# XX TCG MOR Test

### XX.1TCG MOR Test

Reference Document:

*TCG PC Platform Reset Attack Mitigation Specification*,v1.10 Revision 17, January 21, 2019, Chapter 4.

These tests support platform firmware that implements the MemoryOverwriteRequestControl and MemoryOverwriteRequestControlLock UEFI variables in accordance with TCG PC Platform Reset Attack Mitigation Specification, v1.10.

### XX.1.1 MemoryOverwriteRequestControl Platform Reset Check

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
|  |  | Verify MemoryOverwriteRequestControl is created by platform firmware as specified in the tcg spec. | 1. Reboot System.  2. Verify GetVariable() returns MemoryOverwriteRequestControl with correct attributes and DataSize.  If MemoryOverwriteRequestControl does not exist, the test should exit with an exit code that notifies user TCG MOR is not enabled on platform.  If MemoryOverwriteRequestControl exists with the incorrect attributes or DataSize, the test should exit with an exit code that notifies the user the platform has not created the variable as defined in the spec upon reboot. |
|  |  | Verify that setting MemoryOverwriteAction\_BitValue within MemoryOverwriteRequestControl clears the MemoryOverwriteAction\_BitValue on system reset. | 1. Invoke SetVariable() with valid GUID, variable name, attributes, and DataSize and Set MemoryOverwriteAction\_BitValue Bit 0.  2. Reboot System.  3. Invoke GetVariable() with valid GUID, variable name, attributes, and DataSize  4. Verify MemoryOverwriteAction\_BitValue Bit 0 is Cleared. |
|  |  | \*\* Reviewer Note: we may need an additional test for checking that the platform actually wiped system reset \*\* |  |

### XX.1.2 MemoryOverwriteRequestControl.SetVariable()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
|  | 0x7a1e79a3, 0x4064, 0x4372, 0xbb, 0x64,0x55, 0xb8, 0xf2, 0xa5, 0xa3, 0x26 | MemoryOverwriteRequestControl.SetVariable() returns EFI\_INVALID\_PARAMETER by passing in all valid parameters excluding DataSize = 0.  The state of MemoryOverwriteRequestControl shall not change. | Save the state of GetVariable() with valid GUID, variable name, and attributes.  Invoke SetVariable() with valid GUID, variable name, and attributes. DataSize set to 0 should return EFI\_INVALID\_PARAMETER.  a. Verify that GetVariable().attributes == SetVariable() attributes being passed.  b. Verify SetVariable() returns EFI\_INVALID\_PARAMETER.  c. Verify that GetVariable returns the same state as the saved initial state. |
|  | 0xb0e717c4, 0xb1e2, 0x49f7, 0xb2, 0xd7,0x60, 0x58,0x97, 0x7d, 0x09, 0x2c | MemoryOverwriteRequestControl.SetVariable() returns EFI\_INVALID\_PARAMETER by passing in all valid parameters excluding incorrect attributes.  The state of MemoryOverwriteRequestControl shall not change. | Save the state of GetVariable() with valid GUID, variable name, and attributes.  Invoke SetVariable() with valid GUID, variable name, and DataSize.  Pass an attributes value that does not match the attributes returned from GetVariable().  a. Verify that GetVariable().attributes != SetVariable() attributes being passed.  b. Verify SetVariable() returns EFI\_INVALID\_PARAMETER.  c. Verify that GetVariable() returns the same state as the saved initial state. |

### XX.1.3 MemoryOverwriteRequestControlLock.SetVariable()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
|  | 0xa8e1b5e6, 0xfc09, 0x461c, 0xb0, 0xe9, 0x2a, 0x49, 0xcd, 0x25, 0xc1, 0x24 | MemoryOverwriteRequestControlLock.SetVariable() passing valid parameters excluding Attributes == 0 must return EFI\_WRITE\_PROTECTED. | Invoke SetVariable() with valid GUID, variable name, Data, and DataSize. Set Attributes == 0.  Verify SetVariable() returns EFI\_WRITE\_PROTECTED. |
|  | 0x26f04a9b, 0x7b7a, 0x4f47, 0xbe, 0xa8, 0xb1, 0xa6, 0x02, 0x65, 0x19, 0x8a | MemoryOverwriteRequestControlLock.SetVariable() passing valid parameters excluding DataSize == 0 must return EFI\_WRITE\_PROTECTED. | Invoke SetVariable() with valid GUID, variable name, Data, and Attributes. Set DataSize== 0.  Verify SetVariable() returns EFI\_WRITE\_PROTECTED. |
|  | 0x4d1d9985, 0x91e2, 0x4948, 0x89, 0x16, 0xbb, 0x98, 0x13, 0x62, 0x39, 0x1d | MemoryOverwriteRequestControlLock.SetVariable() passing valid parameters excluding Data == NULL must return EFI\_WRITE\_PROTECTED. | Invoke SetVariable() with valid GUID, variable name, Attributes, and DataSize. Set Data == NULL.  Verify SetVariable() returns EFI\_WRITE\_PROTECTED. |
|  | 0xfb59cab7, 0x4f8c, 0x4ded, 0xa4, 0x1c, 0xc8, 0x41, 0x20, 0x1c, 0x37, 0x22 | MemoryOverwriteRequestControlLock.SetVariable() passing valid parameters excluding Attributes != 7 must return EFI\_INVALID\_PARAMETER. | Invoke SetVariable() with valid GUID, variable name, Data, and DataSize. Set Attributes != 7.  Verify SetVariable() returns  EFI\_INVALID\_PARAMETER. |
|  | 0x0363d22f, 0xc66a, 0x4872, 0xa5, 0x46, 0x06, 0x7f, 0x6a, 0x0d, 0xdb, 0xcd | MemoryOverwriteRequestControlLock.SetVariable() passing valid parameters excluding (DataSize != 0 && DataSize != 8) must return EFI\_INVALID\_PARAMETER | Invoke SetVariable() with valid GUID, variable name, Data, and Attributes. Set DataSize != 0 && DataSize != 8.  Verify SetVariable() returns EFI\_INVALID\_PARAMETER. |

### XX.1.4 MemoryOverwriteRequestControlLock Unlocked state

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
|  |  | Changing MemoryOverwriteRequestControlLock in Unlocked state  Should return an output value of 0. | 1. Reboot System to make firmware initialize MemoryOverwriteRequestControlLock to Unlocked. 2. Invoke GetVariable() with valid GUID, variable name, attributes, DataSize, and Data. 3. Verify that the output value is 0. 4. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize = 1, and Data Input = 0. 5. Verify Lock.SetVariable() returns EFI\_SUCCESS. |
|  |  | MemoryOverwriteRequestControlLock in Unlocked state with an invalid first byte for Data should return EFI\_INVALID\_PARAMETER. | 1. Verify MemoryOverwriteRequestControlLock is in the Unlocked state. 2. Invoke SetVariable() with valid GUID, variable name, attributes, DataSize = 1, Data first byte != 0 && Data first byte != 1. 3. Verify SetVariable() returns EFI\_INVALID\_PARAMETER |

### XX.1.5 MemoryOverwriteRequestControlLock Locked w/o key state

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
|  |  | Changing MemoryOverwritRequestControlLock to Locked state without key Should return EFI\_SUCCESS when MemoryOverwriteRequestControlLock is currently unlocked. | 1. Reboot System to make firmware initialize MemoryOverwriteRequestControlLock to Unlocked. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 1. 3. Verify SetVariable() returns EFI\_SUCCESS. 4. Verify GetVariable() returns output value 1. |
|  |  | Changing MemoryOverwriteRequestControlLock to Unlocked state Should return  EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock is currently Locked without key. | 1. Verify MemoryOverwriteRequestControlLock is currently locked without key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 0. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 1. |
|  |  | Changing MemoryOverwriteRequestControlLock to Locked without key state Should return EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock is currently Locked without key. | 1. Verify MemoryOverwriteRequestControlLock is currently locked without key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 1. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 1. |
|  |  | Changing MemoryOverwriteRequestControlLock to Locked state with key  Should return EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock was already Locked without key. | 1. Verify MemoryOverwriteRequestControlLock is currently locked without key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 8-byte randomgenkey. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 1. |

### XX.1.6 MemoryOverwriteRequestControlLock Locked state with key

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
|  |  | Changing MemoryOverwriteRequestControlLock to Locked state with a key  must return EFI\_SUCCESS when MemoryOverwriteRequestControlLock is currently unlocked.    MemoryOverwriteRequestControlLock must be locked with key. | 1. Reboot System to make firmware initialize MemoryOverwriteRequestControlLock to Unlocked. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 8-byte randomgenkey. 3. Verify SetVariable() returns EFI\_SUCCESS. 4. Verify GetVariable() returns output value 2. |
|  |  | Changing MemoryOverwriteRequestControlLock to Unlocked state with the 8 byte key used in the previous SetVariable() Lock with key action  must return EFI\_SUCCESS when MemoryOverwriteRequestControlLock is currently Locked with key.    MemoryOverwriteRequestControlLock must be unlocked. | 1. Verify MemoryOverwriteRequestControlLock is currently locked with 8-byte generated key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 8-byte randomgenkey. 3. Verify SetVariable() returns EFI\_SUCCESS. 4. Verify GetVariable() returns output value 0. |
|  |  | Changing MemoryOverwriteRequestControlLock to Unlocked state with an empty key must return EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock is currently Locked with key.    MemoryOverwriteRequestControlLock must not change. | 1. Verify MemoryOverwriteRequestControlLock is currently locked with 8-byte generated key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and empty key. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 2. |
|  |  | Changing MemoryOverwriteRequestControlLock to Unlocked state with an invalid DataSize must return EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock is currently Locked with key.    MemoryOverwriteRequestControlLock must not change. | 1. Verify MemoryOverwriteRequestControlLock is currently locked with 8-byte generated key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, and DataSize != 8. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 2. |
|  |  | Changing MemoryOverwriteRequestControlLock to Locked without key state must return EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock is currently Locked with key.    MemoryOverwriteRequestControlLock must not change. | 1. Verify MemoryOverwriteRequestControlLock is currently locked with 8-byte generated key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 1. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 2. |
|  |  | Changing MemoryOverwriteRequestControlLock to Unlocked state with an 8 byte value that is not equal to the 8-byte key used in the previous SetVariable() Lock with key action  must return EFI\_ACCESS\_DENIED when MemoryOverwriteRequestControlLock is currently Locked with key. | 1. Verify MemoryOverwriteRequestControlLock is currently locked locked with 8-byte generated key. 2. Invoke SetVariable() with Valid GUID, variable name, attributes, DataSize, and Data Input = 8 byte value != 8-byte generated key. 3. Verify SetVariable() returns EFI\_ACCESS\_DENIED. 4. Verify GetVariable() returns output value 1. |