**# Title**: Memory Attribute Protocol

**# Status**: Submitted to industry standard forum

**# Document**: UEFI Specification Version 2.9

**# License**: SPDX-License-Identifier: CC-BY-4.0

**# Submitter**: [TianoCore Community](<https://www.tianocore.org>)

**# Summary of the change**

**[Background]**

In current PI specification, the GCD service and CPU\_ARCH protocol provide API to change the memory attribute. However, there is no such service in UEFI specification. In UEFI security sub team meeting, people believe that there is value to expose such API for the OS loader. As such, the OS loader may setup proper memory protection.

**[Proposal]**

This proposal adds a new protocol - **EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL**. This protocol abstracts the memory attributes setting, such as READ\_ONLY for CODE, and NON\_EXECUTABLE for DATA.

It is similar to the **DXE\_SERVICES.SetMemorySpaceAttributes()** or **EFI\_CPU\_ARCH\_PROTOCOL**.**SetMemorySpaceAttributes()**.

It is same as the EDKII – **EDKII\_SMM\_MEMORY\_ATTRIBUTE\_PROTOCOL** (<https://github.com/tianocore/edk2/blob/master/MdeModulePkg/Include/Protocol/SmmMemoryAttribute.h>), which is used to manage the page table based memory attributes in SMM.

Reference:

1. UEFI Specification 2.9 - www.uefi.org
2. PI Specification 1.7 - www.uefi.org
3. EDKII - https://github.com/tianocore/edk2

**# Benefits of the change**

The OS loader may use this protocol to set the page table based memory protection.

**# Impact of the change**

This is an optional protocol.

# Detailed description of the change [normative updates]

**UEFI Specification**

37.7 Memory Protection   
**EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL**

**Summary**

This protocol abstracts the memory attributes setting or getting in UEFI environment.

**GUID**

**#define EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL\_GUID \**

**{0xf4560cf6, 0x40ec, 0x4b4a, 0xa1, 0x92, 0xbf, 0x1d, 0x57, 0xd0, 0xb1, 0x89}**

**Protocol Interface Structure**

**typedef struct \_EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL {**

**EFI\_GET\_MEMORY\_ATTRIBUTES GetMemoryAttributes;**

**EFI\_SET\_MEMORY\_ATTRIBUTES SetMemoryAttributes;**

**EFI\_CLEAR\_MEMORY\_ATTRIBUTES ClearMemoryAttributes;**

**} EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL;**

**Parameters**

*GetMemoryAttributes*

Get the memory attributes for a memory region. See the **GetMemoryAttrbutes()** function description.

*SetMemoryAttributes*

Set the memory attributes for a memory region. See the **SetMemoryAttrbutes()** function description.

*ClearMemoryAttributes*

Clear the memory attributes for a memory region. See the **ClearMemoryAttrbutes()** function description.

**Description**

The **EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL** is used to abstract memory attribute access for a memory region.

**EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL.GetMemoryAttributes**

**Summary**

This service retrieves the memory attributes of a memory region.

**Prototype**

**typedef**

**EFI\_STATUS**

**(EFIAPI \*EFI\_GET\_MEMORY\_ATTRIBUTES)(**

**IN EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL  \*This,**

**IN EFI\_PHYSICAL\_ADDRESS         BaseAddress,**

**IN UINT64          Length,**

**OUT UINT64          \*Attributes**

**);**

**Parameters**

*This*

The protocol interface pointer.

*BaseAddress*

The physical address that is the start address of a memory region.

*Length*

The size in bytes of the memory region.

*Attributes*

Pointer to the bit mask of memory attributes returned. The bit mask of attributes is defined in the **GetMemoryMap()**function description in the UEFI 2.0 specification*.* The valid *Attributes* is **EFI\_MEMORY\_RP**, **EFI\_MEMORY\_XP**, and **EFI\_MEMORY\_RO**.

**Description**

This function retrieves the attributes of the memory region specified by BaseAddress and Length. If different attributes are got from different part of the memory region, EFI\_NO\_MAPPING will be returned.

**Status Code Returned**

|  |  |
| --- | --- |
| EFI\_SUCCESS | The attributes are got for the memory region. |
| EFI\_INVALID\_PARAMETER | *Length* is zero.  *Attributes* is NULL. |
| EFI\_NO\_MAPPING | Attributes are not consistent cross the memory region. |
| EFI\_UNSUPPORTED | The processor does not support one or more bytes of the memory resource range specified by *BaseAddress* and *Length*. |

**EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL.SetMemoryAttributes**

**Summary**

This service sets the memory attributes of a memory region.

**Prototype**

**typedef**

**EFI\_STATUS**

**(EFIAPI \*EFI\_SET\_MEMORY\_ATTRIBUTES)(**

**IN EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL  \*This,**

**IN EFI\_PHYSICAL\_ADDRESS         BaseAddress,**

**IN UINT64          Length,**

**IN UINT64          Attributes**

**);**

**Parameters**

*This*

The protocol interface pointer.

*BaseAddress*

The physical address that is the start address of a memory region.

*Length*

The size in bytes of the memory region.

*Attributes*

The bit mask of memory attributes. The bit mask of attributes is defined in the **GetMemoryMap()**function description in the UEFI 2.0 specification*.* The valid *Attributes* is **EFI\_MEMORY\_RP**, **EFI\_MEMORY\_XP**, and **EFI\_MEMORY\_RO**.

**Description**

This function sets the given attributes of the memory region specified by *BaseAddress* and *Length*. This function only adds new attributes with the bit mask 1 and ignores the attributes with the bit mask 0.

**Status Code Returned**

|  |  |
| --- | --- |
| EFI\_SUCCESS | The attributes are set for the memory region. |
| EFI\_INVALID\_PARAMETER | *Length* is zero.  *Attributes* specified an illegal combination of attributes that cannot be set together. |
| EFI\_UNSUPPORTED | The processor does not support one or more bytes of the memory resource range specified by *BaseAddress* and *Length*.  The bit mask of attributes is not supported for the memory resource range specified by *BaseAddress* and *Length*. |
| EFI\_OUT\_OF\_RESOURCES | Requested attributes cannot be applied due to luck of system resources. |
| EFI\_ACCESS\_DENIED | Attributes for the requested memory region are controlled by system firmware and cannot be updated via the protocol. |

**EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL.ClearMemoryAttributes**

**Summary**

This service clears the memory attributes of a memory region.

**Prototype**

**typedef**

**EFI\_STATUS**

**(EFIAPI \*EFI\_CLEAR\_MEMORY\_ATTRIBUTES)(**

**IN EFI\_MEMORY\_ATTRIBUTE\_PROTOCOL  \*This,**

**IN EFI\_PHYSICAL\_ADDRESS         BaseAddress,**

**IN UINT64          Length,**

**IN UINT64          Attributes**

**);**

**Parameters**

*This*

The protocol interface pointer.

*BaseAddress*

The physical address that is the start address of a memory region.

*Length*

The size in bytes of the memory region.

*Attributes*

The bit mask of memory attributes. The bit mask of attributes is defined in the **GetMemoryMap()**function description in the UEFI 2.0 specification*.* The valid *Attributes* is **EFI\_MEMORY\_RP**, **EFI\_MEMORY\_XP**, and **EFI\_MEMORY\_RO**.

**Description**

This function clears the given attributes of the memory region specified by *BaseAddress* and *Length*. This function only removes new attributes with the bit mask 1 and ignore the attributes with the bit mask 0.

**Status Code Returned**

|  |  |
| --- | --- |
| EFI\_SUCCESS | The attributes are cleared for the memory region. |
| EFI\_INVALID\_PARAMETER | *Length* is zero.  *Attributes* specified an illegal combination of attributes that cannot be cleared together. |
| EFI\_UNSUPPORTED | The processor does not support one or more bytes of the memory resource range specified by *BaseAddress* and *Length*.  The bit mask of attributes is not supported for the memory resource range specified by *BaseAddress* and *Length*. |
| EFI\_OUT\_OF\_RESOURCES | Requested attributes cannot be applied due to luck of system resources. |
| EFI\_ACCESS\_DENIED | Attributes for the requested memory region are controlled by system firmware and cannot be updated via the protocol. |

**# Special Instructions**

NO