ثغرات الحقن بالتغطية (Blind Sql Injection)

وكيفية الوقاية منها وإغلاقها

الباحث:شادي شماس

ملخص البحث

مع التطور الكبير في التقنيات المستخدمة في الهجمات أصبحت التهديدات الأمنية على شبكة الإنترنت ومواقعها أحد أكبر التحديات في الوقت الحالي، ويمكن القول إن إحدى أسهل وأخطر هذه الهجمات هي ثغرات الحقن بلغة الإستعلام SQL (Sql Injuction) التي باتت تشكل تهديداً كبيراً وخطيراً لأي موقع أو تطبيق ويب والذي تكون إحدى مكوناته هي قاعدة بيانات، فهذه الثغرات تمكن المهاجم أو المخترق (Hackers) من الحصول على المعلومات أو البيانات الحساسة والهامة والتي تكون ذات قيمة عالية وكبيرة من قواعد البيانات، وتتميز أساليب الهجوم من خلال هذه الثغرات بأنها سهلة التعلم بالإضافة إلى أن الأضرار التي يمكن أن تسببها أو تحدثها هذه الثغرات تتراوح من أضرار بسيطة ومعقولة إلى أضرار تصيب كامل النظام المعلوماتي، ويجب التنويه إلى أن هناك العديد من التطبيقات والمواقع الإلكترونية على الإنترنت تكون مصابة بهذه الثغرات.

في هذا البحث أجريت دراسة يمكن وصفها بأنها دراسة مفصلة عن إحدى أهم وأبسط وأخطر الثغرات الأمنية في المواقع وهي ثغرة حقن لغة الإستعلام Sql والتي تعتبر من أكثر الثغرات إنتشاراً على الإطلاق والتي تستخدم لإختراق قاعدة البيانات وسرقة محتوياتها من المعلومات أو تخريبها، وقمت بذكر بعض الطرق والأساليب التي يمكن إتباعها لحماية قواعد البيانات من هجمات هذه الثغرات والتي تساعد في بعض الأحيان في إغلاق هذه الثغرات الأمنية وسدّها.

كلمات مفتاحية:

Hacking, sql injection ,Data Base ,Exploits, Vulnerabilities, blind sql injection.

١ -مقدمة البحث:

لقد إزداد إعتمادنا في الآونة الأخيرة على تطبيقات الويب بشكل ملحوظ فأصبحت جزءاً من أنشطة حياتنا اليومية، فعندما نقوم بممارسة هذه الأنشطة والأعمال عن طريق مواقع الويب على الإنترنت فإننا نعتقد في معظم الأحيان بأن هذه التطبيقات آمنة وموثوقة، وكلما كان هناك زيادة وتسارع في مستوى التطور التقني وفي توافر هذه التطبيقات، كلما كان هناك زيادة في عدد ومستوى الهجمات التي تستهدف هذه التطبيقات، وأحد أخطر أنواع هذه الهجمات هي ثغرات حقن لغة الإستعلام Sql والتي هي نوع من الهجمات التي تستغل ضعف ما أو ثغرة في البرنامج، فهي تعتمد على إضافة كود برمجي مكتوب بلغة SQL إلى المتغيرات الممررة للنظام المعلوماتي الذي يتم التعامل معه بحيث يتم تنفيذ هذا الكود أو الشفرة مع الكود الأساسي الموجود في النظام وبهذا يتمكن المخترق من الوصول الغير مصرح له إلى نظام قواعد البيانات أو القيام بعملية إسترجاع معلومات حساسة مخزنة ضمن قواعد البيانات.

٢ - هدف البحث وطريقته:

دراسة نوع من أنواع الثغرات الأمنية في المواقع وتطبيقات الويب بشيء من التفصيل وذلك من خلال توضيح مفهومي الحقن بلغة الإستعلام (Sql Injection) والحقن الساتر بلغة الإستعلام (Blind Sql Injection) والتمييز بينهما فهذا النوع من الثغرات يعتبر من أبسط أنواع العبث بقواعد البيانات و يستخدم عادةً من قبل المخترقين في الهجوم على الأنظمة المعلوماتية التي تتعامل مع الويب ولكنه في الوقت نفسه يعتبر من أشد وأخطر الأنواع التي تؤذي نظام قواعد البيانات ككل وتضر به، بالإضافة إلى عرض أهم السياسات والإجراءات التي يجب إتخاذها والدوال التي يجب العمل بها لإغلاق أو سدّ أو ترقيع هذا النوع من الثغرات والوقاية منه قدر الإمكان وذلك بغية الحفاظ على سلامة قواعد البيانات وبالتالي سلامة البيانات أو المعلومات وذلك بأسلوب عملي مزود بالكود البرمجي الذي يوضح كيفية تطبيق هذه السياسات والدوال لصيانة

قواعد البيانات والحفاظ عليها، وذلك لأنه في الآونة الأخيرة حصل تطور كبير في مجال قواعد البيانات، و إزدادت التطبيقات التي تعتمد على قواعد البيانات، فأصبحنا نلاحظ بأن معظم تطبيقات الويب (مواقع الويب) تعتمد بشكل أساسي على قواعد البيانات وأصبحت هذه القواعد جزء لا يتجزأ من النظام المعلوماتي الذي يتم بناؤه وتطبيقه ضمن المنظمات، كالبنوك والشركات التجارية وشركات الطيران وغيرها الكثير، وهذا بدوره أدى إلى إزدياد عدد المخربين الذين يهتمون بإيجاد الآليات المناسبة للهجوم على قواعد البيانات وذلك بهدف سرقة البيانات أو المعلومات المخزنة ضمن جداول قواعد البيانات أو تخريبها أو حتى تدميرها بالكامل، لذلك كان من الواجب علينا الإهتمام بحماية البيانات ومنع أي شكل من أشكال الوصول إليها و العبث بها.

٣-المناقشة

٣-١-مفهوم حقن لغة الإستعلام SQL (Sql Injection):

هو إستعلام ضار يمكن دمجه مع إستعلام صحيح يقوم بإحضار البيانات أو المعلومات من قاعدة البيانات، وبالتالي يكون بمقدور المهاجم أو المخترق (Hacker) إضافة شفرة من لغة الإستعلام SQL إلى مدخلات النظام بحيث يتم تنفيذها بعد تنفيذ الشفرة الأساسية على أجهزة المخدم أو السيرفر (Server) أو معها، وبالتالي السماح للمخترق بالدخول الغير مرخص للنظام أو حدوث تشويه في بيانات النظام[4].

٣-٢- مفهوم الحقن الساتر بلغة الإستعلام SQL (Blind Sql Injection):

الحقن الساتر (blind sql injection) هو طريقة تعتمد على تطبيع الإستعلام الموجود في التطبيق وذلك بإستخدام عبارات بوليانية تعيد إحدى القيمتين true أو false وأشهر تلك العبارات And 1=1, 1=0 - or 1=1, 1=0 بحيث تجعل الإستعلام ككل في صالح المخترق وينفذ ما يريده دون الحاجة في بعض الأحيان لكتابة إستعلام كامل[4] .

٣-٣-الفرق بين الحقن والحقن الساتر بلغة إستعلام SQL:

يمكن القول بأن هناك ثغرات تدعى Sql Injection أي الحقن بإستعلام sql، وثغرات تدعى Blind Sql Injection أي الحقن الساتر أو الحقن بالتغطية

بإستعلام sql والفرق بينهما هو أنه في ثغرات Sql Injection يعتمد المخترق على خطوتين أساسيتين الأولى هي تتبع رسائل الخطأ التي تظهر من سواقة القاعدة (driver) والثانية تطبيق إستعلام يعتمد على رسائل الخطأ تلك، فمثلاً إذا تم إستخدام عبارة UNION SELECT فالطريقة المشهورة هي كتابة تعداد أو nulls طالما أن هناك رسالة خطأ "Wrong Fetch Array" ويتم الوقوف حتى تختفي، ولكن مع ظهور حمايات مختلفة لهذه الثغرة وإنتشارها أضحت الخطوة الأولى نادرة الوجود، فكان لابد من إبتكار وسيلة تعطي النجاة لهذه الثغرة في عالم الإختراق فظهر ما يسمى بـ blind folded حيث لم يعد من الضروري الإعتماد على رسائل الخطأ في الإستغلال بإستخدام هذا الأسلوب المخيف[2][4] .

٣ - ٤ -إكتشاف ثغرات الحقن والحقن الساتر بلغة الإستعلام وطريقة إستثمارها [1][9][4][2]:

فكما هو معلوم فإن حقول الجداول في قواعد البيانات تنحصر في ثلاثة أنواع رئيسية هي القيم المحرفية والقيم الرقمية والتواريخ والأكثر إنتشاراً هما النوعين الأوليين، وفي لغة الإستعلامات يتم تمرير الأرقام للسيرفر (Sever) كما هي، بينما المحارف والتواريخ تمرر مع وجود علامات التنصيص (') وكمثال على ذلك لنأخذ الإستعلامين التاليين:

(1) SELECT * FROM Users WHERE UserID = 2
(2) SELECT * FROM Users WHERE UserName = 'University'

فبالنسبة للإستعلام الأول فإذا إفترضنا أنه موجود ضمن تطبيق ويب والرابط كان على الشكل التالي:

/my_site/admin.asp?UserID=2

يكون إختبار الثغرة بطريقتين الأولى إضافة علامة تنصيص (') أما الثانية فتكون بتحويل القيمة لحاصل جمع قيمتين مثل ١+١ كما في المثالين التاليين:

/my_site/index.asp?UserID=2'
/my_site/index.asp?UserID=1 + 1

في الحالة الأولى إذا ظهر خطأ فهذا يعني أن الثغرة موجودة، وذلك لأن هذا يشير على أن المخدم أو السيرفر (Server) قَبِل القيم كمدخلات للمتحول بدليل أنه أخذها ووجدها

خطأ وأظهر رسالته، أما في الحالة المعاكسة إذا كانت الثغرة مرقعة فإن المخدم لن يقبل القيم المدخلة، سواء كانت صحيحة أم خاطئة، أما في الحالة الثانية إذا لم يظهر الخطأ فمعنى هذا أن الثغرة موجودة، لأن هذا يدل على أن المخدم قَبِل القيم كمدخلات للمتحول بدليل أنه أخذها ووجدها صحيحة، أما بالنسبة للإستعلام الثاني:

```
SELECT * FROM Users WHERE UserName= 'University'
```

فيكون إختبار الثغرة بطريقتين أيضاً إما بإضافة علامة التنصيص (') أو تحويل القيمة لحاصل جمع قيمتين محرفيتين مثل 'U'+'niversity' والنتيجة في الحالتين هي كالتالي:

```
(1) SELECT * FROM Products WHERE UserName= 'U'niversity'
(2) SELECT * FROM Products WHERE UserName= 'U' +
'niversity'
```

إذاً يمكن القول بأن الخطوات الصحيحة لإستغلال الموقع بثغرة حقن الإستعلام عبارة عن ثلاث خطوات هي:

١. التأكد من كون الموقع قابل للحقن.
٢. معرفة نوع قاعدة البيانات في التطبيق.
٣. كتابة الإستغلال المناسب.

ننتقل الآن للحقن بالتغطية blind folded بشكل أوسع ولنتذكر أولاً أن فكرة الحقن تعتمد على حقن عبارات sql تكون جزء من جملة WHERE والتي تستخدم بكثرة مع عبارة SELECT في الإستعلامات داخل تطبيقات المواقع بشكل خاص، لذلك يمكن للمخترق أن يضيف ما يريد من العبارات البوليانية والتي تجعل من الجملة ما يريده المهاجم نفسه، وذلك لأن WHERE هي تعليمة نتيجتها إما TRUE أو FALSE وأشهر هذه العبارات 1=1 or و 1=1 and وعبارة or هي الأكثر إستخداماً خاصةً مع مربعات الإدخال لأنها يمكن أن تلغي الشرط السابق وتضع نفسها مكان الإستعلام الذي يسبقها، فبفرض أنه يوجد موقع إحدى صفحاته تقوم بعرض سجل أو عدة سجلات وفق شروط معينة، وأن هذا السجل هو سجل ضمن جدول اسمه books يتألف من الحقول التالية (رقم الكتاب book_id، عنوان الكتاب book_title، محتوى الكتاب book_content) وكان الإستعلام مكتوب في ملف وليكن اسمه مثلاً book.asp على الشكل التالي:

```
SELECT * FROM books WHERE book_id =40000
```

فالصفحة الواحدة تعرض الكتاب ذو الرقم ٤٠٠٠، فمثلاً لو قام المهاجم بكتابة or 1=1 كما في الشكل التالي:

```
/my_site/news.asp?book_id=40000 or 1=1
```

فإن الإستعلام سيصبح على الشكل التالي:

```
SELECT * FROM books WHERE book_id=40000 or 1=1
```

وكما نعلم أن ١=١ يعطي القيمة true دائماً وبالتالي النتيجة تكون إظهار جميع الكتب ضمن الجدول حتى الكتاب ذو الرقم ٤٠٠٠، وأحياناً قد يتعرض المهاجم لمشكلة وهي أن 1=1 or قد لا تعمل والسبب هو أن الإستعلام قد يكون معقد وذلك عندما تكون تعليمة WHERE تحوي على أكثر من عبارة شرطية، فمثلاً لو كان لدينا الرابط التالي على الإنترنت:

```
/my_site/index.asp?Col_ID=233
```

وكان الإستعلام في التطبيق على الشكل التالي:

```
SELECT * FROM Tbl WHERE Col_ID=233 AND Col_style
BETWEEN 'style1' AND 'style2' AND Col_show='true' ORDER
BY Col_ view ASC, Col_Col_id DESC
```

في هذا الإستعلام لو تم حقن 1=1 or سيصبح الإستعلام كالتالي:

```
SELECT * FROM Tbl WHERE Col_ID=233 or 1=1 AND
Col_style BETWEEN 'style1' and 'style2' AND Col_show='true'
ORDER BY Col_ view ASC, Col_Col_id DESC
```

فنلاحظ بأن الإستعلام هنا لن يعمل بشكل صحيح أو بعبارة أخرى لن ينفذ المطلوب منه، والحل هنا يكون بإستخدام علامة التعليق (--) والتي تجعل أي شيء بعد هذه الإشارة ليس له معنى وبالتالي لن يتم إرساله للمخدم كجزء من الإستعلام بل سيهمل وكأنه غير موجود وبالتالي يصبح الإستعلام على الشكل التالي:

```
SELECT * FROM Tbl WHERE Col_ID=233 or 1=1 -- AND
Col_style BETWEEN 'style1' and 'style2' AND Col_show='true'
ORDER BY Col_ view ASC, Col_Col_id DESC
```

ولنأخذ مثال آخر لندعم الفكرة بشكل جيد ولنفرض أنه يتم العمل على موقع ما وخصوصاً ضمن لوحة تحكم المدير (Administrator)، ولنفرض أيضاً أنه تم كتابة إشارة التنصيص (') ضمن مربع اسم المستخدم (User Name) أما مربع كلمة المرور

فقد تُرِكَ فارغاً، ثم تم التسجيل بالضغط على زر log in، فإنه في هذه الحالة سيتم إظهار الإستعلام وذلك نتيجة الخطأ الذي سوف يحصل بعد إدخال علامة التنصيص (') كما في الشكل التالي مثلاً:

```
1- SELECT * FROM thedatabaseuser.user
2- WHERE UserName = 'ADMIN' AND PassWord = 'alsoadmin'
```

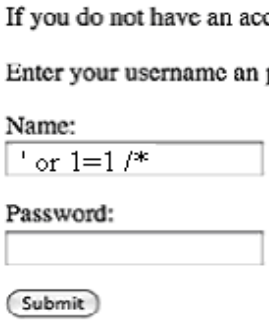
فالسطر الأول يطلب من القاعدة إختيار جميع حقول الجدول user أما السطر الثاني فيضع شرط WHERE لتحديد سجل واحد (record) فقط للشرط المعين ألا وهو أن يكون حقل اسم المستخدم يحوي القيمة ADMIN وحقل كلمة المرور يحوي القيمة alsoadmin وهاتين القيمتين كلاهما من النمط المحرفي، وبالتالي يستطيع المخترق أن يطبق على اللوحة عملية التغطية blind folded وذلك بطريقتين الأولى هو أن يقوم بكتابة الإستعلام السابق بالشكل التالي:

```
SELECT * FROM thedatabaseuser.user WHERE UserName = '' or
      '' = '' ' AND PassWord = '' or '' = ' '
```

فتظهر في الصفحة جميع سجلات الجدول، أو أن يستخدم علامة التعليق وبالتالي الإستعلام سيصبح بالشكل التالي:

```
SELECT * FROM thedatabaseuser.user WHERE UserName = '' or
1=1/*' AND PassWord = ''
```

والشكل التالي يوضح ذلك:



الشكل ١: تطبيق عملية التغطية من خلال إشارة التعليق [1]

٣ - ٥ - ترقيع ثغرات الحقن بلغة الإستعلام SQL وإغلاقها:

لاحظنا سابقاً أن هذه الثغرات تظهر في البرامج التي تستخدم قواعد البيانات في عملها مثل MYSQL-MSSQL-ORACLE وغيرها ولكنها تستخدم لغة واحدة لإستعلاماتها

وتسمى هذه اللغة SQL، وخطورتها تكمن في أنها تسمح بتنفيذ إستعلامات SQL غير التي موجودة في الملف المصاب أو إجراء عمليات التعديل على عبارات SQL الموجودة في الملف المصاب، وتوجد مجموعة من الحلول البرمجية يمكن إستخدامها لسد هذه الثغرة أو إغلاقها منها:

- تشفير جميع البيانات الهامة والحساسة المخزنة ضمن قاعدة البيانات، وذلك لأنه لو تمت سرقة هذه البيانات من قِبَل المخترقين فإنه لا يمكن الإستفادة منها، ولذلك لابد من إتباع بعض النصائح المستخدمة في عملية التشفير، كأن يتم حفظ كلمات المرور للمستخدمين بعد تشفيرها بأحد دوال تشفير كلمة المرور (hash) مثل الدالة ()Md5، أو إستخدام إحدى الدوال ()base64_decode أو ()base64_encode وأن يتم حفظ أرقام البطاقات الإئتمانية (Credit Cards) بعد تشفيرها ومنع الكتابة عليها، وذلك بإستخدام طرق التشفير المختلفة [4][3][8].
- يجب القيام بعملية تحديث النسخة الإحتياطية لقاعدة البيانات بشكل مستمر وذلك لتفادي سقوط النظام بالكامل فيما لو تم مسح قاعدة البيانات الأصلية أو تخريبها [4][2].
- يجب التأكد من مدخلات المستخدم، ويجب أن تتم عملية التأكد في المخدم (Server) وليس على جهاز المستخدم وذلك بإستخدام لغات البرمجة المناسبة كلغة جافا سكريبت مثلاً (JavaScript)، لأن عملية التحقق مهمة للغاية، ولابد من تنفيذها في مكان آمن، وذلك لأن جهاز المستخدم يمكن تعطيله من قبل المستخدم [10][4][3].
- عدم السماح للمستخدمين بإستخدام الكلمات الدليلية (key words) المستخدمة في أوامر SQL، إذ أن معظم الكلمات الدليلية لا ترد في اسماء المستخدمين و لا يسمح بها في كلمات المرور و لكن البعض منها ممكن أن يرد كـon – in، لذلك يجب التحقق من أن هذه المدخلات قد ورد فيها أي من الكلمات الغير مسموح بإستخدامها أم لا [8].
- يجب العمل على فصل المدخلات و وضعها في معاملات (Parameterized Input)، وهذه الطريقة تعتبر جيدة لمعرفة فيما إذا تم إضافة أشياء أخرى لحقل معين، وهي أيضاً تسهل عملية تعامل النظام البرمجي مع نظام قاعدة البيانات[8][4] .

▪ إستخدام بعض برامج الكشف عن الثغرات الخاصة بحقن لغة الإستعلام، والتي تعتبر من الأدوات الهامة التي تساعد مدير الموقع على تقوية نظام المعلومات الإداري للشركة، و معرفة النقاط التي يمكن أن تُستَغَل لعملية التخريب، ومن هذه الأدوات لدينا [4]:

الأداة	شرح مختصر لها
Scrawir	هي إحدى الأدوات المجانية التي تتيح معرفة الثغرات الخاصة بالحقن في نظام المعلومات الإداري
Bako’s SQL injection scanner	هي إحدى الأدوات المشهورة التي يمكن من خلالها فحص النظام أو أي نظام يتم التعامل معه و ذلك بشكل آلي و له خواص إضافية كثيرة
Paros proxy	هي إحدىالأدوات التي تفحص النظام البرمجي للكشف عن الثغرات الشائعة
Acunetix	هي إحدى أشهر الأدوات التي تقوم بفحص النظام للبحث عن الثغرات الخاصة بحقن لغة الاستعلام و تقدم تقريراً عن أي مخاطر محتملة يمكن إستغلالها لتخريب النظام
Netsparker – Web Application Security Scanner	هو عبارة عن برنامج فحص أو ماسح لخوادم الويب يقوم بالبحث عن الثغرات الموجودة و إعداد تقرير بها.

▪ يجب إستخدام إتصال آمن بين النظام البرمجي ونظام قاعدة البيانات وذلك لأنه عندما يكون نظام ما يتعامل مع الشبكة العنكبوتية فإنه سيكون عرضةً لسرقة البيانات في حالة التراسل[4].

▪ حماية الحقول الخاصة بكلمات المرور و ذلك بجعل الكلمات المطبوعة في حقول كلمات المرور غير مرئية للأشخاص القريبين من شاشة الجهاز، وبالتالي فإن هذه الحقول يجب أن تكون من نوع (Input PassWord) و ليس (Text)، إضافةً إلى

الحذر من إستخدام خاصية التكميل التلقائي لكلمات المرور من طرف المتصفح [4][7].

- القيام بعملية مراقبة المعاملات وذلك لضمان أمان تطبيق الويب، لذلك فإنه ينبغي على المطور إتباع آلية المراقبة الكلية وأن تكون عالية الجودة لكل المعاملات المقدمة من قبل المستخدم أو المستخرجة من الـURL، وكذلك يجب تحليل كل المعاملات لتنقيتها من الحروف الزائدة والشوائب و إستعمالها بحذر في الأوامر المختلفة [3][7].
- إستخدام الدالة is_numeric() التي تختبر فيما إذا كان المتحول من نوع رقم أم لا والتي عادةً يتم إستخدامها قبل تمرير المتحولات إلى الإستعلامات التي يتم تنفيذها على قاعدة البيانات، أو إستخدام الدالة intval() التي تقوم بتجاهل أي شيء يتم وضعه بعد المتحول الممرر لهذه الدالة، وهذه الدالة يجب تمرير المتحول إليها قبل تنفيذ الإستعلام وذلك تماماً كالدالة التي تسبقها، وسنقوم بتوضيح ذلك من خلال ذكر مثال يتضمن الخطأ البرمجي للثغرة وكيفية ترقيعها من خلال إستخدام إحدى الدالتين المذكورتين سابقاً[4]:

```
1. <?
2. mysql_connect(localhost,root,"");
3. $d=mysql_select_db(my_books);
4. $id= $_GET['id'];
5. $result=mysql_query("select * from book where id='$id' ");
6. while($row=mysql_fetch_array($result))
7. {
8. echo $row[content];
9. }
10. ?>
```

كما نلاحظ فإن هذا الكود البسيط يقوم بتنفيذ الإستعلام الموجود في السطر الخامس وذلك بعد إجراء عملية الإتصال بقاعدة البيانات ويقوم بعرض بيانات الكتاب الذي يحمل الرقم المطابق تماماً للرقم المخزن ضمن المتحول id، حيث يتم الحصول على قيمة المتحول id من خلال الدالة GET، ونلاحظ أنه لا يوجد أي حماية مفروضة على هذا المتحول، لذلك يمكن من خلال إستخدام إحدى الدالتين السابقتين فرض

شروط على قيمة المتحول id المدخلة من قبل المستخدم ليصبح الترقيع على الشكل التالي:

```
$id=is_numeric($_GET['id']) ;
   $result=mysql_query("select * from book where id='$id' ");
```

أو

```
$id=intval($_GET['id']) ;
   $result=mysql_query("select * from book where id='$id' ");
```

- إستخدام سلسلة الهروب (\) وذلك عند الحاجة إلى إستخدام علامة الاقتباس (') ضمن عبارة SQL حتى لا نحصل على خطأ برمجي، فمثلاً إذا كان لدينا الكود البرمجي التالي SELECT * FROM table WHERE field='text'; والذي يقوم بإختيار البيانات التي تحتوي على الكلمة text في الحقل field، فإذا أراد المستخدم إختيار السجلات التي تحتوي على الكلمة text's ضمن الحقل field، وقام بكتابة جملة الإستعلام التالية [2]:

```
SELECT * FROM table WHERE field='text's';
```

فهنا سيكون الضعف وستقع الثغرة، وذلك لأنه ببساطة سوف يتم التدخل في الإستعلام من قبل المخترق ويستطيع في هذه الحالة القيام بعملية حقن الإستعلام بإستعلام آخر وذلك من أجل القيام بغرض معين وليكن حذف قاعدة البيانات مثلاً أو أخذ صلاحيات المدير أو ما شابه، وبالتالي يمكن تغيير صيغة الإستعلام من قبل المهاجم ليصبح بالشكل التالي:

```
SELECT * FROM table WHERE field='text' ; DELETE FROM
table;
```

وبمجرد تنفيذه سيتم حذف قاعدة البيانات، ولذلك بإستخدام سلسلة الهروب (\) يتم تجاوز عملية الوقوع بهذه الثغرة وبالتالي يصبح الإستعلام بالشكل التالي[2] :

```
SELECT * FROM table WHERE field='text\'s';
```

كما يمكن إستخدام الدالة ()addslashes التي تضيف علامة / قبل أي اقتباس (') أو (") والمثال التالي يوضح كيفية إستخدامها[4] :

```
<?php
mysql_query("insert into guestbook set message =
```

'".addslashes($_POST['message'])."'"
?>

- تحديد عدد الأحرف التي يقوم المستخدم بإدخالها ضمن مربعات النص الموجودة ضمن النموذج (form) والموجود بدوره في صفحة الموقع الإلكترونية بحيث يتناسب مع العدد المحدد ضمن الحقول المقابلة لمربعات النص هذه وذلك لأن عملية عدم التحقق من المدخلات ستؤدي إلى وقوع أخطاء وهذه الأخطاء تقع عندما يقوم المستخدم بإدخال قيم أكبر من اللازم أو الحد المسجل له، وهذا ما يدعى بثغرات تقزيم الأعمدة، حيث تتمثل ثغرات تقزيم الأعمدة في إدخال قيمة شبيهة بقيمة موجودة سابقاً لكن مع إختلاف جزئي يقوم به المخترق لإجبار التطبيق (موقع الويب) على أخذ القيمة وتخزينها في قاعدة البيانات، فعندما يتم إدخال قيمة طويلة لقاعدة البيانات يقوم الـMYSQL بتقزيمها حتى لو تخطت الحد المعلن لها وهنا يكمن الخطر، فبفرض أن شركة ما تملك تطبيق ويب حيث يمكن لمستخدمي هذا الموقع التسجيل في الموقع وبفرض أن اسم المدير معروف مثلاً وليكن Administrator وبفرض أنه لا يوجد حد لطول القيمة المدخلة، و أن عمود (حقل) القاعدة user محدد بـ٢٥ حرف كحد أعظمي، فإذا قام المخترق بمحاولة تسجيل عضو جديد باسم Administrator ستفشل المحاولة نظراً لوجود دالة محددة تقوم بعمل بحث عن العضو وتحديد هل يوجد اسم بذلك أم لا، ولتكن مثلاً تحمل الاسم التالي ()ExamAlreadyRegistred، فإذا إفترضنا أن المخترق قام بإدخال قيمة عدد أحرفها يتجاوز ٢٥ حرف ولتكن مثلاً 'Administrator……x' ستقوم الدالة السابقة بالبحث عن الاسم في قاعدة البيانات وطبعاً لن يتم العثور عليه وذلك لأن عدد الخانات تجاوز ٢٥ خانة وبالتالي سيقوم التطبيق (موقع الويب) بالقبول به في مرحلة أولى وإرساله إلى قاعدة البيانات، وعلى مستوى قاعدة البيانات سيقوم MYSQL بتقزيم الاسم نظراً لمحدودية العمود بـ٢٥ حرف وسيصبح بالشكل التالي 'Administrator …… حيث النقاط موضوعة هنا فقط لتوضيح المثال وليس لأنها تعني شيء، ونتيجة لدهاء المخترق أصبح موجود الآن مستخدمين لهم الاسم Administrator، وهنا يقع الإشكال، فعند عملية الدخول يقوم التطبيق بطلب كلمة المرور حسب اسم المستخدم ونظراً لأن المخترق حديثاً قام بتسجيل الاسم المقزم في قاعدة البيانات فنتيجة العبارة

SELECT ستقوم بإعطاء قيمته أولاً، وبالتالي يستطيع معرفة كلمة المرور التي قام المدير المقزم بالتسجيل بها وبالتالي يمكنه التحكم بالموقع لأنه يتمتع بصلاحيات المدير المسؤول (Administrator)[4] .

- إستخدام ما يسمى GreenSQL وهو مشروع مفتوح المصدر يمثل جدار ناري مخصص لقواعد البيانات لحمايتها من هجمات SQL INJECTION، حيث يعمل بين الموقع وقاعدة البيانات ويحدد ماهية البيانات المتاح الوصول إليها و ماهي البيانات الممنوع الوصول إليها أي أنه يعمل نوع من فلترة للإستعلامات التي تنفذ على قاعدة البيانات، والمشروع متوفر للتحميل على , CentOS , debian , Opensure fedora , ubuntu , FreeBSD ويتوفر على واجهة جميلة لإدارته مع إمكانيات عديدة في إعداده [4] .
- يجب على المبرمج أثناء إنشاء الجدول جعل حقول اسم المستخدم (User Name) وكلمة المرور (Pass Word) تأخذ الصفة NOT NULL في إستعلام Create حتى لا يستطيع المستخدم تجاوز هذه الحقول دون تعبئة قيمة فيها، وذلك على الشكل التالي مثلاً:

```
CREATE Table user ( Id int(10), User Name varchar(30) NOT
NULL, Pass Word varchar(30) NOT NULL PRIMARY KEY(id) )
```

وبالتالي هذا يجنب من إمكانية حدوث ثغرة SQL INJECTION، لأنه في حالة عدم ضبط هذه الخاصية سيستطيع المخترق التحكم في إستعلام SELECT وكتابته بالشكل التالي مثلاً:

```
SELECT * FROM thepornoplayer.user WHERE User Name = '' or
1=1 ' AND Pass Word = '';
```

وبالتالي فإنه يطلب من القاعدة إختيار جميع أعمدة الجدول user والذي فيه الحقل User Name فارغ أو أن يكون ناتج ١=١ صحيح true، وهنا كما نعلم أنه ستكون النتيجة true إذا كان أحد الشرطين محققين أو كليهما، ثم يأتي ليختبر فيما إذا كانت كلمة المرور فارغة، ونلاحظ هنا وجود and والتي تعيد true إذا كان كلا الشرطين محققين معاً [4].

▪ إستخدام علامتي التنصيص الثنائية (")، بدلاً من علامة التنصيص الفردية التي يدخلها المستخدم، لأن هذه العملية لها دور كبير بإفشال محاولة الإختراق بإستخدام حقن قواعد البيانات [4].

▪ تحديد صلاحية المستخدم، ويعني ذلك تجنب الدخول لقاعدة البيانات بالحساب الذي يعطي المستخدم كامل الصلاحيات للقيام بأي شيء وذلك حتى وإن نجح المهاجم في الدخول إلى قاعدة البيانات فإنه لا يملك الصلاحيات للقيام بأي شيء وهذا يساعد في تجنب إمكانية العبث بقاعدة البيانات[3].

▪ إستخدام رمز المساواة (=) بدلاً من كلمة LIKE، وذلك لأن كلمة LIKE عادةً ما تستخدم في جمل حقن قواعد البيانات، ويفضل إستخدامها في محركات البحث فقط [4].

▪ يجب الإنتباه لعدم كتابة الاسماء الواضحة والمتعارف عليها كتسمية العمود بالاسم، كلمة المرور، أرقام بطاقات الإئتمان وغيرها من التسميات، بل يجب تسميتها باسماء يصعب على المهاجم التنبؤ بها[4][14].

▪ الإبتعاد عن وضع أي معلومات تخص قاعدة البيانات في رسائل الخطأ، ويقصد بذلك رسائل الخطأ التي تظهر للمستخدم، فهذه المعلومات قد تفيد المهاجم في إختراقه لقاعدة البيانات، وكمثال على ذلك: عدد الأعمدة في قاعدة البيانات أو حجم قاعدة البيانات أو اسماء بعض الأعمدة والجداول[4] .

▪ الإهتمام بالحصول على آخر الإصدارات من البرمجيات التي تقوم بعمل مراجعة للأكواد البرمجية المستخدمة في مراجعة مدخلات المستخدم، فهذه البرمجيات تقوم بمراجعة صحة ودقة الأكواد البرمجية التي تم إنشائها وعملها مسبقاً في قاعدة البيانات لكي تقوم بمراجعة مدخلات المستخدم فكلما كانت هذه الأكواد البرمجية صحيحة ودقيقة كلما كانت مراجعتها للجمل التي يدخلها المستخدم أيضا دقيقة للغاية، وبالتالي تمنع الجمل التي قد تكون خطرة على قاعدة البيانات، ويوجد العديد من البرمجيات التي أنتجتها مؤخراً إحدى الشركات المتخصصة في مجال التكنولوجيا الرقمية و الكمبيوتر في مطلع هذا العام والتي يكمن دورها في تجربة المواقع الإلكترونية التي تستخدم لغة ASP

و ASP.Net لكي تتأكد من خلوها من الثغرات الأمنية التي تسمح بحقن قواعد البيانات، ومن هذه البرامج نذكر [15][16]:

١.(Scanning run time) أو الفحص أثناء التشغيل:

هذه الأداة تعمل على الفحص الدقيق للمواقع الإلكترونية التي تكون قيد التشغيل حيث تقوم بتجربتها وكشف أي ثغرة أمنية تسمح بحقن قواعد البيانات لكي يتسنى للشخص معرفة وجود هذه الثغرة وبالتالي التمكن من حلها.

٢. فحص ومراجعة الكود البرمجي:

إن أداة الحماية هذه والمكتوبة بلغة ASP، تقوم بالبحث عن الثغرات الأمنية التي تسمح بحقن قواعد البيانات وتقوم بإطلاع المبرمج عليها لكي يتمكن من التغلب عليها بإغلاقها.

- تدقيق و تنقيح الكلمات التي أدخلها المستخدم، فإذا كانت تتضمن كلمات معروفة ومستخدمة في قاعدة البيانات فإنه يجب عدم السماح بها، مثال على ذلك الكلمات , [12]:

DELETE, SELECT CREATE, UNION

- تعطيل خاصية الإستعلامات المتداخلة (Nested Queries) في نظام قواعد البيانات[4][11].
- في بعض الأحيان تكون البيانات المدخلة صحيحة لكنها تتضارب وتتعارض مع تنسيق جمل الإستعلام نفسها، وهذه مشكلة، ففي هذه الحالة من الأفضل عند الإتصال بالخادم إستخدام الدالة ()mysql_escape_string وذلك كما في الشكل التالي[6] :

```
$User Name=mysqli_real_escape_string($GET['User Name']);
mysaql_query("SELECT * FROM tbl_numbers WHERE User
Name=' ".$User Name." ' ");
```

- القيام بعملية تصفية للبيانات المدخلة من خلال الإستعلام، وذلك بإستخدام الدالة ()filter_sql كما في الشكل التالي[17] :

```
Function filter_sq ($input) ;
{$reg="(delete) | (update) | (insert) ";
```

```
Return (eregi_replace($reg,"",$input)); }
```

- إستخدام الدالتين mysql_escape_string، mysql_real_escape_string اللتين تساعدان في الحماية من ثغرات الحقن وثغرات الحقن الساتر وذلك كما في المثاليين التاليين [4][13]:

```
<?php
$item = "Zak's Laptop";
$escaped_item = mysql_escape_string($item);
printf("Escaped string: %s\n", $escaped_item);
?>
<?php
// Connect
$link = mysql_connect('mysql_host', 'mysql_user', 'mysql_Pass
Word')
OR die(mysql_error());
$query = sprintf("SELECT * FROM users WHERE user='%s'
AND Pass
Word='%s'",mysql_real_escape_string($user),mysql_real_escape_s
tring($Pass Word));
?>
```

- القيام بعملية تنقيح للمدخلات وإزالة بعض العلامات منها كالفاصلة المنقوطة والواصلة المزدوجة وغيرها من العلامات التي قد تسهل على المهاجم الوصول إلى قاعدة البيانات [2][5].
- يجب التأكد مما يدخله الزائر من قيم في عمليات البحث في الموقع أو تسجيل الدخول ومثل هذه الأشياء التي تحتاج إلى إدخال من قبل المستخدم، وذلك للمحافظة على صحة الإستعلامات في الموقع، فمثلاً إذا كان في الموقع محرك بحث بسيط مكون من نموذج (form) أو فورم وملف php وقام الزائر بوضع مسافة من خلال المسطرة (space) وقام بالبحث فستظهر له جميع محتويات المكان الذي يبحث فيه، مما يؤثر على المخدم (Server)، والحل يكمن في التأكد من أن المتغير الخاص بعملية البحث لا يحتوي على فراغ ويتم ذلك من خلال الدالة trim() والكود التالي يوضح طريقة إستخدامها[4] :

```
$search=trim($_GET['search']);
  if(isset($search) and !empty($search) )
```

```
{
الاستعلام#
}
```

وفي حال الرغبة في إجبار المستخدم على تحديد عدد الأحرف في عملية البحث فيمكن ذلك بإستخدام الدالة strlen() وطريقة إستخدامها يوضحها الكود البرمجي التالي الذي يجبر المستخدم على إدخال ٥ أحرف كحد أعظمي وإلا فإنه لا يتم الإستعلام الخاص بعملية البحث[4] :

```
search=trim($_GET['search']);
  if(isset($search) and !empty($search) and strlen($search)>=5 )
{
 الاستعلام#
}
```

نتائج البحث

١. التهديدات الأمنية على شبكة الإنترنت والمواقع التي تعمل عليها هي من أكبر التحديات التي يواجهها المبرمجين ومطوروا النظم.
٢. هذا النوع من الثغرات يتميز بالسهولة من ناحية التعلم والإستثمار ويشكل تهديداً فعلياً للمواقع والتطبيقات البرمجية التي تعتمد على قواعد البيانات.
٣. هذا النوع من الثغرات منتشر بشكل كبير في المواقع وذلك بسبب الإنتشار الواسع للمواقع والتطبيقات التي تكون إحدى مكوناتها قواعد البيانات.

٤. هذا النوع من الثغرات خطير جداً وخطورته تكمن في قدرة المهاجم على تطبيع الإستعلام ليكون في صالح المخترق وليصبح قادراً على إحضار المعلومات أو البيانات من قاعدة البيانات.

٥. إن عملية تشفير البيانات الموجودة ضمن قاعدة البيانات تعتبر من الأمور الهامة حتى لا يستطيع المخترق التعامل معها و الإستفادة منها في حال حصوله عليها وسرقتها بإستخدام هذا النوع من الثغرات.

٦. إن المعوقات المهمة التي تعوق حماية المواقع الإلكترونية من الإختراق هي غياب الإجراءات الخاصة بأمن البرمجيات وقواعد البيانات كتغيير كلمة المرور بشكل دوري، وتحميل برامج دفاعية ترفض أي برامج غريبة عن النظام، وأيضاً النسخ الاحتياطي للمعلومات، وتشديد الحماية على البريد الإلكتروني، وتثبيت قنوات آمنة عبر شبكة الإنترنت، وتحديد الجهات المخول لها الاتصال والمفاتيح العمومية.

٧. إن عملية التأكد من مدخلات المستخدم في الخادم أو السيرفر تُعتبر من الأمور الهامة التي يجب أخذها بعين الإعتبار.

٨. إن هذا النوع من الثغرات يؤثر بشكل سلبي على عملية التحقق من هوية المستخدم (Authentication)، فالمخترق يستطيع القيام بعملية الإتصال بالنظام دون معرفة كلمة المرور باستخدام حقن لغة الاستعلام وذلك لأن الكود البرمجي المكتوب بلغةSQL والذي يستخدم للتحقق من المستخدم وكلمة المرور ضعيف.

٩. إن هذا النوع من الثغرات يؤثر بشكل سلبي على السرية (Confidentiality)، فالمهاجم يتمكن من الوصول الغير مصرّح إلى نظام قواعد البيانات والإطلاع على البيانات الحساسة والعبث بها أو سرقتها.

١٠. إن هذا النوع من الثغرات يؤثر بشكل سلبي على التكامل (Integrity)، فالمخترق يستطيع إستخدام حقن لغة الاستعلام للتعديل والإضافة على البيانات الحساسة.

١١. إن هذا النوع من الثغرات يؤثر بشكل سلبي على إستمرارية توفر الخدمة (Availability)، فالمخترق يستطيع مسح البيانات والجداول من قواعد البيانات.

المراجع

[1] Allen Harper, Shon Harris, Jonathan Ness, Chris Eagle, Gideon Lenkey, ,Terron Williams, 2011-Gray Hat Hacking The Ethical Hacker's Handbook. The McGraw-Hill Companies, 3nd ed, United States,721 pages.

[2] Dafydd Stuttard, Marcus Pinto,2008- The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws. Wiley Publishing, Inc, Indianapolis, Indiana,771pages.

[3] Joel Scambray, Mike Shema,2002-Hacking Exposed Web Applications. Brandon

A. Nordin, United States of America,327pages.

[4] Justin Clarke,2009- SQL Injection Attacks and Defense . Syngress Publishing, Elsevier, United States of America, 494 pages.

[5] McDonald, Stuart, (2004), SQL Injection: Modes of Attack, Defence, and Why It Matters، [Available online]. from http://msdn.microsoft.com/en-us/library/ms161953.aspx

[6] Shreve, Justin, (2008), 5 Helpful Tips for Creating Secure PHP Applications ، [Available online]. Retrieved Des 22, 2009 from http://net.tutsplus.com/tutorials/php/5-helpful-tips-for-creating-secure-php-applications

[7] SQL Injection Attacks by Example, (2009), [Available online]. from http://unixwiz.net/techtips/sql-injection.html

[8] Sruthi Bandhakavi, Prithvi Bisht, P. Madhusudan and V.N. Venkatakrishnan, Preventing SQL Injection Attacks using Dynamic Candidate Evaluations

[9] Stuart McClure, Joel Scambray ,George Kurtz,2005- Hacking Exposed: Network Security Secrets & Solutions. The McGraw-Hill Companies, 5nd ed, United States of America, 692 pages.

[10] Stuart McClure, Saumil Shah, Shreeraj Shah,2003-Web Hacking: Attacks and Defense. Addison Wesley, United States of America ,528page.

[11] OWASP ,(2009),SQL Injection, [Available online]. Retrieved 30,8,2009, from http://www.owasp.org/index.php/SQL_Injection.

[12] http://www.codeproject.com/KB/database/SqlInjectionAttacks.aspx

[13] http://www.v4-team.com/cc/showthread.php?t=97448

[14] http://www.sans.org/reading_room/whitepapers/securecode/sql-injection-modes-attack-defence-matters_23

[15] -http://www.sqlmag.com/article/sql-server/3-free-tools-that-prevent-sql-injection attacks.aspx

[16] http://isc.sans.org/diary.html?storyid=4621

[17] جمال،هاني ، (٢٠٠٤) ، أخطاء المبرمجين الأمنية فى PHP ، تم استرجاعه في ١٤٣١/١/٥هـ على الرابط http://www.zajilnet.com/forum/index.php?showtopic=11213

Blind SQL Injection Exploits

Protection Against it and Patching it

ASISTANT: SHADI SHAMMAS

SUMMARY

The security threatings on the internet and its sites has become one of the biggest challenges in the present days , according to the huge evolvement in the technics which used in attacks . We can say that one of the easiest and the most dangerous attacks is the injuction gaps , in inquiring language , (SQL Injuction) . SQL has made a big and dangerous threatening to any site or a web application which the data base is one of its components . These gaps make attackers or Hackers able to get information and the sensitive and important data which has big and huge value in data bases . The ways of attack is distinguished through these gaps because of learning ease , beside these gaps may cause damages range from simple damages , possible damages to damages can hit the whole informational system . We also have to refer that many of applications and electronic sites on internet maybe affected by these gaps .

In this research , I make a study about the easiest and the most important and dangerous security gaps in sites which named the informational language injuction gap SQL . It considers the most common in breaking data base and stealing its information or destroy it . In addition , I mention some ways and methods can follow to protect data base from these gaps attacks , and they sometimes can help in closing these gaps and block it .