All About That (Data)Base

Matt Bromiley
Jacob Christie
Dedicated to Ken Johnson
Agenda

- Who We Are
- Why Do We Care So Much About SQL?
- Making the Challenge Harder
- Those “Other” Artifacts
- Conclusion
WHO WE ARE
$ whoarewe

Matt Bromiley works at Mandiant, having lived a previous life working with crazy chefs and 2 of the Big 4

[＠]505forensics[.com]

Jacob Christie is a Consultant at Mandiant, having cut his teeth doing desktop support and having done forensics at 2 of the Big 4

jc4n6.blogspot.com
WHY DO WE CARE SO MUCH ABOUT SQL?
Why Do We Care So Much About SQL?
MAKING THE CHALLENGE HARDER
Making the Challenge Harder

- In previous talks, we have covered research into using a SQL database to pull back artifacts
  - Direct DB artifacts can be valuable
  - Can be damaging to volatile artifacts
  - Requires an analyst to know SQL, or a DB admin to (let you on the box OR run code for you)
  - None of the above are common

- Yes, we’re all asked to wear many hats. How many are also asked to be a DBA?

- Databases were not designed as forensic artifacts
  - They’re designed to store, sort, index, optimize, and deliver data quickly
  - Beat the crap out of memory
Making the Challenge Harder (cont.)

- So we thought….take away the database
- Put an analyst in a situation they are comfortable being in
- Let the analyst use tools they are already familiar with
- Why?
  - Let’s your team add to the workflow, instead of building a new workflow
  - No one has to become a SQL guru to interpret data
  - Value is provided to the client faster, with a better delivery of results
Making the Challenge Harder (cont.)

- What does that leave us with?
  - External logs
  - AKA Logs that can be opened *without* having to use a SQL Server client
  - Possibly copying off SQL data/log files and loading into a separate instance
    - Great; gives us a view of the data
    - Does it give us a view of the attacker?
  - Saving the best for last…memory!!
Making the Challenge Harder (cont.)

- Why memory?
  - Microsoft wants your memory

- SQL Server 2014+ utilizes In-Memory Optimization, which allows developers and DBAs to optimize the DB based on business requirements
- Also means SQL Server is pushing much more “stuff” to memory
- Just would be nice if more companies upgraded their SQL instances…
Making the Challenge Harder (cont.)

- Why memory? (more important)
  - Forensic analysts are comfortable with memory acquisitions
    - If they're not...you need to talk to your team
  - The basics:
    - Analyzing memory allows an analyst to quickly pivot, whereas a siloed artifact approach doesn’t work
THOSE “OTHER” ARTIFACTS
Logs
Those “Other” Artifacts - Logs

- Logs!
  - SQL-Generated
    - Trace Logs
    - Error Logs
  - External
    - Windows Event Logs
    - Web Logs
    - Firewall Logs
    - Network Traffic (Netflow, PCAP, etc.)
Those “Other” Artifacts (cont.) – SQL-Generated Logs

- **Trace Logs**
  - Internal because they come from within SQL; but you may find a client that exports these logs and stores elsewhere;
  - Store information about logins, DML/DDL operations
  - SQL Server 2005+ runs default trace
    - Stored automatically
  - Low volatility; files are persistent
  - New trace file is created each time MS SQL is started
  - Can be removed from a system and preserved forensically
  - Examined later on a separate, analysis system
    - Use ‘SQL Server Profiler’ to load a trace file
    - Can be loaded using a SQL Server client *elsewhere*
Those “Other” Artifacts (cont.) – SQL-Generated Logs

- **Error Logs**
  - Same location as trace files
  - Plain text (FINALLY!), can be opened by ${insert_your_fav_text_editor}
  - Tracks startup, shutdown, login, logout, authentication success/failures
    - Oddly enough, not just errors..
  - Can be removed from a system and preserved forensically
  - Examined later on a separate, analysis system
Those “Other” Artifacts (cont.) – SQL-Generated Logs

- **Error Logs (cont.)**
  - Important to track account activity!
  - Authentication Failure States ->
  - Can be very useful to detail attacker activity

<table>
<thead>
<tr>
<th>State Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Invalid ID</td>
</tr>
<tr>
<td>5</td>
<td>Invalid ID</td>
</tr>
<tr>
<td>6</td>
<td>Invalid Windows login attempt</td>
</tr>
<tr>
<td>7</td>
<td>Incorrect password/disabled</td>
</tr>
<tr>
<td>8</td>
<td>Incorrect password</td>
</tr>
<tr>
<td>9</td>
<td>Invalid password</td>
</tr>
<tr>
<td>11</td>
<td>Server access failure with valid creds</td>
</tr>
<tr>
<td>12</td>
<td>Server access failure with valid creds</td>
</tr>
<tr>
<td>18</td>
<td>Password must be changed</td>
</tr>
</tbody>
</table>
Those “Other” Artifacts (cont.) – External Logs

- **Windows Event Logs**
  - System, Security, Application
  - Vista+ may give customized Application-specific logs

- **Web Logs**
  - How does an external user access your DB? *Log this route!*

- **Firewall Logs**
  - Know your ports, know your IPs
  - Suspected data theft? Know how to aggregate byte counts

- **Network Traffic (Netflow, PCAP, etc.)**
  - Know your ports, know your IPs
THOSE “OTHER” ARTIFACTS

Memory
Those “Other” Artifacts (cont.) - Memory

- Memory is an essential artifact in database forensics
- Anyone want to try and image a DB server?
- 128GB > 6TB
Those “Other” Artifacts (cont.) - Memory

- The SQL Data Page
Those “Other” Artifacts (cont.) - Memory

- The SQL Data Page (cont.)
  - THE storage format for many things…not just SQL!
  - 8092 bytes in length
    - Header (96 bytes)
    - Data + Pointers (7996 bytes)
    - Pointer structure is stored in reverse (first record is stored last)
    - Any hints as to what a “footer” might be?
Those “Other” Artifacts (cont.) - Memory

- SQL Data Record Header

\x01\x01\x00\x00\x20\x02

m_headerVersion
m_type
m_typeFlagBits
m_level
m_flagBits
Those “Other” Artifacts (cont.) - Memory

- **SQL Data Record Footer**
  - Any thoughts?
  - \x60\x00
  - start of first record

<table>
<thead>
<tr>
<th>Offset</th>
<th>0037953360</th>
<th>0037953376</th>
<th>0037953392</th>
<th>0037953408</th>
<th>0037953424</th>
<th>0037953440</th>
<th>0037953456</th>
<th>0037953472</th>
<th>0037953488</th>
<th>0037953504</th>
<th>0037953520</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>21 21 21 21 21 21 E4 1E 87 1E 2A 1E CD 1D 70 1D</td>
<td>13 1D B6 1C 59 1C FC 1B 9F 1B 42 1B E5 1A 88 1A</td>
<td>2B 1A CE 19 71 19 14 19 B7 18 5A 18 FD 17 A0 17</td>
<td>43 17 E6 16 89 16 2C 16 CF 15 72 15 15 15 B8 14</td>
<td>5B 14 FE 13 A1 13 44 13 E7 12 8A 12 2D 12 D0 11</td>
<td>73 11 16 11 B9 10 5C 10 FF 0F A2 0F 45 0F E8 0E</td>
<td>8B 0E 2E 0E D1 0D 74 0D 17 0D BA 0C 5D 0C 00 0C</td>
<td>A3 0B 46 0B E9 0A 8C 0A 2F 0A D2 09 75 09 18 09</td>
<td>BB 08 5E 08 01 08 A4 07 47 07 EA 06 8D 06 30 06</td>
<td>D3 05 76 05 19 05 BC 04 5F 04 02 04 A5 03 48 03</td>
<td>EB 02 8E 02 31 02 D4 01 77 01 1A 01 BD 00 60 00</td>
</tr>
</tbody>
</table>
Those “Other” Artifacts (cont.) - Memory

- Let’s parse it!
- Our first Volatility parser – sqldatafinder.py
  - Modeled after earlier plugins that allow for header/footer searching
  - Parses a memory image, finds sqlserver process
  - Finds this, so you don’t have to!
  - Extracts data pages to an output directory

Next Steps:
- Output data pages in order of DB; reconstruct tables if possible
Those “Other” Artifacts (cont.) – Memory Parsing

- What is the value?
- The closer we are to the breach; the more relevant the data in memory (duh)
- Does my database contain…
  - PHI
  - PII
  - PCI
  - Something else offensive
- Did the attacker..
  - Create a new table
  - Make a change that has not committed yet
  - Run malicious commands
Those “Other” Artifacts (cont.) – Memory Parsing

- What else?
  - Anyone ever interact with a SQL Server?
  - Your attacker has too!
  - What tools are typically used to interact with SQL?
Those “Other” Artifacts (cont.) – Memory Parsing

- SQL Server Management Studio contains valuable artifacts as well!
  - If the process was still running or if the remnants are still available
- By dumping memory and parsing, we can identify queries issued by management studio
- We can also capture the text of the user’s query box

```sql
Use AdventureWorks2008R2;
Select * from HumanResources.Employee.PayHistory
```
Those “Other” Artifacts (cont.) – Memory Parsing

- What’s next?
- Second Volatility parser (TBD); sqlmsparser.py
  - If found, parse the contents of SQL Management Studio to identify text box queries and/or queries issued
  - Output commands to analyst
- Still working on this structure; highly dependent on the user as SQL code is not space/case/whitespace sensitive
  - Yes, a query can have 100 lines in between two statements
Those “Other” Artifacts (cont.) – Memory Parsing

- Lastly…

- The plan cache

  - When a statement is received, the query processor identifies the most efficient way to execute; the “execution plan”
    - Hence, the “plan cache”
    - Stores previously executed SQL “plans”

- Can be used to find queries from stored procedures and functions
  - Not all commands touch table records, or force a data page
  - Great for finding user account activity/creation commands
  - Some details are stored in plain text!!
Those “Other” Artifacts (cont.) – Memory Parsing

- The plan cache (cont.)

- HIGHLY VOLATILE
  - Subject to SQL flushing policies
  - Stored procedure updates
  - Stopping the SQL Server service (clears associated memory)
  - May be cleared by an attacker!!!
Those “Other” Artifacts (cont.) – Memory Parsing

- The plan cache (cont.)
Those “Other” Artifacts (cont.) – Memory Parsing

- The plan cache (cont.)
- Inside of SQL, the plan cache can be accessed/converted

```
SELECT * FROM sys.dm_exec_sql_text(plan_handle)
```

- Third Volatility parser (TBD); sqlplancache.py
  - Identifies the plan cache in memory, extracts and converts to nice, pretty queries
  - Still defining structure, however all components are in memory
Conclusion

- Determine if the team needs to change based on an artifact, or if the artifact can changed based on your process
  - Do we need to go to the DB?
  - **Caveat:** Never change an artifact

- Non-database SQL artifacts can provide a lot of value for your team; decide if it’s relevant to your investigations

- Learn what the artifacts how, and how to parse, interact with, and interpret them.
  - Do error logs only contain errors?

- Writing Volatility parsers is freakin’ fun; do it!
THANK YOU!