Fast Bitcoin data mining

Tutorial

(used in Cryptanalysis COMPGA18 and Applied Cryptography GA12, part of UCL M.Sc. Information Security)

version 1.09 March 2016

Nicolas Courtois Jason Papapanagiotakis Guangyang Song Chris Park

blog.bettercrypto.com

Introduction

This document is aimed to help anyone in mining the Bitcoin blockchain for interesting events at state-of-the-art speed. We assume very little prior knowledge about programming. This tutorial is separated into 3 parts:

- Part 1 demonstrates how to setup and run the BitcoinDatabaseGenerator project that will parse the blockchain files and create an SQL database.
- Part 2 shows the modifications necessary to add additional fields to the database.
- Part 3 is a tutorial on how to extract random numbers used more than once for the elliptic curve digital signature in the Bitcoin blockchain using the previously created database.

Part 1

The following components are necessary and we provide step by step instructions.

1. Bitcoin core (free), also tested to work with Bitcoin XT
2. Microsoft SQL EXPRESS 2008/2012/2014 (free but with some issues, we strongly recommend a full version, see section Limitations).
3. A free tool called BitcoinDatabaseGenerator compiled with Visual Studio 2013. We will later add functionality to this tool.
4. Visual Studio 2008 or higher to access the data at very high speed in C/C++.

1.1 Step 1. Bitcoin Core

1.1.1 Install Bitcoin Core

Download the software from: bitcoin.org/bin/

Note:

When asked for the bitcoin file location, specify a drive with a lot of space (at least 60 GB required).

For example: “X:\active_live_blockchain\satoshi120”

1.1.2 Run Bitcoin Core

Then here is the command line (or desktop shortcut) exact syntax to run it:

"C:\Program Files\Bitcoin\bitcoin-qt.exe" "-datadir=X:\active_live_blockchain\satoshi120"

Run for several days to synchronise with network.

1.1.3 Stop Bitcoin Core

Stop it and wait for 1 minute or so.

1.2 Step 2: Install Microsoft SQL.

We need at least Microsoft SQL EXPRESS 2012 (free) or better.

WE STRONGLY recommend FULL PAID version like Microsoft SQL Server 2014 Enterprise edition.

- It is free for UCL staff at e5.onthehub.com.
- We also advise to install the "Management Studio".
- All other options below have some unpleasant limitations.

1.2.1 Pre-requisites - Windows

We require any version of Windows x64 (99 % of PCs are x64 nowadays). Can be any of Windows 7,8,10 or 2008 Server or even XP x64 version.

The file system must be NTFS, not FAT32.

The root SQL directory should not be created in a compressed directory (or needs to be de-compressed before working with it).

Note: Virtual Windows under MAC or Linux is OK and can be sometimes faster than native Windows. We recommend Oracle VM Virtual Box for this.

1.2.2 Check SQLEXPRESS and other existing installs in Windows

Optional, one can skip this section.

In order check if it is OK, go to My Computer=>management=>services and search for SQL services.

- Roughly speaking good configuration makes that an SQL server service is installed set to Automatic and Started/running.
- Beware of windows 10: here SQL EPXPRESS 2008: it is not compatible and has lots of problems, better to uninstall.
- On some machines there is none. OK.
- If there is one already, keep it,
  we will install another instance in order to deal with large files.

1.2.3 Get SQLEXPRESS 2012 (free but not perfect)

(A paid version is a again lot better, see section Limitations).

We can use Microsoft® SQL Server® 2012 Express version (free)

microsoft.com/en-gb/download/details.aspx?id=29062

- For small SQLEXPRESS service installation
  - This is already preinstalled on many Windows PCs, right click on my computer and see if SQLEXPRESS is already running in services, it might be necessary to change it from Disabled to Automatic and then Start it (if not running). If does not start, uninstall it totally in Programs and Features.
- If not already installed, do as follows:
  - run ENU\x64\SQLEXPR_x64_ENU.exe 132.3 MB
  - main install: only the core Express database server engine (windows Service, visible in My Computer=>management=>services).

Tested and works with limitations:

database size will be limited to 10 Gigabytes, not sufficient, but good for testing.

1.2.4. Local DB Variant - Optional

localdb variant (not using Windows services!).

Tested to work with Visual Studio 2010.

Install Local DyDb: by running ENU\x86\SqlLocaLDB.MSI

=> LocalDB is a lightweight version which runs in user mode.

- Usage:
  "Server name" = “(localdb)\v11.0”
  "Driver={SQL Server Native Client 11.0};"

  SERVER=(localdb)\v11.0;

  IntegratedSecurity=true;

  MultipleActiveResultSets=True;

  AttachDbFileName=C:\MyData\Database1.mdf
- This only works if we already have a working mdf file.
- One way to create a correct yet empty file D:\MyData\Database1.mdf initially is to create it with Visual Studio Server Explorer pane as follows:
  - Right click and ask to create a new SQL database, server name=SQLEXPRESS, Database name to be created = Database1
  - Go to the right/their directory (for example C:\Program Files\Microsoft SQL Server\MSSQL11.SQLEXPRESS12\MSSQL\DATA)
  - Stop THE SQLEXPRESS service for 1 minute
  - Select two files Database1.mdf and Database1.ldf and just copy (better NOT erase originals) them to any directory like D:\MyData\.
- TIP: to make connections A LOT FASTER on needs to disable TCP/IP for this server: type SQLServerManager11.msc right click on "MS SQL Server Native Client 11.0 Config (32bit) go to ‘Client Protocols’ and DISABLE TCP/IP for and enable only 'memory sharing' (as rank order #1) and 'named piping' (as rank order #2)

- Optional / avoid
  BIG install which may conflict with some versions of Visual Studio or it requires Visual Studio 2010 updates (most complete stand-alone install).
  - Run
    ENU\x64\SQLEXPRADV_x64_ENU.exe 1.3 GB
  - This installs
    Express with Tools + LocalDB + advanced services:
    THE FULL version (no other installer needed): includes the database engine and SQL Server Management Studio Express). Has both LocalDB and Express, SQL Azure, Express Tools, Reporting Services and Full Text Search.

1.2.5. Installing fresh SQLEXPRESS instance (optional)

Several instances of this SQL server can be installed on any PC. Because we are going to create very large files, like 60 GB, we recommend to install a separate instance as follows:

- run ENU\x64\SQLEXPR_x64_ENU.exe
- select New stand-alone installation
- fill "Named instance:" = SQLEXPRESSBIG
- instance root directory:
  - change C:\Program Files\Microsoft SQL Server\ to something like
  - X:\BIG Microsoft SQL Server\
- This is where large files will be stored!!

[optional]

Emergency solution for lack of space: it is also possibly AT RUNTIME or later to stop windows SQL service, rename DATA to DATA2 (for now), re-create DATA directory and map this exact empty NTFS directory with Disk Management to a new partition with lots of space, then move the DATA2 there, and restart the windows service ("Start").

1.2.6 Limitations – full SQL Server is the best.

10 GB is the maximum size of one database in SQLEXPRESS or similar. This is like 82 dat files only. Solutions are:

- There is less limitation if the SQLSERVER 2012 (the most basic edition) is installed under "Windows10 Pro". (Actually larger database can be produced only as the unique default "master" database, not for a names custom one like MyBitcoinData).
  - In another run we have used SQLEXPRESS 2008 fresh minimal install and Windows 8.1. Enterprise. This also worked (54 Giga bytes) when we used the master database.
  - Not OK, not a solution: SQL EXPRESS 2014 also has 10 GB size limit. However, if installed in a "good" windows version like above, the master can be large.
  - Not OK, not a solution: SQL Server 2014 Developer Edition is 46 GBP (public price) on microsoftstore.com. Reportedly 5 GB limit.
- **Get an upgrade:** For example, **SQL Server 2014 Enterprise edition** is free for UCL staff at e5.onthehub.com.

1.3 Step 3: Building BitcoinDatabaseGenerator

We use the tool called BitcoinDatabaseGenerator compiled with VS 2013.

Download it from github.com/ladimolnar/BitcoinDatabaseGenerator

- We need the following files:
  - BitcoinDatabaseGenerator.exe
  - some dlls in the same directory:
    - ZeroHelpers.dll + AdoNetHelpers.dll + BitcoinBlockchain.dll + BitcoinDataLayerAdoNet.dll + ResharperAnnotations.dll
  - [optional] and maybe also these: Microsoft.Threading.Tasks.dll + Microsoft.Threading.Tasks.Extensions.Desktop.dll + Microsoft.Threading.Tasks.Extensions.dll
- Here is an older ready exe from us [not the latest]:
  - https://drive.google.com/file/d/0BwTDoLtsy_L9SEszZ3pNeXU4U28/view?usp=sharing

  Unpack in one directory like "d:\work"

The normal (slow way) is to compile all projects inside called BitcoinDatabaseGenerator with Visual Studio 2013.

- Good the change the output directory in each project to our chosen directory like "d:\work" which will contain the exe and dlls to be run.
- If visual studio complains that he cannot find sth like StyleCop.targets,
  - Do NOT upgrade Visual 2013 (fails anyway)
  - SOLUTION 1 just remove the line which adds StyleCop.targets from inside
  - \BitcoinDatabaseGenerator\Sources\BitcoinDatabaseGenerator\BitcoinDatabaseGenerator.csproj
- SOLUTION 2: download and install StyleCop.targets from stylecop.codeplex.com/downloads/get/323236

1.4 Step 4: Create a Database

[Optional, this step 4 is actually not needed]

One method is to use Visual Studio. In Visual Studio 2010 [or another version]:

- Go to View => Other windows => Server Explorer pane (Ctrl+Alt+S), place conveniently
- Under Data Connection right click "Create New SQL Server Database"
  in Server Name put PC-NAME \SQLEXPRESSBIG below,
- your PC-NAME can be found under
  My Computer -> right click -> properties
- Under New Database name put MyBitcoinData, click OK

Result: this will create 2 files:

- X:\BIGMicrosoftSQLServer\
  MSSQL11.SQLEXPRESSBIG\MSSQL\DATA\MyBitcoinData.mdf
- X:\BIG Microsoft SQL Server\
  MSSQL11.SQLEXPRESSBIG\MSSQL\DATA\MyBitcoinData_log.ldf

Remark:
MOVING FILES: these database files CAN be moved (preferably together) to another computer, just stop the SQL service for SQLEXPRESSBIG in My Computer=>management=>services, copy files and Start it again!

Warning: erasing the database called “master” will prevent SQLEXPRESS from starting again, need to uninstall and re-install carefully (slow process).

1.5 Step 5: Run BitcoinDatabaseGenerator.exe

In windows search type “cmd”,

run cmd.exe then type:

- D:
- CD d:\work
- BitcoinDatabaseGenerator.exe /BlockchainPath X:\active_live_blockchain\satoshi110\blocks /SqlServerName PC-NAME\SQLEXPRESSBIG /SqlDbName MyBitcoinData

- WOW, this is incredibly fast!!
  - We get like 12-15 seconds per each .dat file on a dual core laptop CPU.
  - The writing speed (incl. writing the log file) is: up to 50Mb/s on my laptop.

Once this process is complete you can browse the data inside the database with tools like SQL Server Management Studio that comes for free with SQL Server.

1.5.1 Work Around for 10 GB Limitation

As already explained with free editions of SQLEXPRESS it is possible to work with databases >10 GB in the following way.

- This works in many but not all versions of Windows, as explained above, Windows10 Pro and Windows 8.1. Enterprise work for sure.
- In all database projects, do not use a custom database, but do everything with the master. Replace “MyBitcoinData” by “master” everywhere.

1.5.2 Ready Database

Our ready database (54 Gbytes, last updated 10/2015) can be found at:
https://drive.google.com/folderview?id=0BwTDoLtsy_L9cFFiODVjd2w0WWs&usp=sharing

Part 2

This part is guide on the necessary steps you need to take to add additional columns to the database. This can help to extract further data from the blockchain into the database, be aware that this could also increase significantly the size of the database.

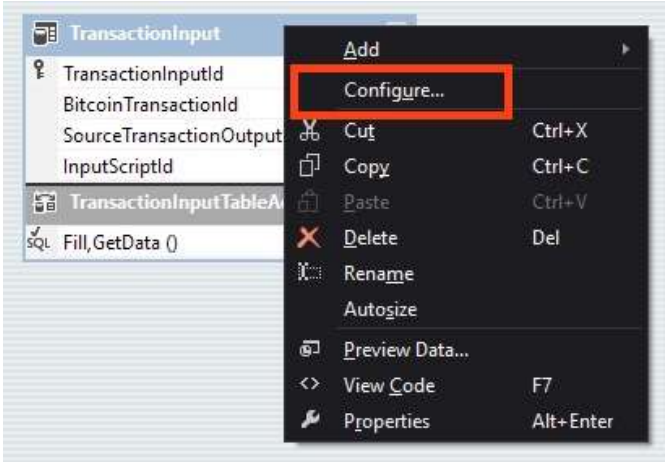
2.1 Code changes

Build

1. NuGet package (2.8.1 or above) or visual studio update (3 or above) needed

How to Add a Column (InputScriptID)

1. BitcoinDataLayerAdoNet/Schema/Tables.sql
   - This is modifying the database scheme.
   - In the table, in which you would like to add a column, add a column name with a type.
   - For example, in "TransactionInput", "InputScriptID BIGINT NOT NULL" is added.
2. Recompile & Run (/DropDb option)
   - This is for applying new database scheme with the new column added.
   - By running, it will apply the new "Table.sql" database scheme that you changed to the database.
3. DataSets/[Table_where_new_column_added].xsd -> configure
   - Go to the table you modified.
   - Right click the name of the table.



Go to Configure…

- Select the database you are using. Next
- Select "Use SQL statements". Next
- Click "Query Builder…"
- Tick the column you added (in this case InputScriptID).OK then Next, Finish.

You have to be able to see the new column added on .xsd file.

- BitcoinDatabaseGenerator/SourceDataPipeline.cs
    - This is where actual data is transferred to the database. The column you want to add should be coded here.

You might find useful to look at the given APIs to add a useful column in the db. (~\BitcoinDatabaseGenerator-master\Sources\packages\BitcoinBlockchain.1.2.0.0\lib\net45\BitcoinBlockchain.XML)

- For example, "this.transactionInputDataSetBuffer.TransactionInput.AddTransactionInputRow()" method has one more argument, which is "inputScriptId++".

BitcoinDataLayerAdoNet/Data/TransactionInput.cs

Add the new column to the constructor and add the method.

- For example, "public TransactionInput()" has one more argument, which is "long inputScriptId".

Part 3

This part is a step-by-step tutorial on how to extract random numbers used more than once for the elliptic curve digital signature in the Bitcoin blockchain.

3.1 Introduction

Randomly choose the same 32-byte long number more than once should happen only with negligible probability but due to various reasons this phenomenon can be observed in the public Bitcoin ledger. This can lead to different types of attacks as described by Courtois [1] where some or all users can steal the contents of a wallet that used one of those "bad randoms". This makes the discovery of those numbers and the study of the way they appear a worthwhile goal.

3.2 Prerequisites

In order to recover all reused random numbers, the complete Bitcoin blockchain is needed in a database. To do this follow the instructions found in Part 1 and 2. Once you have completed those steps proceed to the next section.

3.3 Extracting reused random numbers

In this step you are going to create two additional tables in the database that was previously created. The first table will be called *Randoms* and contain a list of all random numbers used together with the ID of the *inputscript* in which they appeared, finally, the second table will be called *BadRandoms* and will store all random numbers that appeared more than once together with a counter of how many times they have been used.

[1] Courtois, N.T., Valsorda, F. and Emirdag, P., 2014. Private Key Recovery Combination Attacks: On Extreme Fragility of Popular Bitcoin Key Management, Wallet and Cold Storage Solutions in Presence of Poor RNG Events.

In order to create the two tables mentioned, you will need Microsoft SQL Server Management Studio that should have been installed together with SQL Server, if not, download and install the corresponding version for your database, this software is free.

The next step is to download the *RandomsToTables.sql* script from here https://github.com/JasonPap/Reused-Bitcoin-Numbers/blob/master/SQL/RandomsToTables.sql

and open it with SQL Server Management Studio and execute it (F5), this will take up to 3 hours depending on the hardware and the load of the machine. Once the query is completed you can look at the two new tables.

3.4 Generating webpage with reused random numbers

The final step of this tutorial is about how to generate a webpage that contains the random numbers used more than once together with the transactions in which they appeared.

1. Download and install Python 2.7
2. Next you will need *pymssql*, a Python module that makes the connection to SQL Server very easy.
    a. Download it from here and place the file into C:/Python27 https://www.lfd.uci.edu/~gohlke/pythonlibs/#pymssql
    b. Now install it using pip command line, cd into C:/Python27 and run the following: `pip install pymssql-2.1.1-cp27-none-win_amd64.whl`
    c. Instructions to enable the use pip can be found here https://stackoverflow.com/questions/4750806/how-do-i-install-pip-on-windows
3. Download the python script db2wb.py from here https://github.com/JasonPap/Reused-Bitcoin-Numbers/blob/master/Python/db2wb.py
4. Edit the 6th line of the file and insert the details of your database (user/password)
5. Run the script from command line: *python db2wb.py*

After those steps a file named index.html should appear on the same directory as the python script. Open it with any web browser. You can download the final result from here based on data collected the 07/03/2016. https://github.com/JasonPap/Reused-Bitcoin-Numbers/blob/master/Python/index.html