنید و بر اساس سیاست‌های خود با آن‌ها برخورد کنید.

۳- به دلیل اینکه در انواع حملات تروجان، اجرای کد تروجان توسط کاربر بی توجه انجام می‌شود و این کدها با سطح دسترسی او اجرا می‌گردند. بنابراین توابعی را که برای اجرا، نیاز به حداقل دسترسی آن کاربر را دارند باید با دقت مانیتور شوند.

۵- ساخت trace ها و event monitor ها را مانیتور کنید. اگر تروجان‌ها در این حوزه تزریق شوند به راحتی می‌توانند مانند آنچه که در key logger اتفاق می‌افتد، تمامی اطلاعات درباره DB از قبیل نام‌های کاربری، اطلاعات ترمینال و برنامه کاربردی و حتی در بعضی موارد گذرواژه‌ها را به حمله کننده ارسال کنند. برای مقابله دو انتخاب دارید یا به صورت پیوسته تمامی دستوراتی که اجرا می‌شوند را مانیتور کنید تا در صورت وقوع ساختن یا تغییر در اینگونه اشیاء

2Baseline
[۲]Trigger

هشدار داده شود و یا به صورت دوره‌ای trace ها و event monitor ها را مرور کنید. این فرآیند می‌تواند به صورت دستی یا خودکار انجام شود.

۶- ساخت job و scheduling را مانیتور و حسابرسی کنید. بسیاری از تروجان‌ها از job های زمان‌بندی شده برای انجام اعمال خود استفاده می‌کنند.

۷- مواظب پیوست[۱]های SQL در ایمیل[۲]ها باشید. بسیاری از تروجان‌های DB از طریق ایمیل منتقل می‌شوند. پیوست‌های SQL ایمیل را بدون بررسی باز نکنید.

رمزنگاری و مدیریت کلید

محرمانگی یک نیاز مهم در محافظت داده از حمله، خرابکاری یا دزدی است. محرمانگی اطلاعات موضوع یک علم جا افتاده به نام رمزنگاری است. در این بخش ما رمز کردن دو دسته داده، داده در حال انتقال و داده ساکن را بررسی خواهیم کرد.

توصیه‌ها:

۱- داده در حال انتقال رمز شود. به صورت کلی انتخاب‌هایی که در این حوزه برای رمزنگاری ارتباط DB وجود دارد به چهار دسته زیر تقسیم می‌شوند:

- مخصوص به DB (مانند Oracle Advance Security)
- روش‌های برمبنای اتصال (مانند SSL)
- تونل‌های امن (مانند SSH)

---

[۱] Attachment
[۲] E-mail

- تکیه بر سیستم عامل (مانند رمزنگاری IPsec)

انتخاب هرکدام از موارد به پارامترهای مختلفی از قبیل پشتیبانی DB، زیرساخت شبکه، کارایی و غیره بستگی دارد.

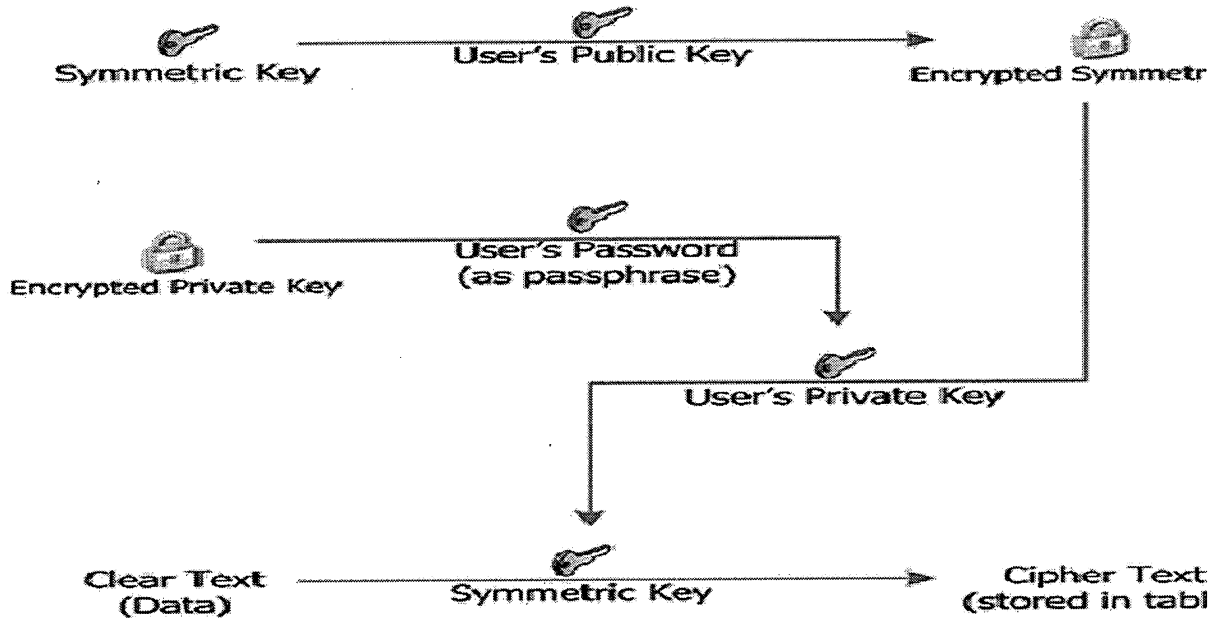
۲- داده ساکن را رمز کنید. یعنی داده‌هایی که داخل DB ذخیره شدند. این لایه اضافی امنیت، اغلب برای داده‌های حساس که میزان محرمانگی بالایی دارند و باید نسبت به داده‌های دیگر بیشتر محافظت شوند مانند گذرواژه‌ها، اطلاعات بانکی و غیره استفاده می‌شوند. برای اینکار سه گزینه پیش رو داریم رمزکردن در برنامه کاربردی، رمزکردن فایل و رمزکردن در DB که رمزکردن در DB عملی‌ترین گزینه می‌باشد.

۳- برای رمزکردن داده‌هایی که به هم مربوط نیستند باید از کلیدهای متفاوت استفاده کرد تا مطمئن شویم که کسی که با داشتن یک کلید که به بخشی از داده‌ها دسترسی دارد به بخش دیگر دسترسی نداشته باشد.

۴- به همه کاربران یک کلید عمومی و یک کلید خصوصی داده شود. کلید خصوصی هر کاربر نوعاً بوسیله گذرواژه کاربر محافظت می‌شود و درواقع نزد او محفوظ است.

۵- هرگاه به کاربری اجازه دسترسی به داده رمزشده داده شد، کلید متقارنی که برای کشف رمز مورد استفاده قرار می‌گیرد با کلید عمومی کاربر رمز شود و بعد در دسترس قرار گیرد.

۶- هرگز کلیدها را به عنوان داده روی جداول ذخیره نکنید و از ابزارهای مربوطه برای مدیریت کلید استفاده نمایید.



۷- فرآیند بازگشتی برای زمانی که کلید گم شد جهت از دست ندادن داده از قبل تعریف نمایید.

۸- تأثیر رمزنگاری را بر پشتیبان گیری[۱] و برگرداندن پشتیبان[۲] بررسی کنید و راه‌حل‌های مناسب تهیه نمایید. مثلاً پشتیبان چگونه ذخیره شود رمز شده یا نشده؟ تغییر دوره‌ای کلیدها در برگرداندن پشتیبان چگونه تأثیر می‌گذارد؟ و ...

۹- در صورتی که در یک محیط توزیع شده کار می‌کنید، رمزنگاری چگونه روی آن اثر می‌گذارد؟ کلیدها چگونه به اشتراک گذاشته شوند؟ و ...

۱۰- تأثیر رمزنگاری روی رونوشت برداری چگونه است؟

۱۱- تأثیر رمزنگاری روی کارائی چگونه است؟ با توجه به نیازهایتان کدام الگوریتم رمزنگاری مناسب است؟

۱۲- تأثیر رمزنگاری روی فضای تحت اشغال دیسک چگونه است؟

۱۳- تأثیر رمزنگاری روی حسابرسی چگونه است؟

موارد ۷ الی ۱۳ مواردی هستند که باید با توجه به سیاست‌های سازمان، DB مورد استفاده و موارد دیگر در مورد

---

[۱] Backup
[۲] Recovery

استفاده از رمزنگاری، هرکدام به دقت بررسی شوند.

۱۴- فقط چیزهای مهم رمز شوند.

۱۵- ستون‌هایی که به عنوان کلید یا ایندکس[۳] استفاده می‌شوند رمز نشوند.

موارد امنیتی درباره نرم افزار DB

در این بخش به طور خلاصه توصیه‌هایی امنیتی درباره نرم افزار DB خواهیم داشت.

توصیه‌ها:

۱- نسخه نرم‌افزار استفاده شده، توسط شرکتش یا پروژه متن‌بازش پشتیبانی شود.

۲- فایل‌های موقتی باقی‌مانده از فرآیند نصب که ممکن است شامل گذرواژه باشند حذف گردند.

۳- نرم افزار DB به طور مداوم وصله شود.

۴- از رویه‌های خارجی[۴] استفاده نکنید.

۵- DB را سرور وب نکنید و سرور وب را مستقیماً به رویه‌های ذخیره شده متصل نکنید زیرا رویه های ذخیره شده قسمتی از DB می‌باشند و برای این کار مناسب نیستند.

۶- هرگز داخل رویه‌های ذخیره شده HTML نسازید.

موارد امنیتی مربوط به داده محرمانه[۵]

---

[۳] Index

[۴] External procedures

[۵] Restricted data

در این بخش به طور خلاصه مواردی را در حوزه امنیت داده محرمانه روی DB ذکر می‌کنیم.

توصیه‌ها:

۱- تنها، داده محرمانه‌ای را که برای کار نیاز است در DB نگه‌دارید. هرگاه ممکن است اطلاعات قدیمی که دیگر نیازی به آن‌ها وجود ندارد، پاکسازی شوند.

۲- افزونگی داده محرمانه را در سراسر سیستم حذف کنید. اگر تنها برای فرآیند تطبیق آن‌ها را نیاز دارید، عناصر داده محرمانه با اعمال توابع درهم ساز[۶] ذخیره گردند.

۳- در صورت امکان داده محرمانه را از اطلاعات فردی قابل شناسایی جدا کنید و آن‌ها را تا زمانی که نیازی پیش نیامده به صورت آفلاین نگه دارید.

۴- در صورت انتقال داده برای برنامه‌های کاربردی دیگر، نیاز است آن‌ها را از وجود داده محرمانه و نیازمندی‌های امنیتی آن آگاه کنید.

۵- داده محرمانه در محیط غیر تولید (مانند محیط توسعه) با همان استانداردهای محیط تولید باید نگه‌داری شود. در غیر این صورت داده در این محیط‌ها یا باید با استفاده از الگوریتم‌های استاندرد صنعتی رمز گردد یا اینکه برای این سیستم‌ها داده تست فراهم گردد.

۶- عناصر داده محرمانه که در داخل DB هستند، مستند شوند.

۷- داده محرمانه به هیچ وجه به عنوان کلید داخل جداول استفاده نشود.

موارد امنیتی مربوط به پشتیبان گیری و برگرداندن پشتیبان

[۶] Hash function

در این بخش توصیه‌هایی امنیتی درباره فرآیند تهیه نسخه پشتیبان و استفاده از آن را به طور خلاصه خواهیم داد.

توصیه‌ها:

۱- رویه‌های پشتیبان گیری و برگرداندن آن مستند شوند و نیازهای صاحب داده را برآورده کنند.

۲- رویه‌های پشتیبان گیری و برگرداندن آن به صورت دوره‌ای تست شوند.

۳- فواصل نگه‌داری نسخه پشتیبان مستند شود و طوری باشد که بتواند جوابگوی نیازهای کاری و توقعات صاحب داده باشد.

حسابرسی

در این بخش ما معرفی کوتاهی از حسابرسی و دسته‌های مختلف آن انجام می‌دهیم. هرکدام از دسته‌ها ممکن است برای محیط شما جهت برآوردن نیازمندی‌هایتان، ضروری باشد. چون DB از نظر توانایی‌ها بسیار غنی است، شما می‌توانید بسیاری از انواع دنباله‌های حسابرسی را برای DB خود تعریف کنید. این به معنی این نیست که تمامی دسته-هایی که در این بخش گفته می‌شوند برای DB شما مناسب و لازم هستند. استفاده از هرکدام از آن‌ها بستگی به نیازمندی‌های شما دارد. یک پیاده‌سازی خوب برای حسابرسی مستلزم فهم درست نیازمندی‌ها می‌باشد.

توصیه‌ها:

۱- ورود و خروج به DB را حسابرسی کنید. در این حوزه می‌توان سه دسته تعریف کرد، ورود موفق، خروج موفق و ورود ناموفق که آخری از نظر امنیتی مهم‌تر می‌باشد. برای مقابله همانطور که در بخش‌های گذشته اشاره شد، می‌توان تولید یک هشدار نمود یا با استفاده از فایروال راه آن را سد کرد یا می‌توان به صورت پیشرفته‌تر یک الگو از ورود و خروج موفق با توجه به محیطی که DB در آن قرار دارد درست نمود و با توجه به آن، خطرات احتمالی را شناسایی نمود.

۲- مبداءهای استفاده از DB را حسابرسی کنید. اطلاعاتی از قبیل گره، IP شبکه شخص یا برنامه متصل شده به DB.

اطلاعاتی از قبیل نام ماشین در مقایسه با IP می‌توانند حاوی اطلاعات بیشتری باشند.

۳- استفاده از DB را خارج از ساعات کاری معمول حسابرسی کنید. این مورد به نوع کار در محیط DB بستگی دارد. ممکن است بعضی کارهای برنامه‌ریزی شده در خارج از وقت معمول نیز انجام شوند. بنابر این می‌باید اینگونه کارها به عنوان اعمال معمول و نرمال هشداری تولید نکنند تا باعث بالا رفتن false-positive نگردند.

۴- فعالیت‌های DDL را حسابرسی کنید. دستورات DDL چون در ساخت شمای DB دخالت دارند می‌توانند به صورت بالقوه خطرناک باشند.

۵- پیغام‌های خطای DB را حسابرسی کنید. این نوع حسابرسی بسیار مهم است و باید یکی از دنباله حسابرسی‌هایی باشد که شما پیاده می‌کنید.

۶- تغییرات در منابع کد رویه‌های ذخیره شده و تریگرها را حسابرسی کنید. یعنی تغییرات در کد رویه‌ها را با روش‌های مختلف که در سه دسته کلی چک کردن متناوب کد و مقایسه آن با کد قبلی، استفاده از سیستم حسابرسی و امنیت DB خارج از DB و استفاده از امکانات داخلی DB قرار می‌گیرند، بررسی شوند.

۷- تغییر در دسترسی‌ها، تعریف کاربر و ورود[۷] و دیگر موارد امنیتی را حسابرسی کنید. این دسته حتماً باید در حسابرسی DB حضور داشته باشد. تغییرات در موارد زیر باید مورد حسابرسی قرار گیرد:

- اضافه یا حذف کردن کاربر، ورود و نقش[۸]
- تغییر نگاشت بین ورود و کاربران یا نقش‌ها
- تغییر اجازه دسترسی‌ها[۹] چه برای کاربر و چه برای یک نقش
- تغییر گذرواژه‌ها
- تغییرات در موارد امنیتی در سطح سرور، DB، دستور یا شیء

---

[۷] Login
[۸] Role
[۹] Privilege

۸- ساخت، تغییر و استفاده از لینک‌های DB و رونوشت برداری را حسابرسی کنید.

۹- تغییر در داده‌های حساس را حسابرسی کنید.

۱۰- دستورات SELECT را برای مجموعه‌های محرمانه حسابرسی کنید.

۱۱- هرگونه تغییری را روی تعریف آنچه که باید حسابرسی شود، حسابرسی کنید.

در بخش‌های بالا سعی شد به طور خلاصه به امنیت DB از زوایای مختلف پرداخته شود و توصیه‌هایی عمومی بدون توجه به اینکه DB چه باشد ارائه شد. قسمت اعظمی از این توصیه‌ها با دیدی مفهومی به مقوله امنیت، در قالب یک تکنولوژی یا ساختار خاص نمی‌گنجد. در ادامه به عنوان نمونه دو چک لیست از دو DB پرکاربرد در کشور آورده شده است. بدیهی است که مواردی که گفته می شود ممکن است باگذشت زمان دچار تغییرات شوند.

چک لیست mySQL

بند	توصیه امنیتی	YES/ NO
۱.	سروری که mySQL روی آن است را از نظر فیزیکی امن سازی کنید	
۲.	در تنظیمات:0=local-infile-- (جهت غیر فعال سازی LOCAL در دستورات LOAD DATA)	
۳.	در تنظیمات: safe-show-database-- (جهت اطمینان از اینکه دستور SHOWDATABASE متناسب با سطح دسترسی کاربر اطلاعات نشان داده می‌شود)	
۴.	در تنظیمات: safe-user-create-- (جهت اطمینان از امن بودن و عدم دسترسی کاربر غیر مجاز برای ساختن کاربر)	

۵.	در تنظیمات: secure-auth-- (عدم اجازه به گذرواژه های قبل از نسخه ۴,۱)	
۶.	در تنظیمات: skip-name-resolve--	
۷.	در تنظیمات: skip-symbolic-links-- (جهت عدم اجازه استفاده از symbolic link روی جداول تحت لینوکس)	
۸.	در تنظیمات: از skip-grant-table-- استفاده نکنید	
۹.	در تنظیمات: از enable-named-pipe-- تحت ویندوز استفاده نکنید (از دسترسی شبکه تحت TCP/IP به جای آن استفاده کنید)	
۱۰.	اجازه دسترسی SUPER، FILE یا PROCESS را به کاربران غیر مدیر ندهید	
۱۱.	حتی الامکان MySQL را روی همان میزبانی که سرور وب روی آن قرار دارد، قرار ندهید	
۱۲.	از قوی بودن گذرواژه کاربر root مطمئن شوید	
۱۳.	تنظیمات پیش‌فرض را تغییر دهید	
۱۴.	دسترسی به تابع Load-file را محدود کنید	
۱۵.	دسترسی ها به بارگذاری داده در فایل یا SELECT داخل فایل را محدود کنید	
۱۶.	اجازه دسترسی توسعه دهندگان به قسمت تولید را ندهید	
۱۷.	قابلیت حسابرسی را فعال سازید	

چک لیست Microsoft SQL Server

بند	توصیه امنیتی	YES/ NO
۱۸.	تمامی service pack ها و hotfix ها را هم برای ویندوز و هم برای SQL server اعمال کنید	
۱۹.	مطمئن شوید که تمامی فایل‌های داده و سیستمی SQL روی پارتیشن NTFS نصب شده‌اند و دارای سطح دسترسی درست و مناسب هستند	
۲۰.	فایل‌های نصب را پاک کنید	
۲۱.	تمامی کاربران و DB های نمونه را پاک کنید	
۲۲.	تمام گذرواژه ها را دوباره بررسی کنید	
۲۳.	چگونگی تعلق نقش‌ها به کاربران در DB و سرور دوباره بررسی شود	
۲۴.	به طور مرتب نقش‌ها و عضویت گروه را بررسی کنید	
۲۵.	کتابخانه‌های شبکه‌ای که مورد نیاز نیستند را حذف کنید	
۲۶.	هرگز اجازه دسترسی از راه دور به سیستم عامل DB را ندهید	
۲۷.	دسترسی به رویه های extended (رویه های XP--) را حذف کنید یا محدود کنید	
۲۸.	رویه ها extended ساخته شده توسط کاربر را هرگز نصب نکنید	
۲۹.	قابلیت‌های ایمیل SQL را غیرفعال کنید (به دنبال راه‌های جایگزین برای روش‌های اعلام کردن بگردید)	
۳۰.	درصورت عدم نیاز قابلیت جستجوی full-text را نصب نکنید	
۳۱.	در صورت عدم نیاز Microsoft Distributed Transaction Cordinator	

	را نصب نکنید	
۳۲.	نبود تروجان های startup را بررسی کنید (اطمینان حاصل کنید که هیچ فراخوانی مبهمی در master..sp-helpstartup نیست)	
۳۳.	تروجان های مربوط به گذرواژه ها را چک کنید (بوسیله مقایسه master..sp-password با نمونه آن در نرم‌افزار تازه نصب شده)	
	دسترسی محدود به داده – دسترسی به سرور	
۳۴.	ورودها از طریق گروه‌های کاربری در ویندوز مدیریت شوند	
۳۵.	کاربران تنها به همان اندازه‌ای که برای کار لازم دارند دسترسی داشته باشند	
۳۶.	برای دیدن فرا داده (metadata) سیستم دسترسی VIEW DEFINITION تعریف شود	
۳۷.	سرورهای از راه دور (Remote Servers) را با سرورهای متصل (Linking Servers) جایگزین کنید	
۳۸.	غیر از زمان‌هایی که نیاز است زدن کوئری‌های ad hoc از طریق سرورها غیرفعال شود	
	دسترسی محدود به داده – مدیریت هویت کاربر	
۳۹.	کاربر Guest در تمامی DB ها مگر جایی که لازم باشد کاربر ناشناس وارد شود را غیر فعال کنید	

۴۰.	کاربران تنها به DB های مورد نیازشان دسترسی داشته باشند	
۴۱.	اجازه دسترسی کاربران از طریق نقش‌ها تعریف شود	
۴۲.	SQL Server Agent برای کارهای با دسترسی فراتر از دسترسی خود، از اعتبارنامه (credentials)ها استفاده کند	
۴۳.	دسترسی به اشیاء DB را داخل ماژول‌هایی از قبیل رویه‌های ذخیره شده، توابع، تریگرها و اسمبلی‌ها محدود شود	
۴۴.	در ماژول‌ها کدهای متفاوت با هم با تعریف بافت‌های مختلف مثلاً global، func code و غیره مشخص کنید و همه کدها را در بافت پیش‌فرض ننویسید	
۴۵.	از مکانیزم‌های جلوگیری از دستکاری غیرمجاز مانند امضا کردن برای ماژول ها استفاده کنید	
۴۶.	به جای SETUSER از EXECUTE AS استفاده کنید	
۴۷.	نقش‌های برنامه کاربردی را با EXECUTE AS جایگزین کنید	
دسترسی محدود به داده – دسترسی به اشیاء		
۴۸.	میزان دسترسی بسیار پایین (اگر نگوییم هیچ دسترسی) به سرورها یا نقش‌های همگانی داده شود	
۴۹.	اشیاء شبیه به هم در یک شما به صورت گروه در آیند	
۵۰.	از شماهای شخصی سازی شده به جای dbo استفاده کنید	
۵۱.	هر شما مالک مخصوص به خود داشته باشد	
۵۲.	TRUSTWORTHY را خاموش کنید	

دسترسی محدود به داده – دسترسی به حسابرسی		
۵۳.	سناریویی خاص با اهداف مشخص برای حسابرسی داشته باشید	
۵۴.	ورودهای ناموفق و موفق برای داده‌های حساس، حسابرسی شوند	
۵۵.	DML،DDL و رخداده‌های سروری را حسابرسی کنید	
۵۶.	از WMI برای هشدار دادن رخدادهای اورژانسی استفاده کنید	
۵۷.	حسابرسی C2 تنها زمانی که لازم است، فعال کنید	
تنظیمات امنیتی موتور DB – امنیت فیزیکی		
۵۸.	موتور DB در مرکز داده امن با دسترسی محدود قرار داشته باشد	
۵۹.	نسخه‌های پشتیبان در جایی امن نگه داری شوند	
۶۰.	دسترسی به فایل‌های موتور (ldf، mdf.، ndf.) را محدود شود	
۶۱.	دسترسی به فایل‌های باینری موتور (sqlserver.exe در فولدر binn) را محدود کنید	
۶۲.	دسترسی به فایل‌های حسابرسی محدود شود	
۶۳.	نسخه‌های پشتیبان کلیدهای رمزنگاری خصوصی و عمومی در محلی امن نگه داری شوند	
تنظیمات امنیتی موتور DB – تنظیمات		
۶۴.	احراز هویت از طریق ویندوز انجام شود	
۶۵.	اگر احراز هویت SQL فعال است، حساب کاربری SA غیر فعال باشد یا حداقل نامش تغییر کند	
۶۶.	اگر حساب کاربری SA فعال است، گذرواژه‌ای قوی داشته باشد	

۶۷.	خصوصیات غیر لازم غیرفعال باشد	
۶۸.	xp_cmdshell غیر از زمانی که مطلقاً مورد نیاز است، غیر فعال باشد	
۶۹.	Cross ownership chaining غیر از زمانی که چندین DB به صورت یک واحد توسعه داده شده‌اند، خاموش باشد	
۷۰.	مرتبا Best Practices Analyzer را روی DB اجرا کنید	
۷۱.	آخرین service pack ها روی موتور اعمال شوند	
	رمزنگاری داده حساس – سطح DB	
۷۲.	رمزنگاری داده را با استفاده ازTDE مورد ارزیابی قرار دهید	
۷۳.	از کلیدهای متقارن برای رمزکردن و از کلیدهای نامتقارن برای امضا و محافظت از کلیدهای متفارن استفاده کنید	
۷۴.	از امضاهای دیجیتال نسخه پشتیبان تهیه کنید	
	رمزنگاری داده حساس – سطح سلول یا ستون	
۷۵.	اطلاعات حساس و مهم (ازقبیل اطلاعات کارت‌های اعتباری) ذخیره شده روی ستون‌ها را رمز کنید	
۷۶.	الگوریتمی مناسب برای رمزنگاری انتخاب کنید	
۷۷.	از درهم‌سازی برای ذخیره گذرواژه‌ها و دیگر اطلاعات از این قبیل استفاده کنید	
۷۸.	در ساخت کلید متقارن از آرگومان‌های KEY_SOURCE و IDENTITY VALUE جهت قابلیت بازگشت و داشتن پشتیبان استفاده کنید	

۷۹.	می‌توانید از احراز هویت برای امنیت بیشتر داده‌های رمز شده استفاده کنید	
	اتصال DB – تنظیمات موتور DB	
۸۰.	موتور تنها به پروتوکل‌هایی که واقعاً نیاز است گوش دهد	
۸۱.	Service Broker، HTTP و database-morroring را تنها زمانی که نیاز است درست کنید	
۸۲.	اتصالات را با استفاده از احراز هویت ویندوز هر زمان که لازم بود برقرار کنید	
۸۳.	موتور برای ارتباط امن از امضای دیجیتال معتبر استفاده کند	
۸۴.	از تنظیم Force Encryption جهت امنیت اتصالات موتور استفاده کنید	
۸۵.	برای احراز هویت کلاینت ها حتی الامکان از کربروس (Kerberos) استفاده کنید	
۸۶.	روی ویندوزهای سرور ۲۰۰۳ و XP موتور DB تحت دامنه‌ای فعالیت کند که توسط هیچ سرویس دیگر مورد استفاده قرار نمی‌گیرد	
۸۷.	گذرواژه حساب کاربری موتور را به طور منظم تغییر دهید	
۸۸.	Name instance های DB به پورت‌های ثابتی گوش دهند تا دنبال کردن آن‌ها مثلاً در فایروال آسان‌تر باشد	
	اتصال DB – تنظیمات کلاینت	
۸۹.	کلاینت‌ها با SSL متصل شوند	
۹۰.	حداقل دسترسی مورد نیاز به حساب‌های کلاینت‌ها داده شود	
۹۱.	مدیران با استفاده از حساب‌های کاربری ویندوز وارد شوند نه sa	

۹۲.	زمانی که مدیران کار مدیریتی انجام نمی‌دهند باید حساب با دسترسی کمتر استفاده کنند	
اتصال DB – تنظیمات سیستم عامل		
۹۳.	کامپیوتر سرور توسط یک فایروال که به صورت پیش‌فرض اجازه عبور نمی دهد، محافظت شود	
۹۴.	سیستم‌های عامل کلاینت و سرور طوری تنظیم شوند که از Extended Protection برای احراز هویت استفاده کنند	
۹۵.	سیستم عامل خود را همواره بروز نگه دارید	

منابع

۱. Natan, Ron Ben. *Implementing Database Security and Auditing*. Digital Press, 2005.

۲. David Litchfield, Chris Anley, John Heasman, and Bill Grindlay. 2005. *The Database Hacker's Handbook: Defending Database Servers*. John Wiley & Sons.

۳. security.berkeley.edu

۴. www.greensql.com

پیوست ۱ : نصب و راه‌اندازی امن MySQL

MySQL یکی از پایگاه داده‌های شناخته شده روی اینترنت است که اغلب در کنار PHP استفاده می‌شود. در کنار فواید بسیار MySQL مانند استفاده آسان و کارایی بالا، شیوه نصب پیش فرض MySQL مخصوصا رمز عبور خالی root و برخی آسیب‌پذیری های بالقوه در برابر حملات سرریز بافر، این پایگاه داده را به هدف ساده‌ای برای مهاجمین تبدیل کرده است. این مستند جهت امن‌سازی هر چه بیشتر MySQL برای استفاده کاربران تهیه گردیده است. در ادامه برخی اقدامات لازم جهت امن‌سازی MySQL در برابر حملات مختلف آورده شده است.

در این مستند فرض شده است که وب سرور آپاچی همراه با ماژول PHP روی سیستم و در مسیر */chroot/httpd* نصب شده‌اند. همچنین فرض شده است که MySQL تنها توسط برنامه‌های کاربردی PHP مورد استفاده قرار می‌گیرد و روی همان host نصب شده است. گرچه این اقدام از نظر امنیتی صحیح نمی‌باشد، اما برای توضیح موارد امنیتی به شکلی

ساده‌تر این فرض انجام شده است.

۱. نیازمندی های امنیتی

به منظور دستیابی به بالاترین سطح امنیت لازم است تا موارد زیر در نصب و راه اندازی، رعایت شوند:

- پایگاه داده MySQL باید در محیط chroot اجرا شود.
- پروسه‌های پایگاه داده MySQL بایستی تحت UID/GID که توسط پروسه‌های سیستم دیگری استفاده نشده باشد، اجرا شوند.
- تنها دسترسی local به MySQL مجاز باشد.
- حساب کاربری MySQL root بایستی توسط یک رمز عبور امن که به سختی قابل حدس زدن باشد، حفاظت گردد.
- نام حساب کاربری Admin باید تغییر کند.
- دسترسی همزمان به پایگاه داده به وسیله حساب کاربری nobody (خالی) غیر فعال گردد.
- تمام پایگاه‌های داده نمونه به خصوص پایگاه داده TEST باید حذف شوند.

۲. نصب MySQL

پیش از امن‌سازی باید MySQL را نصب کنیم. همان طور که ذکر شد برای نصب، یک گروه واحد و حساب

کاربری روی سیستم ایجاد می‌نماییم که تنها به پایگاه داده MySQL اختصاص داشته باشد:

```
pw groupadd mysql
pw useradd mysql -c "MySQL Server" -d /dev/null -g mysql -s /sbin/nologin
```

۱,۲ کامپایل

MySQL را در مسیر /usr/local/mysql/ کامپایل و نصب می‌کنیم.

```
./configure --prefix=/usr/local/mysql --with-mysqld-user=mysql --with-unix-socket-
path=/tmp/mysql.sock --with-mysqld-ldflags=-all-static
make
su
make install
strip /usr/local/mysql/libexec/mysqld
scripts/mysql_install_db
chown -R root /usr/local/mysql
chown -R mysql /usr/local/mysql/var
chgrp -R mysql /usr/local/mysql
```

عموما فرآیند نصب سرور نیز شبیه آنچه دربالا آمده است. تنها تغییر در برخی پارامترهای افزوده شده در دستور configure/ می‌باشد. تفاوت مهم در استفاده از پارامتر --static-all =with-mysqld-ldflags- که موجب می‌شود به طور ایستا لینک گردد، است.

۲,۲ کپی کردن فایل تنظیمات

پس از اجرای دستورات بالا بایستی فایل تنظیمات پیش فرض را مطابق با اندازه پایگاه داده (کوچک، بزرگ) کپی نمایید به عنوان نمونه:

```
cp support-files/my-medium.cnf /etc/my.cnf
chown root:sys /etc/my.cnf
chmod 644 /etc/my.cnf
```

۳,۲ روشن کردن MySQL server

در این مرحله MySQL کاملا نصب شده است و می توان با اجرای دستور زیر آن را روشن نمود.

```
/usr/local/mysql/bin/mysqld_safe &
```

۴,۲ امتحان اتصالات

اتصالات به پایگاه داده را از طریق زیر برقرار سازید.

```
/usr/local/mysql/bin/mysql -u root mysql Welcome to the MySQL monitor.  Commands
end with ; or \g. Your MySQL connection id is 2 to server version: 4.0.13-log Type
'help;' or '\h' for help. Type '\c' to clear the buffer. Mysql> show databases; +--
--------+ | Database | +----------+ | mysql    | | test     | +----------+ 2 rows
                                    in set (0.00 sec) mysql> quit;
```

به محض اینکه اتصال با موفقیت برقرار شد می‌توانید پایگاه داده را خاموش نمایید.

```
/usr/local/mysql/bin/mysqladmin -u root shutdown
```

و امن‌سازی نرم‌افزار را آغاز نمایید. به عبارت دیگر باید اطلاعات ذخیره شده در فایل لاگ usr/local/mysql/var/hostname.err/ را بررسی نموده و مشکلات احتمالی را برطرف سازید.

۳. ساختن سرور Chroot

اولین گام در امن‌سازی MySQL فراهم‌سازی محیط chroot شده، می‌باشد که MySQL در آن اجرا خواهد شد. جهت ایجاد محیط chroot ساختار دایرکتوری به شکل زیر ایجاد می‌کنیم

```
mkdir -p /chroot/mysql/dev
mkdir -p /chroot/mysql/etc
mkdir -p /chroot/mysql/tmp
mkdir -p /chroot/mysql/var/tmp
mkdir -p /chroot/mysql/usr/local/mysql/libexec
mkdir -p /chroot/mysql/usr/local/mysql/share/mysql/english
```

۱,۳ تنظیمات سطح دسترسی

سطح دسترسی به پوشه‌های ایجاد شده در بخش قبل می‌بایست به شکل زیر باشد:

```
chown -R root:sys /chroot/mysql
chmod -R 755 /chroot/mysql
chmod 1777 /chroot/mysql/tmp
```

۲,۳ ایجاد ساختار دایرکتوری:

در گام بعدی فایل‌های زیر باید در مسیر جدید به شکل زیر کپی شوند:

```
cp /usr/local/mysql/libexec/mysqld /chroot/mysql/usr/local/mysql/libexec/
cp /usr/local/mysql/share/mysql/english/errmsg.sys
/chroot/mysql/usr/local/mysql/share/mysql/english/
cp /etc/hosts /chroot/mysql/etc/
cp /etc/host.conf /chroot/mysql/etc/
cp /etc/resolv.conf /chroot/mysql/etc/
cp /etc/group /chroot/mysql/etc/
cp /etc/master.passwd /chroot/mysql/etc/passwords
cp /etc/my.cnf /chroot/mysql/etc/
```

۳,۳ ایجاد یک رمز عبور مطمئن:

از فایل‌های /chroot/mysql/etc/passwords و /chroot/mysql/etc/group تمامی خطوط بجز mysql account و group را حذف کنید. سپس رمز عبور پایگاه داده را به شکل زیر بسازید.

```
cd /chroot/mysql/etc
pwd_mkdb -d /chroot/mysql/etc passwords
rm -rf /chroot/mysql/etc/master.passwd
```

ملاحظات خاص :

باید یک device file خاص /dev/null ایجاد کنید:

```
ls -al /dev/null  crw-rw-rw-  1 root  sys    2,   2 Jun 21 18:31 /dev/null mknod
    /chroot/mysql/dev/null c 2 2 chown root:sys /chroot/mysql/dev/null chmod 666
                                                    /chroot/mysql/dev/null
```

همچنین باید پایگاه داده MySQL را که شامل grant table هایی می‌شود که در زمان نصب ایجاد شده‌اند کپی کنید.

```
cp -R /usr/local/mysql/var/ /chroot/mysql/usr/local/mysql/var
Chown -R mysql:mysql /chroot/mysql/usr/local/mysql/var
```

در این مرحله MySQL آماده اجرا در محیط chroot است. توسط دستور زیر می‌توانیم اجرای صحیح آن را تست کنیم.

```
chrootuid /chroot/mysql mysql /usr/local/mysql/libexec/mysqld &
```

در صورت بروز هر خطایی از دستور truss یا دستورات مشابهی مانند ktrace/kdump, strace, استفاده کنید. که کمک می‌کند تا دلیل بروز خطا را مشخص نموده و آن را برطرف سازید.

توجه داشته باشید که جهت اجرای process mysqld برنامه chrootuid به جای chroot (در آپاچی و PHP)، استفاده شده است. همان طور که ذکر شد mysqld در محیط chroot اجرا می‌شود و owner آن کاربر mysql است نه root.

Chrootuid به طور پیش فرض روی اکثر سیستم‌ها نصب نیست و ممکن است لازم باشد تا آن را نصب کنید.

۴. تنظیمات سرور:

مرحله بعدی انجام تنظیمات سرور پایگاه داده، هماهنگ با نیازهای امنیتی است. در نصب MySQL فایل اصلی تنظیمات /etc/my.cnf، می‌باشد. در اینجا به دلیل اجرای سرور در محیط chroot از دو فایل/etc/my.cnf و /chroot/mysql/etc/my.cnf استفاده می‌شود. اولی توسط MySQL server استفاده شده و دومی توسط ابزارهای MySQL مانند mysqladmin، mysqldump مورد استفاده قرار می‌گیرد. که در هر دو برخی تغییرات باید صورت گیرد.

۱,۴ غیر فعال کردن دسترسی از راه دور:

اولین تغییر روی پورت tcp 3306 که به طور پیش‌فرض MySQL به آن گوش می‌کند، انجام می‌شود. در صورتی که پایگاه داده توسط application های php که به صورت local نصب شده‌اند مورد استفاده قرار گیرد به راحتی می‌توان این پورت را غیر فعال کرد که این مسئله احتمال حمله به پایگاه داده MySQL از طرف سایر میزبان‌ها و از طریق اتصال‌های مستقیم tcp/ip را بسیار محدود خواهد ساخت. (چنانچه نیاز بود تا دسترسی‌های غیر محلی صورت پذیرد، می‌توانید نام پورت را در /etc/my.conf تغییر دهید و البته این تغییر پورت باید با سایر application های مورد استفاده هماهنگ گردد.) اما دسترسی‌های محلی همچنان از طریق سوکت mysql.sock ممکن خواهد بود. جهت غیر فعال کردن شنیدن روی پورت مذکور پارامتر زیر باید به بخش [mysqld] از فایل /chroot/mysql/etc/my.cnf اضافه شود:

```
skip-networking
```

اگر به هر دلیلی دسترسی از راه دور به پایگاه داده نیاز بود از پروتکل ssh به شکل زیر استفاده کنید.

```
backuphost$ ssh mysqlserver /usr/local/mysql/bin/mysqldump -A > backup
```

۲,۴ بهبود امنیت محلی :

تغییر بعدی غیر فعال کردن دستور LOAD DATA LOCAL INFILE است که از خواندن غیر مجاز فایل های محلی جلوگیری می‌کند به این منظور پارامتر زیر باید به بخش [mysqld] از فایل chroot/mysql/etc/my.cnf/ اضافه شود:

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

۳,۴ تغییر رمز عبور admin

یکی از مهمترین اقدامات در امن‌سازی mySQL تغییر رمز عبور admin است که به صورت پیش فرض خالی است.

بدین منظور باید mySQL را run کنید

```
chrootuid /chroot/mysql mysql /usr/local/mysql/libexec/mysqld &
```

و سپس از طریق زیر رمز عبور administrator را تغییر دهید:

```
/usr/local/mysql/bin/mysql -u root
mysql> SET PASSWORD FOR root@localhost=PASSWORD('new_password');
```

بهتر است که رمز عبور را از طریق commandline مثلا با دستور "mysqladmin password" تغییر ندهید زیرا ممکن است سایر کاربرانی که از سرور استفاده می‌کنند با استفاده از دستوراتی مانند "*ps aux*" و یا بازدید فایل‌های history, ~/.bash_history./~ بتوانند به این رمز عبور دسترسی پیدا کنند.

۴,۴ پاک کردن پایگاه داده های پیش فرض:

در این مرحله باید پایگاه داده نمونه (test) و تمام حساب‌های کاربری آن بجز حساب کاربری local root را پاک کنید:

```
mysql> drop database test;  mysql> use mysql;  mysql> delete from db;  mysql>
 delete from user where not (host="localhost" and user="root");  mysql> flush
                                                                  privileges;
```

این مانع برقراری ارتباطات بدون نام پایگاه داده می‌شود.

۵,۴ تغییر نام admin

همچنین توصیه می‌شود نام پیش‌فرض حساب کاربری administrator (یا root) را به عبارتی که حدس آن مشکل باشد، تغییر دهید. چنین تغییراتی باعث می‌شود تا احتمال بروز حملاتی مانند brute force یا حمله dictionary به رمز عبور administrator کاهش یابد. در این مورد مهاجم نه تنها باید رمز عبور را حدس بزند بلکه ابتدا باید نام حساب کاربری administrator را نیز حدس بزند.

```
mysql> update user set user="mydbadmin" where user="root";
mysql> flush privileges;
```

۶,۴ پاک کردن تاریخچه:

بالاخره نیز باید محتویات تاریخچه mySQL در فایل (mysql_history./~) را پاک کنید زیرا تمام دستورات اجرا شده در در آن ذخیره شده‌اند(به خصوص رمز عبور).

```
cat /dev/null > ~/.mysql_history
```

۷,۴ فعال ساختن logging

برای فعال‌سازی بخش logging به دو روش می‌توان عمل نمود: اول اینکه با استفاده از فایل my.conf در مسیر /etc/ در قسمت [mysqld] بخش log_error را فعال نموده و با نام فایل مورد نظر مقداردهی کنید. راه دیگر این است که وارد محیط پایگاه داده شده و با استفاده از دستور زیر متغیر log_error را بیابید.

```
mysql> show variables like 'log_error';
```

و با دستور update مقدار آن را با نام فایل مورد نظر مقداردهی کنید.

باید توجه شود که log نباید در پارتیشن سیستمی ذخیره شود یعنی در لینوکس نباید در root ذخیره گردد. به این منظور با استفاده از دستور زیر متغیر log_bin را بیابید.

```
mysql> show variables like 'log_bin';
```

و مقدار آن را به on تغییر دهید. سپس در فایل my.conf در قسمت [mysqld] بخش log_bin را فعال نموده و با مسیر مورد نظر مقداردهی کنید. توجه نمایید که log روی همان پارتیشنی که MySQL قرار دارد ذخیره نشود. در ضمن باید کنترل کنید تا اجازه دسترسی و نوشتن در فایل Log فقط به عهده MySQL و root باشد.

۵.ارتباط میان PHP و MySQL

در خصوص ارتباط MySQL با PHP باید به این مسئله اشاره کرد که وقتی یکی از آن ها در محیط chroot اجرا می شود، مشکل ارتباطی بینشان رخ می‌دهد. زیرا PHP به وسیله سوکت */tmp/mysql.sock* با MySQL ارتباط برقرار

می‌کند. برای حل مشکل باید هر بار که MySQL را اجرا می‌کنیم لینکی به PHP chrooted environment ایجاد نمایید:

```
ln /chroot/mysql/tmp/mysql.sock /chroot/httpd/tmp/
```

توجه شود که سوکت /chroot/mysql/tmp/mysql.sock و دایرکتوری /chroot/httpd/tmp بایداز نظر فیزیکی روی یک فایل سیستم قرار داشته باشند در غیر این صورت نمی‌توانند با یکدیگر ارتباط برقرار کنند (لینک کار نمی‌کند).

گام های نهایی : ۵,۱

در این مرحله می‌توانید تمام پایگاه‌های داده و حساب‌های کاربری که به وسیله برنامه‌های کاربردی خاص PHP مورد استفاده قرار می‌گیرند، را ایجاد کنید. تاکید می‌شود که این حساب‌های کاربری باید تنها حق دسترسی به پایگاه داده-ای را دارند که توسط برنامه‌های کاربردی PHP مورد استفاده قرار می‌گیرد و نباید حق دسترسی به پایگاه داده mySQL را با هر سیستمی داشته باشند. همین طور نباید امتیاز دسترسی administrator داشته باشند (همین طور برای FILE, GRANT, ALTER, SHOW DATABASE, RELOAD, SHUTDOWN, PROCESS SUPER و...).

در نهایت باید یک shell script برای اجرای MySQL در زمان روشن شدن سیستم عامل ایجاد کنیم. یک نمونه در زیر آورده شده است:

```
#!/bin/sh CHROOT_MYSQL=/chroot/mysql CHROOT_PHP=/chroot/httpd
SOCKET=/tmp/mysql.sock MYSQLD=/usr/local/mysql/libexec/mysqld
PIDFILE=/usr/local/mysql/var/`hostname`.pid CHROOTUID=/usr/local/sbin/chrootuid
echo -n " mysql" case "$1" in start) rm -rf ${CHROOT_PHP}/${SOCKET}
nohup ${CHROOTUID} ${CHROOT_MYSQL} mysql ${MYSQLD} >/dev/null 2>&1 & sleep
5 && ln ${CHROOT_MYSQL}/${SOCKET} ${CHROOT_PHP}/${SOCKET} ;; stop)
kill `cat ${CHROOT_MYSQL}/${PIDFILE}` rm -rf ${CHROOT_MYSQL}/${SOCKET}
;; *) echo "" echo "Usage: `basename $0` {start|stop}" >&2
exit 64 ;; esac exit 0
```

اسکریپت بالا را باید در مسیر */usr/local/etc/rc.d/* و با نام *mysql.sh*. قرار گیرد.

۶.۶. جمع بندی:

اعمال شیوه‌های مطرح شده به ما این امکان را می‌دهد تا به طور قابل توجهی امنیت MySQL را ارتقا دهیم. با اجرای پایگاه داده در محیط chroot و غیر فعال کردن حالت گوش دادن روی پورت tcp ۳۳۰۶ و اعمال رمز عبور قوی برای کاربران می‌توانیم پایگاه داده را در برابر بسیاری از حملات حفظ کنیم. هر چند هیچ شیوه‌ای هنوز نمی‌تواند امنیت کامل را موجب شود اما اعمال موارد مذکور می‌تواند احتمال حمله را از سوی کاربرانی که صفحه وب یا وب سرور را می‌بینند، بسیار محدود نماید.

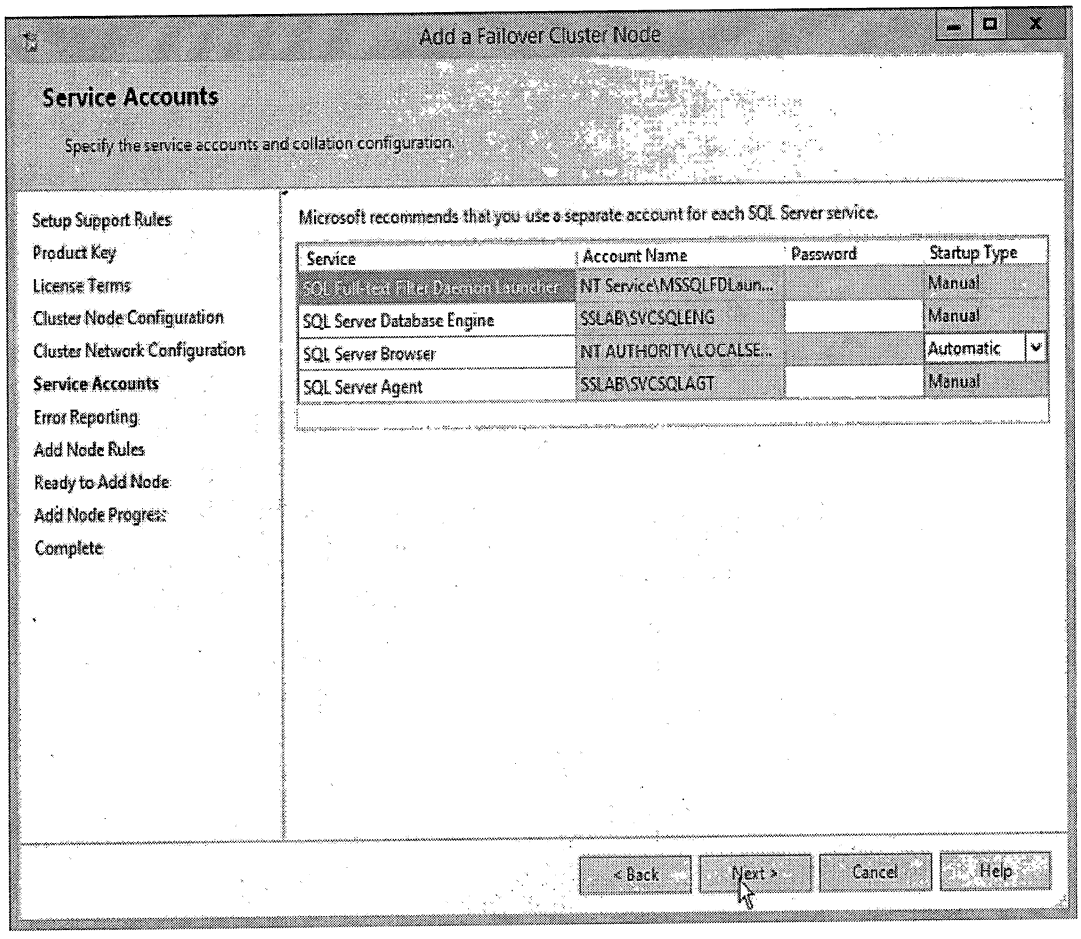
پیوست ۲ : پیاده‌سازی نکات امنیتی در Microsoft SQL

توجه شود که تنظیمات مورد نظر بر روی نسخه SQL Server Enterprise 2012 انجام شده است.

- ✓ با توجه به نیازهای سرور و آشنایی کامل با سیستم‌عامل و سرویس‌های در حال اجرا بر روی آن، باید تمام سرویس‌های غیرضروری حذف شوند. به عنوان مثال یک سرور پایگاه داده نیازی به سرویس‌های License Logging، Plug and Play، Remote Access Autodial Manager، Remote Access Connection Manager، Remote Access Server، Telephony، Server Message Block (SMB), و ... نخواهد داشت.
- ✓ برای غیرفعال کردن سرویس‌های ذکر شده باید در ویندوز در قسمت run عبارت view services را تایپ کرد تا وارد کنسول مدیریت سرویس‌ها شویم. سپس باید سرویس‌های زیر را غیرفعال کرد.
  - Distributed Transaction Coordinator (MS DTC)
  - Microsoft search
  - SQLServerAgent

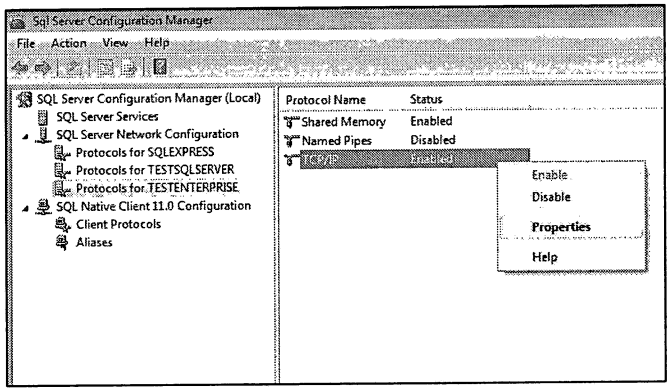
- MSSQLServerADHelper

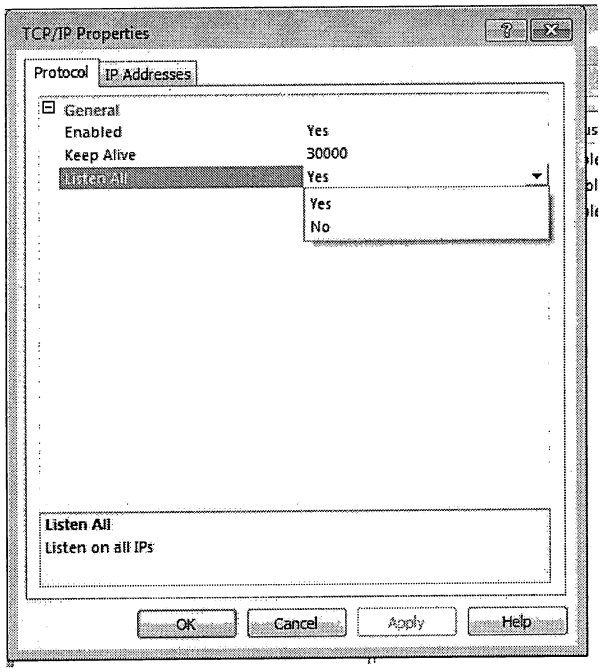
✓ هر پایگاه داده تعدادی سرویس در حال اجرا بر روی سیستم دارد. از این سرویس‌ها کاربرانی استفاده می‌کنند که ممکن است سطح دسترسی بیش از حد نیاز، به سیستم داشته‌باشند. باید توجه داشت که این سرویس‌ها تحت اجرای کاربرانی باشند که دسترسی کامل (admin) به سیستم **نداشته** باشند. در نرم‌افزار SQL Server در حین نصب نام کاربر مورد نظر پرسیده می‌شود. تصویر زیر مرحله موردنظر در نصب نرم افزار را نشان می‌دهد. همان جا باید یک کاربر با دسترسی محدود ایجاد کنیم (کاربر جدید در سیستم عامل ایجاد کنیم). بهتر است برای هر سرویس یک کاربر جداگانه با محدودیت دسترسی درنظر گرفته شود. نکته مهم دیگر در این مبحث استفاده از رمز عبور قوی برای کاربران موردنظر می‌باشد.



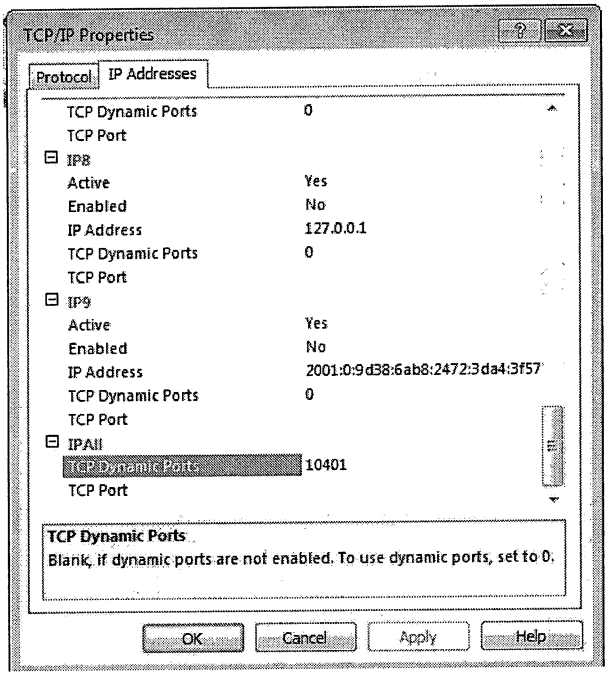
✓ یکی از نکات امنیتی برای محافظت از پایگاه داده، غیرفعال کردن پروتکل‌های بدون استفاده می‌باشد. بدین منظور بعد از نصب SQL server در بخش run از ویندوز کنسول SQL server configuration manager را باز می‌کنیم.

سپس به بخش SQL server network configuration رفته و طبق تصویر بر روی تنظیمات پایگاه داده‌ای که نصب کرده‌ایم و properties می‌گیریم.





سپس در نوار پروتکل، خصیصه listen all را بر روی No می‌گذاریم. سپس به نوار IP Addresses می‌رویم. سپس IP1 و IP2 را غیرفعال می‌کنیم ("enabled" را بر روی "No" قرار می‌دهیم). همچنین port دینامیکی به آن اختصاص نمی‌دهیم. باید مقدار خصیصه "TCP Dynamic port" را پاک کنیم و یک پورت دلخواه برای خصیصه "TCP port" در نظر بگیریم. (به طور مثال پورت ۵۴۵۴ برای آن درنظر می‌گیریم).



✓ برای مقاوم‌سازی پروتکل TCP/IP و جلوگیری از حملات به سیستم باید تنظیمات آن را بهینه کرد. بدین منظور باید تنظیمات موردنظر در رجیستری ویندوز به صورت زیر تغییر داد.

به آدرس `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TcpIp\Parameters` می‌رویم. و برای کلیدهای موردنظر مقادیر بیان شده را قرار می‌دهیم. (در صورتی که این کلیدها موجود نبودند باید آنها را ایجاد کرد).

Value Name	Value (REG_DWORD)
SynAttackProtect	2
TcpMaxPortsExhausted	1
TcpMaxHalfOpen	500
TcpMaxHalfOpenRetried	400
TcpMaxConnectResponseRetransmissions	2
TcpMaxDataRetransmissions	2
EnablePMTUDiscovery	0
KeepAliveTime	300000 (5 minutes)
NoNameReleaseOnDemand	1

- ✓ داخل برنامه SQL server management می‌شویم. بر روی نمونه پایگاه داده راست کلیک می‌کنیم. بر روی facet کلیک می‌کنیم. در پنجره باز شده facet را بر روی Server Configuration انتخاب می‌کنیم.

سپس می‌توان قسمت‌هایی از پایگاه داده را حذف و غیرفعال کرد. در این بخش ما خصیصه XPCmdShelEnabled را غیرفعال می‌کنیم.

- ✓ داخل برنامه SQL server management می‌شویم. در object explorer به بخش زیر می‌رویم.

```
SQLinstance->Security->Logins
```

سپس می‌توان بر روی کاربر guest کلیک کرده و آن را حذف کرد. همچنین می‌توان تنظیمات آنرا محدودتر کرد و roleهایی که برای آن تعیین شده است را محدود نمود.

- ✓ همانند مورد قبل به آدرس مورد نظر می‌رویم. سپس بر روی کاربر SA دبل کلیک می‌کنیم. رمز عبور را برای کاربر موردنظر تغییر می‌دهیم.
- ✓ با تایپ عبارت Local Security Policy در start وارد کنسول مورد نظر شوید سپس وارد پوشه‌های زیر شوید :

```
Account Policies > Password Policies
```

سپس Policy مورد نظر که Password must meet complexity requirements می‌باشد را باز کرده آن را enabled کنید.

- ✓ بر روی my computer راست کلیک کرده و properties گیرید. به بخش remote روید. سپس تمام تیک‌های آنرا بردارید.
- ✓ با تایپ عبارت secpol.msc در start وارد کنسول مورد نظر شوید و سپس وارد مسیر زیر شوید :

```
Local Policies->Security Options
```

Policy مورد نظر که Network access: Let Everyone permissions apply to anonymous users می‌باشد را باز کرده و Disabled می‌کنیم.

Policy مورد نظر که Network access: Named pipes that can be accessed anonymously می‌باشد را باز

کرده و Disabled می‌کنیم.

Policy مورد نظر دیگر Network access: Restrict anonymous access to Named Pipes and Shares را نیز Enabled می‌کنیم.

Policy مورد نظر که Network access: Shares that can be accessed anonymously ... می‌باشد را باز کرده و اگر چیزی داخل کادر نوشته شده باشد را پاک می‌کنیم.

همچنین Policy مورد نظر که Network access: Allow anonymous SID/Name translation می‌باشد را باز کرده آنرا disabled کنید.

همچنین Policy مورد نظر که Network access: Do not allow anonymous enumeration of SAM accounts می‌باشد را باز کرده آنرا Enabled کنید.

- ✓ همانطور که بیان شد وارد برنامه SQL Server می‌شویم. به بخش security و سپس به بخش logins می‌رویم. بر روی همه کاربران دبل کلیک می‌کنیم و تنظیمات مورد نیاز و دسترسی هر کاربر به پایگاه داده را مشخص می‌کنیم.
- ✓ با استفاده از قابلیت ایزوله‌سازی دامین و سرور شما قادر خواهید بود که فقط به کامپیوترها و ارتباط‌هایی اجازه برقراری ارتباط با سرورهای خود را بدهید که عضو دامین شما هستند و ارتباط آنها امن و رمزنگاری شده است. کامپیوترهایی که عضو دامین هستند صرفا می‌توانند با سرور ایزوله شده ارتباط برقرار کنند و سایر کامپیوترهایی که عضو نیستند فقط تحت شرایط خاصی می‌توانند با سرور مورد نظر ارتباط برقرار کنند. برای مثال می‌توان با استفاده از یک Policy به یک سرور SQL دستور داد که صرفا با کامپیوترهایی ارتباط برقرار کند که عضو گروه خاصی هستند یا اینکه دارای Certificate خاصی باشند. پروتکل IPsec مخصوص برقراری چنین ویژگی‌ای در سیستم‌عامل می‌باشد. این پروتکل در لایه اینترنت کار می‌کند. IPsec می‌تواند از جریان‌های داده مابین میزبان‌ها (میزبان به میزبان) ، مابین گذرگاه‌های امنیتی (شبکه به شبکه) یا مابین یک گذرگاه امنیتی به یک میزبان، پشتیبانی کند. همچنین می‌توان از IPsec به منظور رمزگذاری داده‌ها در شبکه استفاده کرد تا برای سوء استفاده کنندگان قابل دسترسی نبوده و در طول مسیر، امکان استفاده غیر مجاز از آنها وجود نداشته باشد.

باید تمامی ارتباطات را با استفاده از پروتکل مورد نظر امن کنیم. برای این منظور ابتدا باید تمامی پورت‌ها و آدرس‌های IP مورد

امن‌سازی پایگاه داده

نیاز سیستم را بشناسیم. سپس با استفاده از IPSec دسترسی را محدود و رمزشده کنیم. نحوه استفاده از این پروتکل با توجه به نیاز سیستم خواهد بود. برای مطالعه بیشتر بر روی این موضوع و آموزش نحوه پیاده‌سازی آن می‌توان به آدرس‌های اینترنتی زیر رجوع کنید.

http://msdn.microsoft.com/en-us/library/ff649249.aspx

http://msdn.microsoft.com/en-us/library/ff648481.aspx

- ✓ معمولا DBMSها ویژگی‌هایی را در اختیار می‌گذارند که می‌توان برای حسابرسی فعالیت‌ها و تغییراتی که روی سیستم روی می‌دهد استفاده کرد. حسابرسی یک فعالیت در بانک اطلاعاتی، مستلزم tracking و logging کردن رویدادهایی است که در سیستم روی می‌دهند.
- ✓ یکی از مهمترین نگرانی‌های یک پایگاه داده افشای اطلاعات است. یکی از راه‌حل‌های مقابله با این موضوع رمزنگاری اطلاعات سیستم می‌باشد. به صورتی که اطلاعات به صورت رمزشده بر روی سیستم ذخیره شود.

TDE یک سطح از رمزنگاری بانک اطلاعاتی است که هم فایل داده و هم فایل لاگ را رمزنگاری می‌کند. در کل تمامی اطلاعات ذخیره شده بر روی دیسک رمز می‌شوند. فایل‌های *.mdf، *.ndf، *.ldf و TempDB همگی رمز می‌شوند. در این روش از database encryption key (DEK) استفاده می‌شود. DEK یک کلید از نوع متقارن است که خودش توسط یک certificate محافظت می‌شود که این certificate در بانک اطلاعاتی master ذخیره شده است.

نحوه پیاده سازی TDE به صورت زیر می‌باشد.

۴ مرحله برای setup کردن آن وجود دارد :

۱- ایجاد یک Master Key

۲- ایجاد یا دریافت یک Protected Certificate توسط master key

۳- ایجاد یک database Key و protect کردن آن توسط Certificate

۴- Set کردن یک دیتابیس به منظور استفاده از Encryption

مرحله اول: ایجادMaster Key

این کلید در بانک اطلاعاتی Master ایجاد می‌شود. فقط قبل از ایجاد باید مطمئن شوید که قبلا ایجاد نشده باشد.

```
USE master;
SELECT * FROM sys.symmetric_keys WHERE name LIKE '%MS_DatabaseMasterKey% '
```

اگر نتیجه این query رکوردی بود، به این معنی است که شما قبلا یک master key ایجاد کرده‌اید.

اگر master key نداشته باشد، به صورت زیر آن را ایجاد می‌کنیم:

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = '12345';
```

مرحله دوم: ایجاد یک certificate توسط Master Key است

قبل از ایجاد آن چک می‌کنیم که با نام مورد نظر ما قبلا certificate ایجاد شده یا نه؟

```
SELECT * FROM sys.certificates where [name] = 'Your Certificate Name'
```

اکنون برای ایجاد آن اقدام می‌کنیم.

```
CREATE CERTIFICATE MyTDECert WITH SUBJECT = 'My TDE Certificate'
```

اگر select بالا را اجرا کنیم، نتیجه رکوردی است که ستون نام آن MyTDECert می باشد.

مرحله سوم: ایجاد یک Encryption Key و Protect کردن آن توسط certificate می باشد.

این مرحله حساس است و حتما باید مراقب باشید.

```
Use AdventureWorks
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_128
ENCRYPTION BY SERVER CERTIFICATE MyTDECert
```

بعد از اجرای این دستور چنین پیغامی را دریافت می‌کنید که مهم است و باید به دقت آنرا بخوانید :

Warning: The certificate used for encrypting the database encryption key has not been backed

up. You should immediately back up the certificate and the private key associated with the certificate. If the certificate ever becomes unavailable or if you must restore or attach the database on another server, you must have backups of both the certificate and the private key or you will not be able to open the database

مرحله چهارم :فعال کردن **Encryption** در بانک اطلاعاتی است.

```
ALTER DATABASE AdventureWorks
SET ENCRYPTION ON
```

در صورتی که بانک شما سنگین باشد این مرحله قدری طول خواهد کشید.

بعد از آن شما با دستور زیر، پی‌خواهید برد که آیا بانک اطلاعاتی شما رمزنگاری شده است یا نه.

```
SELECT DB_NAME(database_id) AS DatabaseName, * FROM sys.dm_database_encryption_keys
```

نتیجه برای یک db دو رکورد است. یکی خود آن و دیگری tempdb.

البته در این روش باز هم بعضی از اطلاعات رمز نمی‌شوند. تمامی اطلاعات موجود در حافظه اصلی رمز نمی‌شوند. همچنین داده-هایی که در حال انتقال هستند نیز بدون رمز منتقل می‌شوند.

- ✓ معمولا بر روی DBMS چندین پایگاه داده به عنوان نمونه آموزشی وجود دارد. به عنوان مثال در پایگاه داده SQL Server دو پایگاه داده Pubs و Northwind وجود دارد. بهتر است این پایگاه داده‌ها از روی DBMS حذف شوند.

یکی از مهترین راه‌کارها برای مقاوم‌سازی پایگاه داده، حذف یا غیرفعال کردن ویژگی‌هایی که از آنها استفاده نمی‌شود، است. اینکه ما از ویژگی خاصی استفاده نکنیم به این معنی نیست که هکرها و نفوذگرها از نقاط ضعف آن ویژگی برای نفوذ به سیستم استفاده نمی‌کنند.

لذا سیاست‌های قابل‌اعمال باید به نحوی باشند که :

الف) دسترسی کاربران به اطلاعات بر اساس هویت کاربر و قوانین مالکیتی از پیش تعریف‌شده (توسط مدیر امنیت سیستم) باشد.

ب) دسترسی کاربران به اطلاعات بر اساس رده‌بندی امنیتی کاربران و داده‌های موجود در سیستم، مدیریت شود.

ج) رده امنیتی کاربر، میزان اطمینان سیستم به وی و رده امنیتی داده، میزان حساسیت آن و ضررهای ناشی از فاش شدن داده بررسی شوند.

د) هر وظیفه شغلی یا مسئولیت در سیستم به عنوان یک نقش تعریف‌شده و مجوزهای لازم برای ایفای نقش، به آن تخصیص یابد.

هـ) کاربران با عضویت در هر نقش می‌توانند به منابع سیستم دسترسی پیدا کنند که این نقش رابط کاربر و مجوزها می‌باشد.