Practical Methods for Dealing with Full Disk Encryption

Jesse Kornblum
Outline

- Introduction
- Types of Targets
- Finding Keys
- Tool Marks
- Example - BitLocker
- BitLocker Weakness
- Conclusion
No Encryption

Application → Operating System ← Hard Drive
Full Disk Encryption

Application → Operating System ← FDE → Hard Drive
Data on the Hard Drive

Four score and seven years ago our fathers brought forth on this continent, a new nation, conceived in Liberty and dedicated...
Without the Key
Searching for Keys in RAM
Targets

- Documented Open Source
  - TrueCrypt
- Undocumented Open Source
  - PGP Whole Disk Encryption
- Documented Closed Source
  - BitLocker Drive Encryption*
- Undocumented Closed Source
  - PointSec
  - Previously unseen tools
Current Methods

• Brute Force
  – Try every block of bytes as possible key
  – See "Linear Scan" paper by Hargreaves and Chivers
  – Doesn't work for split keys
Current Methods

• Key Schedule Search
  – Better brute force
  – Really identifying data that is not a key schedule
  – See "Cold Boot" paper by Halderman et al.
Current Methods

- Source code analysis
  - Requires elbow grease
  - Can't be automated
  - Works great
  - May have to update for each version
  - See "Volatools" paper by Walters and Petroni, BlackHat Federal 2007
Tool Marks

- Marks specific to individual tools
- Associated with physical forensics

Image courtesy Flickr user grendelkhan, http://flickr.com/photos/grendelkhan/118876699/
Tool Marks

- Were the screwdrivers found in the suspect's house used on the screws found on the bank vault?

Image courtesy Flickr user Uwe Hermann, http://flickr.com/photos/uwehermann/92145964/sizes/m/
Computer Forensics Tool Marks

• Anything detectable that software stores in RAM or on disk that identifies the tool in question
  – Most Recently Used lists
  – Header and footer carving
  – Registry keys left after program removed
  – Preferences files in user directories
  – Wiping programs leave traces behind
Cryptographic Tool Marks

- Hard to detect the keys
  - Small
  - Should be random
- Can detect the cryptographic tool itself
  - Programs
  - Drivers
  - Mounted volumes
- Can detect the structure surrounding the keys
BitLocker Drive Encryption

- Full Volume Encryption bundled with Windows Vista Ultimate
- Uses 128 bit AES-CBC + Elephant diffuser
  - Can configure for 256 bit and/or without diffuser
- Crypto developed by Niels Ferguson
  - also wrote Twofish, Helix, Fortuna RNG, CCM mode
  - Uses AES-CCM for key management
- Actual encryption work is done with 512 bit Full Volume Encryption Key (FVEK)
  - Key is 512 bits regardless of mode being used
BitLocker Drive Encryption

- I am not aware of any backdoors in BitLocker Drive Encryption
- You cannot access a protected volume without the FVEK

Image courtesy of the Microsoft Corporation.

BitLocker Drive Encryption is a registered trademark of the Microsoft Corporation.
BitLocker Drive Encryption

- Good documentation, but not complete
  - Key management systems not described
  - No implementation of elephant provided
- Reverse engineered by Kumar and Kumar
  - Published paper, linux driver to mount protected volumes
  - http://www.nvlabs.in/node/9
BitLocker Drive Encryption

- Brute Force works
  - FVEK is in RAM
- Key schedule search works
  - Finds several schedules
  - Two of the keys make up the FVEK
    - Some assembly required
- Source code analysis
  - Not an option for most of us
BitLocker Tool Marks

- BitLocker AES key schedules
  - Several schedules in memory at any given time
  - Some bits of FVEK used to generate sector keys
  - Other bits of FVEK used to encrypt/decrypt data
  - In default mode, some bits unused
BitLocker Tool Marks

- AES key schedules
  - Encryption and Decryption schedules
BitLocker Tool Marks

- Searching for BitLocker AES key schedules in RAM
  - Overlapped slightly
BitLocker Tool Marks

- 0x0 FVEc pool tag
- 0x14 Algorithm ID, must be 0x8000-0x8003
- 0x1C Start of first BitLocker AES schedule
  - AES key must be at start and end of schedule
  - bytes 0x1C-0x2C and 0x15C-0x16C
  - Zeros at end of schedule if 128-bit mode
- 0x1EC Start of second BitLocker AES schedule
  - Same rules as above
  - Normal 256-bit AES key schedules require 0x1E0 bytes
  - But overlapping saves 0x10 bytes
<table>
<thead>
<tr>
<th>Offset</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1EE5E000</td>
<td>00</td>
<td>00</td>
<td>7A</td>
<td>04</td>
<td>46</td>
<td>56</td>
<td>45</td>
<td>63</td>
<td>3C</td>
<td>FE</td>
<td>D8</td>
<td>83</td>
<td>E0</td>
<td>FE</td>
</tr>
<tr>
<td>1EE5E010</td>
<td>20</td>
<td>E0</td>
<td>E5</td>
<td>82</td>
<td>B0</td>
<td>03</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E020</td>
<td>20</td>
<td>84</td>
<td>01</td>
<td>68</td>
<td>02</td>
<td>D5</td>
<td>44</td>
<td>41</td>
<td>AB</td>
<td>E8</td>
<td>AA</td>
<td>47</td>
<td>5B</td>
<td>12</td>
</tr>
<tr>
<td>1EE5E030</td>
<td>5B</td>
<td>7E</td>
<td>62</td>
<td>57</td>
<td>4A</td>
<td>1F</td>
<td>5A</td>
<td>6A</td>
<td>1F</td>
<td>5A</td>
<td>6A</td>
<td>1F</td>
<td>5A</td>
<td>6A</td>
</tr>
<tr>
<td>1EE5E040</td>
<td>B9</td>
<td>7D</td>
<td>DD</td>
<td>F3</td>
<td>53</td>
<td>D7</td>
<td>7E</td>
<td>E1</td>
<td>AB</td>
<td>E8</td>
<td>AA</td>
<td>47</td>
<td>5B</td>
<td>12</td>
</tr>
<tr>
<td>1EE5E050</td>
<td>1B</td>
<td>D8</td>
<td>9A</td>
<td>C3</td>
<td>48</td>
<td>0E</td>
<td>E4</td>
<td>21</td>
<td>52</td>
<td>5E</td>
<td>8E</td>
<td>82</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E060</td>
<td>4B</td>
<td>CO</td>
<td>89</td>
<td>C3</td>
<td>03</td>
<td>CE</td>
<td>6D</td>
<td>E3</td>
<td>59</td>
<td>55</td>
<td>FE</td>
<td>77</td>
<td>08</td>
<td>0B</td>
</tr>
<tr>
<td>1EE5E070</td>
<td>70</td>
<td>91</td>
<td>0F</td>
<td>83</td>
<td>73</td>
<td>5F</td>
<td>02</td>
<td>0B</td>
<td>2A</td>
<td>0A</td>
<td>FC</td>
<td>7C</td>
<td>21</td>
<td>01</td>
</tr>
<tr>
<td>1EE5E080</td>
<td>72</td>
<td>84</td>
<td>11</td>
<td>58</td>
<td>E4</td>
<td>4A</td>
<td>42</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E090</td>
<td>05</td>
<td>E1</td>
<td>21</td>
<td>0A</td>
<td>67</td>
<td>A2</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E0A0</td>
<td>72</td>
<td>84</td>
<td>11</td>
<td>58</td>
<td>E4</td>
<td>4A</td>
<td>42</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E0B0</td>
<td>70</td>
<td>52</td>
<td>37</td>
<td>4E</td>
<td>03</td>
<td>7D</td>
<td>C7</td>
<td>E7</td>
<td>04</td>
<td>59</td>
<td>EA</td>
<td>07</td>
<td>23</td>
<td>D7</td>
</tr>
<tr>
<td>1EE5E0C0</td>
<td>48</td>
<td>B2</td>
<td>98</td>
<td>68</td>
<td>4B</td>
<td>CF</td>
<td>9E</td>
<td>8F</td>
<td>4F</td>
<td>96</td>
<td>74</td>
<td>88</td>
<td>6C</td>
<td>41</td>
</tr>
<tr>
<td>1EE5E0D0</td>
<td>85</td>
<td>5D</td>
<td>77</td>
<td>F7</td>
<td>5A</td>
<td>D4</td>
<td>22</td>
<td>F2</td>
<td>6C</td>
<td>00</td>
<td>F9</td>
<td>25</td>
<td>73</td>
<td>DE</td>
</tr>
<tr>
<td>1EE5E0E0</td>
<td>00</td>
<td>03</td>
<td>B7</td>
<td>58</td>
<td>DC</td>
<td>09</td>
<td>55</td>
<td>05</td>
<td>36</td>
<td>D4</td>
<td>DB</td>
<td>71</td>
<td>F2</td>
<td>DE</td>
</tr>
<tr>
<td>1EE5E0F0</td>
<td>22</td>
<td>BA</td>
<td>CB</td>
<td>16</td>
<td>DC</td>
<td>0A</td>
<td>E2</td>
<td>5E</td>
<td>EA</td>
<td>5D</td>
<td>8E</td>
<td>D2</td>
<td>29</td>
<td>0A</td>
</tr>
<tr>
<td>1EE5E100</td>
<td>1C</td>
<td>00</td>
<td>F0</td>
<td>EE</td>
<td>FE</td>
<td>BE</td>
<td>29</td>
<td>48</td>
<td>36</td>
<td>57</td>
<td>6C</td>
<td>8C</td>
<td>C3</td>
<td>57</td>
</tr>
<tr>
<td>1EE5E110</td>
<td>AE</td>
<td>E2</td>
<td>7D</td>
<td>47</td>
<td>E2</td>
<td>B0</td>
<td>D9</td>
<td>A6</td>
<td>C8</td>
<td>E9</td>
<td>45</td>
<td>C4</td>
<td>F5</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E120</td>
<td>F0</td>
<td>CD</td>
<td>AB</td>
<td>57</td>
<td>4C</td>
<td>4A</td>
<td>A4</td>
<td>E1</td>
<td>2A</td>
<td>51</td>
<td>9C</td>
<td>D2</td>
<td>3D</td>
<td>E9</td>
</tr>
<tr>
<td>1EE5E130</td>
<td>99</td>
<td>18</td>
<td>35</td>
<td>2F</td>
<td>BC</td>
<td>87</td>
<td>0F</td>
<td>B6</td>
<td>66</td>
<td>1B</td>
<td>38</td>
<td>83</td>
<td>17</td>
<td>B8</td>
</tr>
<tr>
<td>1EE5E140</td>
<td>5A</td>
<td>D3</td>
<td>F1</td>
<td>92</td>
<td>25</td>
<td>9F</td>
<td>3A</td>
<td>99</td>
<td>DA</td>
<td>9C</td>
<td>37</td>
<td>35</td>
<td>71</td>
<td>A3</td>
</tr>
<tr>
<td>1EE5E150</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E160</td>
<td>20</td>
<td>84</td>
<td>01</td>
<td>68</td>
<td>02</td>
<td>DS</td>
<td>44</td>
<td>41</td>
<td>AB</td>
<td>E8</td>
<td>AA</td>
<td>47</td>
<td>5B</td>
<td>12</td>
</tr>
<tr>
<td>1EE5E170</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E180</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E190</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E1A0</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E1B0</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E1C0</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E1D0</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>1EE5E1E0</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
</tbody>
</table>

Pool Tag: FVE
Algorithm: Key Schedule
Start of AES key schedule: Offset 1EE5E000
Zeros: Offset 1EE5E0A0
BitLocker Tool Marks

- Not perfect, but good enough
- Original

<table>
<thead>
<tr>
<th>Offset</th>
<th>0 1 2 3 4 5 6 7 8 9 A B C D E F</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000000</td>
<td>3F 26 C8 B5 FF 87 47 B1 D5 26 12 43 EC CD 78 C6</td>
</tr>
<tr>
<td>00000010</td>
<td>D5 09 AF 19 D1 5A 10 03 B5 4D 1B 73 0E EC 0A 93</td>
</tr>
<tr>
<td>00000020</td>
<td>7A 16 05 EB 54 9F 39 1C 2E 5D 6A DB BC 67 C2 36</td>
</tr>
<tr>
<td>00000030</td>
<td>1C 3D F3 60 AF A1 EB 6F E4 47 B2 E3 A5 B5 38 D9</td>
</tr>
</tbody>
</table>

- Recovered

<table>
<thead>
<tr>
<th>Offset</th>
<th>0 1 2 3 4 5 6 7 8 9 A B C D E F</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000000</td>
<td>3F 26 C8 B5 FF 87 47 B1 D5 26 12 43 EC CD 78 C6</td>
</tr>
<tr>
<td>00000010</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
<tr>
<td>00000020</td>
<td>7A 16 05 EB 54 9F 39 1C 2E 5D 6A DB BC 67 C2 36</td>
</tr>
<tr>
<td>00000030</td>
<td>00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00</td>
</tr>
</tbody>
</table>
Finding Tool Marks

- Perl Script
  - It’s not pretty, but it works
- Volatility Suite
  - Supposed to be for Windows XP SP2 only
  - But can treat any file as a flat file
  - Use the Sliding Window Scanner
  - If/When support is added for Vista,
    - Use Pool Tag Scanner
Finding Tool Marks

• How did we do this?
  – RTFM
    • FIPS certifications are great!
    • Ask developers for help
  – WinHex
  – IDA Pro
  – Checked builds
  – Debugging symbols
• Always trying to answer:
  – How does it know where to look?

Image courtesy of User:Icey on Wikipedia and is public domain

ManTech
International Corporation
Performance

- Brute Force
  - $O(nm)$
- Key Schedule Search
  - $O(nm)$
- Source Code
  - $X^* + O(n)$, where $X^*$ may be infinite
- Toolmarks
  - $X + O(n)$
Forensics Tool Marks

• Requires as much elbow grease as source code analysis
  – Often more
  – Doesn't require the source code
• May require updating for each version
  – TrueCrypt
• May be your only option for previously unseen tools
BitLocker Drive Encryption

- I am not aware of any backdoors in BitLocker Drive Encryption
- You cannot access a protected volume without the FVEK

Image courtesy of the Microsoft Corporation.

BitLocker Drive Encryption is a registered trademark of the Microsoft Corporation.
A Series of Keys

- Full Volume Encryption Key (FVEK)
  - Does actual encryption/decryption
  - Never changes
- Volume Master Key (VMK)
  - Used to encrypt FVEK
  - Never changes
- Various Other Keys
  - TPM key
  - External Keys (USB sticks)
  - Recovery Password
- Each used to decrypt their copy of the VMK
BitLocker Metatadada

- Contains $E(FVEK, VMK)$
  - FVEK encrypted with VMK
- Metadata entries for each key
  - $E(VMK, TPM\ key)$
  - $E(VMK, External\ key)$
  - $E(VMK, Recovery\ key)$
A Series of Keys

- Each entry also contains key encrypted with VMK

- Metadata entries for each key
  - $E(\text{VMK, TPM key})$
  - $E(\text{TPM key, VMK})$
  - $E(\text{VMK, External key})$
  - $E(\text{External key, VMK})$
  - $E(\text{VMK, Recovery key})$
  - $E(\text{Recovery key, VMK})$
Scenario

- Legitimate user has External Key
  - USB token
- System administrator has recovery password

- Legitimate user uses external key to decrypt VMK
- Uses VMK to decrypt the other keys
  - Gets recovery password

- Legitimate access revoked
- Can still access system using recovery password!
Exploit Scenario

- Yes, it’s unlikely
  - But crypto people live for the unlikely
- Has been reported to Microsoft
  - No response
- Full details in “Implementing BitLocker Drive Encryption for Forensic Analysis” to be published in Digital Investigation
Outline

• Introduction
• Types of Targets
• Finding Keys
• Tool Marks
• Example - BitLocker
• BitLocker Weakness
• Conclusion
Questions?

Jesse Kornblum
jesse.kornblum@mantech.com
http://jessekornblum.com/
http://mantech.com/

Image courtesy of Flickr user demosh, http://flickr.com/photos/44222307@N00/1477086299/
Thank you

- ManTech International Corporation for letting me geek out
- Microsoft Corporation for keeping me employed
- Kumar and Kumar for their reverse engineering work
- You for hearing this talk
- Slides are posted on http://jessekornblum.com/

Image courtesy of the Microsoft Corporation.
References

- BitLocker Drive Encryption
  - N. Kumar and V. Kumar, "Bitlocker and Windows Vista",
    http://www.nvlabs.in/node/9
  - FIPS Security Policy:
    http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp947.pdf

- Brute Force Searches
  - C. Hargeaves and H. Chivers, "Recovery of Encryption Keys from Memory Using a Linear Scan",
References

• Key Schedule Searching

• Source Code Analysis