مقاله اي جامع در مورد امنيت و الگوريتم هاي امنيت داده ها.

سلام.

يكـي از دوسـتان در مقالـه اي از روش هـاي كدگـذاري داده هـا پرسـيده بـودن. همـين رو بهانـه خـوبي دونسـتم تا آنچه به نظرم در مورد امنيت داده ها مهم هست رو به نظر دوستان برسونم.
امنيت يك بحث كلي داره كه نميشه در چند خط به اون پرداخت.
امنيت به دو سطح تبديل ميشه: (جدا از بحث شبكه)

۱) امنيت در سطح برنامه
۲) امنيت در سطح داده هاي برنامه

در مورد امنيت در سـطح برنامـه، مطالـب فـراوان هسـت ولـي در يـك نظـر كلـي، منظـور از امنيـت برنامـه، ايجاد روش هايي در پيشگيري از ايجاد دستبرد در برنامه و منابع برنامه مورد نظر، در كوتـاه مـدت يـا بلنـد مدت است. (Web Application يا Desktop Application).

به عنوان يك مثال براي برنامه هاي Web Application عرض كـنم كـه ميشـه تنهـا بـا چنـد خـط كدنويسـي ساده، با ارسال مقادير زياد داده اي از طريق Post بـه سـايت مـورد نظـر، ترافيـك سـايت را بـه طـرز بسـيار زيادي بالا برده، Transfer سايت را تا حدي كه مايليـد زيـاد كـرده و منجـر بـه تركيـدن Band Width (پهنـاي باند) سايت شويد. (كاري رو كه بنده چند وقت پيش بـا قسـمت وبـلاگ سـايت PersianBlog.Com انجـام دادم. البتـه ايـن سـايت از نظـر سـخت افـزاري بسـيار قدرتمنـده و آسـيب چنـداني نديـد ولـي تـا ۲ هفتـه مسئولين سايت موفق به برطرف كردن مشكل نشدند. دوستاني كه يك ماه پيش به قسـمت وبـلاگ ايـن سايت سر زده باشند،بيشتر از پنجـاه هـزار Comment بـا عنـوان vb4u مشـاهده كردنـد كـه اجـازه ارسـال نظرات رو بـه اونهـا نمـي داد چـون بـراي يـك Comment بايـد Comment هـاي قبلـي پـردازش شـوند و از آنجايي كه زمان اجراي Script توسط برنامه نويس حداكثر يك دقيقه در نظر گرفته ميشـه و پـردازش پنجـاه هزار پيغام بيشتر از يك دقيقه طول ميكشه، بنابراين با خطـاي Script Time Out of Range روبـرو خواهيـد شد)

البته اين مشكل در اين سايت وجود نداره و برنامـه نـويس حرفـه اي سـايت بـا ايجـاد يـك Delay در ارسـال پست ها (كه البته تنها راه هم هست) مشكل رو حل كرده. بنابراين اگر دقت كـرده باشـيد، در زمـاني كـه قصد داريد ۲ تا پست رو پشت سر هم و در فواصل زماني اندك انجام بديد، پيغـامي مبنـي بـر اينكـه قـادر نخواهيد بود پست را بلافاصله بعد از پسـت بعـدي بفرسـتيد و بايـد مقـداري صـبر كنيـد، بـه شـما نمـايش داده خواهد شد.
و مثالي هم در مورد برنامه هاي Desktop Application:
خط زير را در نظر بگيريد:

كد:

```
If MyTextBox.Text = "123456” Then
   MsgBox "OK"
Else
   MsgBox "Fail"
End If
```

مثال بالا يكي از موارد اشتباه فاحشي است كه در بسياري از برنامه ها مشـاهده شـده و كلمـه عبـور را به راحتي با يك Hex Editor مي توان به دست آورد.

و اما بحث اصلي من در مورد امنيت در سطح داده هاي برنامه است.
به طور ساده مي تونيم بگيم كه منظور از امنيت در سطح داده، كدگذاري داده است.
يعني ابداع يـا اسـتفاده از روش هـايي كـه باعـث ميشـه داده هـاي مـورد نظـر مـا بـه كاراكترهـايي تبـديل شوند كه تنها براي كساني كه مجاز به استفاده از آنها هستند مفهوم داشته باشند.
در بحث داده ها كه عموم بحث بر سر بانك هاي اطلاعاتي است، دو مورد مورد توجه است:

۱) سيستم مديريت پايگاه داده يا DBMS (DataBase Management System)
۲) نحوه ذخيره داده ها در DBMS

انتخاب مورد اول به عهده كاربر است. يعني با توجه به فاكتور ((اهميت اطلاعات)) انتخاب مي شود.

در ساده ترين و كم اهميت ترين سطح (در زماني كه امنيت و در درجه دوم، سرعت مهم نباشد) يك فايل متني ساده، پاسخگوي نياز ما است.
اكثر كاربران كه در سطح آغازين وارد دنياي Database مي شوند، كار خو درا با بانك اطلاعاتي نرم افزار Access آغاز مي كنند. (كه مطمئنا در يك طراحي حرفه اي و آنجا كه امنيت مطرح باشد، به هيچ وجه پاسخگوي نياز ما نيست.)
امروزه بانك اطلاعاتي نرم افزار SQL Server كه يك نرم افزار كلاينت-سرور است در بيشتر كاربردها به عنوان يك منبع ذخيره سازي قابل اطمينان كه ضريب امنيتي بالغ بر ۱۸ برابر بانك Access دارد، مورد استفاده است. (۱۸ لايه امنيتي)
در سطوح بالاتر نيز، بانك هاي MySQL و Oracle قرار دارند.

و اما مورد دوم كه بحث اصلي ما است.
نحوه ذخيره سازي داده ها به طور پيش فرض به شكل كاراكتري است. حال اگر بتوان يك DBMS قدرتمند را با يك الگوريتم قدرتمند كدكردن داده ها تركيب كرد، مسلما در سطح امنيت توانسته ايم به جايگاه بالايي دست پيدا كنيم.
نكته مهم: به اين نكته مهم توجه داشته باشيد كه نظم، دشمن امنيت است.
هرچه داده هاي ما منظم تر باشند، امكان دستبرد به آنها بيشتر است. به همين خاطر است كه در امور كوچك (نگهداري اطلاعات سربازان يك پادگان)، فايل Pile بيشترين كاربرد را دارد نه فايل ترتيبي (Sequential) يا شاخص دار (Indexing).

دو الگوريتم محبوب كه در كدگذاري داده ها (Encrypt) بيشتر مورد استفاده قرار مي گيرند، به شرح زير هستند:

۱) الگوريتم Huffman.
۲) الگوريتم Cipher (كه ويندوز نيز در كدگذاري داده ها از اين الگوريتم استفاده مي كند)

الگوريتم Huffman، يك الگوريتم قديمي است كه در ايران براي دانشجوهاي كامپيوتر در درس ساختمان داده ها و به عنوان مبحث درخت هاي ويژه كامپيوتري تدريس مي شود.
البته بايد مقداري با ساختمان و مبحث درخت ها آشنايي داشته باشيد.
فرض كنيد رشته زير براي كدگذاري وجود دارد:

كد:

```
aacccbeeee
```

قصد داريم رشته فوق را با استفاده از الگوريتم Huffman، كدگذاري كنيم.
ابتدا تعداد تكرار هر يك از كاراكترها را مي نويسيم:

كد:

```
a = 2
b = 1
c = 3
e = 4
```

پس درصد احتمال ظهور هر يك از كاراكترها به شرح زير شد:
كاراكتر a: 2 به ۱۰
كاراكتر b: 1 به ۱۰
كاراكتر c: 3 به ۱۰
كاراكتر e: 4 به ۱۰

حالا در هر مرتبه هر دو كاراكتري كه احتمال ظهورشان از بقيه كمتر است را با هم تركيب مي كنيم:
كاراكتر b با كاراكتر a كه مي شود ۳
كاراكتر c با كاراكتر e كه مي شود ۷
و در نهايت ۳ با ۷ كه مي شود ۱۰ و ريشه ي درخت را تشكيل مي دهد.
در درختي كه ايجاد مي كنيم، گره هاي داخلي بيانگر جمع درصد ظهور كاراكترها و گره هاي خارجي بيانگر نام كاراكتر است.

البته در اينجا نميشه شكل درخت را كشيد ولي پس از پايان ايجاد درخت، به شاخه هاي سمت چپ عدد صفر و به شاخه هاي سمت راست عدد يك نسبت داده مي شود و در نتيجه، كد Huffman داده ي مورد نظر به شكل زير به دست مي آيد كه به صورت صفر و يك است.

كد:

```
a = 01
b = 11
c = 01
d = 00
```

البته در VB به خاطر ساپورت نكردن از اشاره گرها، پياده سازي اين الگوريتم قدري مشكل است اما غير ممكن نيست. ولي كد زبان C اين الگوريتم را به عنوان پروژه يك درس نوشته ام كه اگر دوستان خواستند تبديل به VB بكنند در اختيارشون خواهم گذاشت.

و اما الگوريتم رمز Cipher كه البته اين الگوريتم يك مقدار قديمي هست ولي اينقدر اين الگوريتم كارامد هست كه ويندوز نيز از اين الگوريتم در كدكردن داده ها استفاده مي كند.
توجه: مي توانيد در پوشه نصب ويندوز و در پوشه System32 فايل Cipher.exe را پيدا كنيد.
و اما نحوه ي عملكرد الگوريتم Cipher:
كليد اين الگوريتم رمز به صورت K=(K1,K2,00000000,Kn) است كه در اين كليد، حرف N، يك عدد ثابت و مثبت و معرف طول كليد است.
تابع رمز گذار (Encrypt) الگوريتم Cipher به شكل زير است:

كد:

```
Ek(x1,x2,000000,xn) = (x1+k1,00000+kn) mod 26
```

و تابع رمزگشاي (Decrypt) الگوريتم Cipher نيز به شكل زير است:

كد:

```
Dk(y1,00000,yn) = (y1-k1,00000,ym-km) mod 26
```

در تابع رمزگذار، xi، حرف متن اصلي و در تابع رمزگشا، yi، حرف متن رمز شده است.
مثال:
فرض كنيد براي كليد رمز، كلمه ي CIPHER را انتخاب كرده ايم. پس در نتيجه، n=6 خواهد شد و معادل كليد رمز، عبارت است از:

كد:

```
k=(2,8,15,7,4,17)
```

و متني كه مي خواهيم به صورت رمز درآيد، عبارت زير است:

كد:

```
"thiscryptosystemis"
```

براي رمز كردن اين متن، آن را به گروه هاي شش تايي تقسيم مي كنيم و كليد را در زير آنها مي نويسيم. يعني:

كد:

```
t h i s c r y p t o s y s t e m i s
c i p h e r c i p h e r c i p h e r
```

حالا معادل عددي هر حرف از دو رديف را مي نويسيم. يعني:

كد:

```
۱۸ ۸ ۱۲ ۴ ۱۹ ۱۸ ۲۴ ۱۸ ۱۴ ۱۹ ۱۵ ۲۴ ۱۷ ۲ ۱۸ ۸ ۷ ۱۹
۱۷ ۴ ۷ ۱۵ ۸ ۲ ۱۷ ۴ ۷ ۱۵ ۸ ۲ ۱۷ ۴ ۷ ۱۵ ۸ ۲
```

توجه: منظورم از معادل عددي، مكان عدد در بين حروف الفباست كه توجه داشته باشيد از صفر شروع مي شود. مثلا: A برابر صفر، B برابر يك، C برابر دو و ...
حال عدد به دست آمده در دو رديف را در مبناي ۲۶ با هم جمع مي كنيم كه مي شود:

كد:

۲۱ ۱۵ ۲۳ ۲۵ ۶ ۸ ۰ ۲۳ ۸ ۲۱ ۲۲ ۱۵ ۲۰ ۱ ۱۹ ۱۹ ۱۲ ۹
حالا عدد به دست آمده را معكوس كرده و به معادل حروف الفبايشان تبديل مي كنيم كه مي شود:

كد:

"VPXZGIAXIVWPUBTTMJ"
پس عبارت معادل رمز شده ي كلمه مورد نظر ما به شكل فوق حاصل شد.
براي رمزگشايي نيز از همان كليد استفاده مي شود و فقط به جاي عمل جمع، از تفريق در مبناي ۲۶ استفاده مي كنيم.

اين هم از اين بحث.
اميدوارم كه مفيد فايده بوده باشه و از اينكه اين مقاله را خوانديد ممنونم.
با بهترين آرزوها...

نويسنده : بهروز راد

كليه حقوق اين مقاله متعلق به نويسنده و سايت برنامه نويس مي باشد .
استفاده از مطالب اين مقاله در صورت ذكر منبع مجاز است .
www.barnamenevis.org