# Network Protocols Managed Network

## EFI\_MANAGED\_NETWORK\_PROTOCOL Test

Reference Document:

*UEFI 2.0 Specification*, Section 21.

### GetModeData()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.1.1 | 0xfd5600b1, 0x958d, 0x4cf3, 0x9a, 0x6a, 0xb4, 0x5e, 0x26, 0x73, 0x19, 0xc6 | EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData **–** invokes GetModeData() with a *MnpConfigData* value other than NULL when the MNP child has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**GetModeData() with a *MnpConfigData* value other than NULL when the MNP child has not been configured. The return status should be **EFI\_NOT\_STARTED**, and the default values are returned in*MnpConfigData*.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.1.2 | 0xf39fc5b4, 0xcea9, 0x498d, 0xb7, 0xe4, 0xce, 0x0a, 0x7c, 0x9e, 0x0b, 0x35 | EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData – invokes GetModeData() to get the previously configured data. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameter for the child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**GetModeData() to get the previously configured data in step 2,  4. Verify the data. The return status should be EFI\_SUCCESS.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.1.3 | 0x5b579cdd, 0xae9b, 0x4415, 0xbd, 0xc0, 0x39, 0xb0, 0x14, 0xcf, 0x29, 0xe2 | EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData **–** invokes GetModeData() with a MnpConfData value of NULL and a SnpModeData value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameter for the child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData() with a MnpConfData value of NULL and a SnpModeDatavalue ofNULL. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.1.4 | 0xd34ce9f5, 0x8fb5, 0x4f50, 0xac, 0x68, 0x64, 0x0e, 0xc9, 0x3b, 0xc0, 0xbf | EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData – invokes GetModeData() with a MnpConfData value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameter for the child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData() with a MnpConfDatavalue ofNULL. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.1.5 | 0xbde40b90, 0xf94f, 0x4c26, 0xac, 0x32, 0x21, 0x07, 0xa4, 0x19, 0x82, 0xde | EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData – invokes GetModeData() with a SnpModeDat*a* value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameter for the child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.GetModeData() with a SnpModeDatavalue of NULL. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### Configure()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.2.1 | 0x4c4b70cd, 0x5492, 0x440f, 0x87, 0xd8, 0xc8, 0x4d, 0x0b, 0x61, 0x02, 0x9f | EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure – invokes Configure() with an invalid *MnpConfigData.ProtocolTypeFilter* value. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() with an invalid MnpConfigData.ProtocolTypeFilter value. The return status should be EFI\_INVALID\_PARAMETER.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.2.2 | 0x437bdc0d, 0xe159, 0x4535, 0x92, 0xe0, 0x56, 0x59, 0xd7, 0xa4, 0xc7, 0xfc | EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure – invokes Configure() after creating a new MNP child. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child. The return status should be EFI\_SUCCESS.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.2.3 | 0x3d69e8d4, 0x34fa, 0x4a15, 0xaa, 0xb1, 0x95, 0x48, 0x13, 0x9a, 0x62, 0x59 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure – invokes Configure() with unicast and broadcast disabled, which means set the parameter *EnableUnicastReceive* and *EnableBroadcaseReceive* set to FALSE**.** | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() with the parameter *EnableUnicastReceive* and *EnableBroadcaseReceive* a set to FALSE. The return status should be EFI\_SUCCESS.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.2.4 | 0x5e075f02, 0x708d, 0x4c3d, 0x8e, 0xc6, 0x53, 0x91, 0x6c, 0x30, 0xf4, 0x2b | EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure – invokes Configure() when the configuration data is reset to NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() when the configuration data is reset to NULL. The return status should be EFI\_SUCCESS.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.2.5 | 0xfbbaf8a7, 0x91ac, 0x497a, 0x9f, 0x9d, 0xec, 0x0a, 0x35, 0x34, 0xa1, 0xd7 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure – invokes Configure() when *ReceiveQueueTimeout* is enabled. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() when *ReceiveQueueTimeout* is enabled. The return status should be EFI\_SUCCESS.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### McastlpToMac()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.3.1 | 0x5902f01b, 0x124a, 0x4fe9, 0x98, 0xfa, 0x07, 0x97, 0x71, 0x4b, 0x39, 0xc3 | EFI\_MANAGED\_NETWORK\_PROTOCOL. McastlpToMac – invokes McastlpToMac() when the child has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.McastlpToMac() when the child has not been configured. The return status should be EFI\_NOT\_STARTED.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.3.2 | 0x0b2990e3, 0xc947, 0x4121, 0xb8, 0xa5, 0x9c, 0x47, 0x7b, 0xac, 0x28, 0xf7 | EFI\_MANAGED\_NETWORK\_PROTOCOL. McastlpToMac – invokes McastlpToMac() with an *IpAddress* value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.McastlpToMac() with an IpAddress value of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.3.3 | 0x0227a52e, 0x22b9, 0x4c6a, 0x8e, 0x13, 0x06, 0x62, 0x4c, 0x92, 0x39, 0x7f | EFI\_MANAGED\_NETWORK\_PROTOCOL. McastlpToMac – invokes McastlpToMac() with an *IpAddress* value that is an invalid multicast IP address. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.McastlpToMac() with an *IpAddress* value that is an invalid multicast IP address. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.3.4 | 0x318eae7a, 0xa94d, 0x4eec, 0xbf, 0xde, 0x4e, 0x04, 0x04, 0xe3, 0x2c, 0x34 | EFI\_MANAGED\_NETWORK\_PROTOCOL. McastlpToMac – invokes McastlpToMac() with a *MacAddress* value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.McastlpToMac() with a *MacAddress* value of NULL.The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.3.5 | 0x8571d2b8, 0xe8e9, 0x450a, 0x84, 0x58, 0xf8, 0xb4, 0xa4, 0xa4, 0xc6, 0x5d | EFI\_MANAGED\_NETWORK\_PROTOCOL. McastlpToMac – invokes McastlpToMac() with the parameter *Ipv6Flag* set to TRUE. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.McastlpToMac() with the parameter *Ipv6Flag* set to TRUE.The return status should be EFI\_UNSUPPORTED.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.3.6 | 0xa6a2d468, 0x07b3, 0x47d7, 0x82, 0xec, 0x76, 0x85, 0x92, 0x6a, 0x78, 0x09 | EFI\_MANAGED\_NETWORK\_PROTOCOL. McastlpToMac – invokes McastlpToMac() to change multicast IPv4 address to MAC. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.McastlpToMac() to change multicast IPv4 address to MAC. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### Groups()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.4.1 | 0xdae4ffb7, 0x4cc2, 0x4d04, 0xbe, 0x90, 0xef, 0xd1, 0x9e, 0x62, 0x94, 0xd8 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() when the child has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups() when the child has not been configured. The return status should be EFI\_NOT\_STARTED.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.2 | 0x60fffa21, 0x3c10, 0x427a, 0xaf, 0x6e, 0xee, 0x78, 0x39, 0x14, 0xc5, 0xbe | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() with the parameter JoinFlag set to TRUE and a MacAddress value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups() with the parameter *JoinFlag* set to TRUE and a *MacAddress* value of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.3 | 0x8e49561e, 0x667b, 0x4da2, 0xae, 0x57, 0xa3, 0x51, 0x07, 0xaa, 0xb0, 0xce | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() with a *\*MacAddress* value that is an invalid multicast MAC address. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups() with a *\*MacAddress* value that is an invalid multicast MAC address. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.4 | 0xbf473ce1, 0x8bf5, 0x4386, 0x81, 0x3b, 0x73, 0x34, 0xff, 0xc1, 0x8b, 0xb2 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() when the supplied multicast group has already been joined. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups() to join a multicast group. The return status should be EFI\_SUCCESS.  4. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups() to join the same multicast group joined in step 3. The return status should be EFI\_ALREADY\_STARTED.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.5 | 0x0ea6fd9b, 0xb4d3, 0x46d0, 0xa9, 0xb5, 0xe3, 0x41, 0x8f, 0x76, 0x59, 0x9e | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() to remove a multicast group that has not been joined. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Groups() to remove a multicast group that has not been joined. The return status should be **EFI\_NOT\_FOUND.**  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.6 | 0x10e81796, 0x75df, 0x4998, 0x95, 0x3b, 0xf6, 0x6a, 0x73, 0x65, 0xa6, 0xdf | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() to join a multicast group. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups ()** to join a multicast group. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.7 | 0x86d023ea, 0xcd2a, 0x4641, 0x82, 0x38, 0x19, 0x4c, 0x5e, 0x1c, 0x72, 0x07 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() to delete a multicast group. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Groups() to delete the multicast group. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.4.8 | 0x28419ce8, 0xe2d3, 0x4434, 0x90, 0xd3, 0xc2, 0xe3, 0xb5, 0x34, 0x50, 0x52 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Groups – invokes Groups() to delete all groups. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new child.  3. Call Groups() to delete all groups. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### Transmit()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.5.1 | 0x5ae0ea70, 0x50d7, 0x49ab, 0xb7, 0x78, 0xb9, 0x12, 0xa9, 0xab, 0x5b, 0x91 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with a Token value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() with a Token value of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.2 | 0x254e59ae, 0x6184, 0x4885, 0x84, 0x9d, 0xd9, 0x96, 0x75, 0x12, 0xd2, 0x5f | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with a Token.Event value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() with a Token.Event value of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.3 | 0xbcf56099, 0x84e9, 0x464b, 0xb8, 0x50, 0x64, 0x26, 0x5f, 0x91, 0x69, 0x6b | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with a *TxData.FragmentCount* value of 0. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() with a *TxData.FragmentCount* value of **0**. The rerurn status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.4 | 0x8612aa9b, 0x2c0d, 0x4512, 0xbf, 0xf9, 0xfd, 0x70, 0xae, 0x62, 0xaf, 0xfa | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() when *(*Token*.TxData.HeaderLength +* Token*.TxData.DataLength)* is not equal to the sum of the Token*.TxData.*FragmentTable*[].FragmentLength* fields. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() when *(*Token*.TxData.HeaderLength +* Token*.TxData.DataLength)* is not equal to the sum of the Token*.TxData.*FragmentTable*[].FragmentLength* fields. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.5 | 0xab47d163, 0x05ef, 0x4aac, 0xaa, 0x45, 0xae, 0x93, 0x8e, 0xf8, 0x25, 0x95 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with one or more Token.TxData.FragmentTable[].FragmentLength fields with values of **0**. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() with one or more Token.TxData.FragmentTable[].FragmentLength fields with values of **0**. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.6 | 0x8030770d, 0x056a, 0x4780, 0x98, 0xbe, 0xef, 0x85, 0x46, 0x7f, 0xb2, 0xec | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with one or more Token.TxData.FragmentTable[].FragmentBuffer fields with values of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() with one or more Token.TxData.FragmentTable[].FragmentBuffer fields with values of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.7 | 0xcd7bf7fb, 0xf3be, 0x4cd7, 0x8a, 0xc3, 0x50, 0x2d, 0xca, 0xe5, 0xcc, 0x5a | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() when the MNP child driver instance has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit() when the MNP child driver instance has has not been configured. The return status should be EFI\_NOT\_STARTED.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.8 | 0x5f54752c, 0xa297, 0x4609, 0x9b, 0x4b, 0x44, 0x77, 0x45, 0x04, 0x18, 0x2d | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with transmit specified data to check the correction of data transmission. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Transmit() with transmit data specified. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.5.9 | 0x54a2a21b, 0x9acf, 0x4f61, 0x9c, 0xc1, 0x8e, 0x31, 0xa8, 0x3e, 0x9e, 0xc4 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Transmit – invokes Transmit() with transmit data not specified. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Transmit() with transmit data not specified. The return status should be EFI\_SUCCESS.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### Receive()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.6.1 | 0xf88f8d45, 0xedd2, 0x4adc, 0xb9, 0xd1, 0x8b, 0xec, 0x49, 0x25, 0xc5, 0x35 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Receive() when the child has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive() when the child has not been configured. The return status should be EFI\_NOT\_STARTED.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.6.2 | 0xe0605ca4, 0x21d1, 0x4692, 0xa4, 0xcc, 0x90, 0x5f, 0xbe, 0xb0, 0xa9, 0xb5 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Receive() when the receive completion token is already in the receive queue. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive() to place the token into the receiving queue.  4. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive() to receive the token which was placed in the receiving queue in step 3. The rerurn status should be EFI\_ACCESS\_DENIED.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.6.3 | 0x9349ff52, 0x8bfb, 0x4018, 0xa8, 0x5a, 0x41, 0x71, 0xb8, 0x36, 0x9f, 0x28 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Receive() with a Token value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive() with a Token value of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.6.4 | 0xfdb1c2d3, 0xcc35, 0x4bc7, 0xac, 0xa6, 0x6d, 0x0f, 0xda, 0x79, 0x85, 0x55 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Receive() with a Token.Event value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive() with a Token.Event value of NULL. The return status should be EFI\_INVALID\_PARAMETER.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.6.5 | 0x23fb0e81, 0xe831, 0x40fa, 0x8c, 0xc9, 0xc4, 0x10, 0x2f, 0x7d, 0x8f, 0xdc | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Receive() to place an asynchronous receiving request into the receiving queue. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Receive() to place an asynchronous receiving request into the receiving queue. The return status should be EFI\_SUCCESS.  4. Verify that the received data is correct.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.6.6 | 0x2c0e86ce, 0xec73, 0x4840, 0x9c, 0x07, 0xb5, 0xf1, 0x75, 0xc6, 0x81, 0x79 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Cancel() to abort the receive | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Receive() to place an asynchronous receiving request into the receiving queue. The return status should be EFI\_SUCCESS.  4. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.Cancel() to abort the receive. The return status should be EFI\_SUCCESS.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.6.7 | 0x4e364693, 0xe0c7, 0x49d3, 0xa0, 0xe5, 0xb8, 0x43, 0xd4, 0x79, 0x84, 0xe6 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Receive – invokes Receive() to place an asynchronous receiving request into the receiving queue. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Receive() to place an asynchronous receiving request into the receiving queue. The return status should be EFI\_SUCCESS.  4. Verify source MAC address correction.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### Cancel()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.7.1 | 0xf8c7e036, 0xfb8e, 0x4fbb, 0x94, 0x7c, 0x1c, 0x72, 0x75, 0xf5, 0xb9, 0x1f | EFI\_MANAGED\_NETWORK\_PROTOCOL.Cancle – invokes Cancel()when the child has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Cancel() when the child has not been configured. The return status should be EFI\_NOT\_STARTED.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.7.2 | 0x36ca4137, 0x5272, 0x469b, 0xad, 0x35, 0xba, 0xb4, 0x25, 0xb6, 0x4c, 0x27 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Cancle – invokes Cancel()when the value of the Token parameter is not NULL but the asynchronous I/O request was not found in the transmit or receive queues. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Cancel()when the value of the Token parameter is not NULLbut the asynchronous I/O request was not found in the transmit or receive queues. The return status should be **EFI\_NOT\_FOUND**.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.7.3 | 0xe873ef06, 0x2a4c, 0x4679, 0xa3, 0xf8, 0xd1, 0x02, 0x17, 0x1c, 0x11, 0xeb | EFI\_MANAGED\_NETWORK\_PROTOCOL.Cancle – invokes Cancel() when the value of the Token parameter is NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Cancel() when the value of the Token parameter is NULL. The return status should be EFI\_SUCCESS.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.7.4 | 0x21288fe0, 0x7c33, 0x423c, 0xaa, 0xd7, 0x95, 0x79, 0xa7, 0xec, 0xc6, 0x04 | EFI\_MANAGED\_NETWORK\_PROTOCOL.Cancle – invokes Cancel()to abort an asynchronous transmit or receive request. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Configure() to configure the parameters for the new MNP child.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Receive() to place a asynchronous request into the receive queue.  4. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Cancel()to abort an asynchronous transmit or receive request. The return status should be EFI\_SUCCESS.  5. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### Poll()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.8.1 | 0xf87f9d7f, 0xbe91, 0x4b28, 0xb6, 0x8d, 0x49, 0x4e, 0x28, 0x18, 0x07, 0xca | EFI\_MANAGED\_NETWORK\_PROTOCOL.Poll – invokes **Poll()** when the child has not been configured. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() to create a new MNP child.  2. Call EFI\_MANAGED\_NETWORK\_PROTOCOL.Poll() when the child has not been configured. The return status should be EFI\_NOT\_STARTED.  3. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### CreateChild()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.9.1 | 0x026c7391, 0x7ebe, 0x4715, 0xba, 0xe4, 0xc5, 0x1b, 0x2e, 0x9a, 0x99, 0xf4 | EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild – invokes CreateChild() with a ChildHandle value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() with a ChildHandle value of NULL. The return status should be EFI\_INVALID\_PARAMETER. |
| 5.23.1.9.2 | 0x48b5ff0b, 0xd688, 0x4644, 0x86, 0x62, 0xa9, 0x63, 0x6f, 0x2f, 0x4c, 0x1c | EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild – invokes CreateChild() with a ChildHandle value of NULL. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild() with a ChildHandle value of NULL. The return status should be EFI\_SUCCESS.  2. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |
| 5.23.1.9.3 | 0x27da9434, 0x20fa, 0x42af, 0x8b, 0xdf, 0x87, 0x8e, 0xc9, 0x8b, 0x3b, 0xb9 | EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild – invokes CreateChild() when the ChildHandle value is an existing instance handle. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild()with valid parameter to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild()**with the parameter ChildHandle pointing to the handle created in step 1. The return status should be **EFI\_INVALID\_PARAMETE**.  3. Call **EFI\_MANAGED\_NETWORK\_PROTOCOL.**Configure() to configure the parameters for the new child.  4. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the created MNP child and clean up the environment. |

### DestroyChild()

|  |  |  |  |
| --- | --- | --- | --- |
| Number | GUID | Assertion | Test Description |
| 5.23.1.10.1 | 0xc400df8b, 0x61d0, 0x4244, 0xb2, 0xec, 0xed, 0x2f, 0xc6, 0x54, 0x8c, 0x7e | EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild – invokes DestroyChild() when the child does not exist. | 1.Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild() when the parameter ChildHandle *i*s NULL. The return status should be EFI\_INVALID\_PARAMETER. |
| 5.23.1.10.2 | 0x9ed9c819, 0x95fc, 0x4b00, 0x99, 0x7c, 0x36, 0x20, 0xfa, 0x9f, 0xad, 0xb3 | EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild – invokes DestroyChild() to destroy an existing child. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild()to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the child handle created in step 1. The return status should be EFI\_SUCCESS. |
| 5.23.1.10.3 | 0x8182f56c, 0x3fe6, 0x4583, 0x9b, 0xb7, 0xfd, 0x8a, 0xe2, 0x1b, 0xe6, 0xac | EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild – invokes DestroyChild()twice to destroy one child handle created before. | 1. Call EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.CreateChild()to create a new MNP child.  2. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the child handle created in step 1. The return status should be EFI\_SUCCESS.  2. Call **EFI\_MANAGED\_NETWORK\_SERVICE\_BINDING\_PROTOCOL.DestroyChild()** to destroy the child handle created in step 1 again. The return status should be **EFI\_UNSUPPORTED**. |