#### writeProtection (Write Protection)

This element specifies the write protection settings which have been applied to a WordprocessingML document. Write protection refers to a mode in which the document's contents cannot be edited, and the document cannot be resaved using the same file name. This setting is independent of the [documentProtection](documentProtection.docx) (§) element, but like document protection, this setting is not intended as a security feature and may be ignored.

When present, the write protection shall [result](result.docx) in one of two write protection behaviors:

* If the password attribute is present, or both attributes are omitted, then the application shall prompt for a password to exit write protection. If the supplied password does not match the hash value in this attribute, then write protection shall be enabled.
* If only the recommended attribute is present, the application should provide user interface recommending that the user open this document in write protected state. If the user chooses to do so, the document shall be write protected, otherwise, it shall be opened fully editable.

If this element is omitted, then no write protection shall be applied to the current document.

[Example: Consider a WordprocessingML document that can be opened but only in a write protected state unless a password is provided, in which case the file would be opened in an editable state. This requirement would be specified using the following WordprocessingML in the document settings:

<w:writeProtection w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />

The writeProtection element is present which specifies that write protection shall be turned on for this document. Since the password attribute is equal to 9oN7nWkCAyEZib1RomSJTjmPpCY= the given WordprocessingML document can only be opened in a write protected state unless a password which matches the hash value 9oN7nWkCAyEZib1RomSJTjmPpCY=is provided; in which case the file would be opened in an editable state. end example]

|  |
| --- |
| Parent Elements |
| settings (§) |

|  |  |
| --- | --- |
| Attributes | Description |
| algIdExt (Cryptographic Algorithm Extensibility) | Specifies that a cryptographic algorithm which was not defined by this Office Open [XML](XML.docx) Standard has been used to generate the hash value stored with this document.This value, when present, shall be interpreted based on the value of the algIdExtSource attribute in order to determine the algorithm used, which shall be application-defined. [Rationale: This extensibility affords the fact that with exponentially increasing computing power, documents created in the future will likely need to utilize as yet undefined hashing algorithms in order to remain secure. end rationale]If this value is present, the cryptAlgorithmClass, cryptAlgorithmType, and cryptAlgorithmSid attribute values shall be ignored in favor of the algorithm defined by this attribute. [Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:algIdExt="0000000A"  w:algIdExtSource="futureCryptography" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The algIdExt attribute value of 0000000A specifies that the algorithm with hex code A shall be used as defined by the futureCryptography application. end example]The possible values for this attribute are defined by the [ST\_LongHexNumber](ST_LongHexNumber.docx) simple [type](type.docx) (§). |
| algIdExtSource (Algorithm Extensibility Source) | Specifies the application which defined the algorithm value specified by the algIdExt attribute.[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:algIdExt="0000000A"  w:algIdExtSource="futureCryptography" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The algIdExtSource attribute value of futureCryptography specifies that the algorithm used here was published by the futureCryptography application. end example]The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§). |
| cryptAlgorithmClass (Cryptographic Algorithm Class) | Specifies the class of cryptographic algorithm used by this protection. [Note: The initial version of this Office Open [XML](XML.docx) Standard only supports a single version - hash - but future versions may expand this as necessary. end note][Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptAlgorithmClass="hash"  w:cryptAlgorithmType="typeAny"  w:cryptAlgorithmSid="1" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptAlgorithmClass attribute value of hash specifies that the algorithm used for the password is a hashing algorithm. end example]The possible values for this attribute are defined by the [ST\_AlgClass](ST_AlgClass.docx) simple [type](type.docx) (§). |
| cryptAlgorithmSid (Cryptographic Hashing Algorithm) | Specifies the specific cryptographic hashing algorithm which shall be used along with the salt attribute and user-supplied password in order to compute a hash value for comparison.The possible values for this attribute shall be interpreted as follows:

|  |  |
| --- | --- |
| Value | Algorithm |
| 1 | MD2 |
| 2 | MD4 |
| 3 | MD5 |
| 4 | SHA-1 |
| 5 | MAC |
| 6 | RIPEMD |
| 7 | RIPEMD-160 |
| 8 | Undefined. Shall not be used. |
| 9 | HMAC |
| 10 | Undefined. Shall not be used. |
| 11 | Undefined. Shall not be used. |
| 12 | SHA-256 |
| 13 | SHA-384 |
| 14 | SHA-512 |
| Any other value | Undefined. Shall not be used. |

[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptAlgorithmClass="hash"  w:cryptAlgorithmType="typeAny"  w:cryptAlgorithmSid="1" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptAlgorithmSid attribute value of 1 specifies that the SHA-1 hashing algorithm shall be used to generate a hash from the user-defined password. end example]The possible values for this attribute are defined by the [ST\_DecimalNumber](ST_DecimalNumber.docx) simple [type](type.docx) (§). |
| cryptAlgorithmType (Cryptographic Algorithm Type) | Specifies the [type](type.docx) of cryptographic algorithm used by this protection. [Note: The initial version of this Office Open [XML](XML.docx) Standard only supports a single [type](type.docx) - typeAny - but future versions may expand this as necessary. end note][Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptAlgorithmClass="hash"  w:cryptAlgorithmType="typeAny"  w:cryptAlgorithmSid="1" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptAlgorithmType attribute value of typeAny specifies that any [type](type.docx) of algorithm may have been used for the password. end example]The possible values for this attribute are defined by the [ST\_AlgType](ST_AlgType.docx) simple [type](type.docx) (§). |
| cryptProvider (Cryptographic Provider) | Specifies the cryptographic provider which was used to generate the hash value stored in this document. If the user provided a cryptographic provider which was not the system's built-in provider, then that provider shall be stored here so it can subsequently be used if available.If this attribute is omitted, then the built-in cryptographic provider on the system shall be used.[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptProvider="Krista'sProvider" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptProvider attribute value of Krista'sProvider specifies that the cryptographic provider with name "Krista's Provider" shall be used if available. end example]The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§). |
| cryptProviderType (Cryptographic Provider Type) | Specifies the [type](type.docx) of cryptographic provider to be used.[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptProviderType="rsaAES" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptProviderType attribute value of rsaAES specifies that the cryptographic provider [type](type.docx) shall be an Advanced Encryption Standard provider. end example]The possible values for this attribute are defined by the [ST\_CryptProv](ST_CryptProv.docx) simple [type](type.docx) (§). |
| cryptProviderTypeExt (Cryptographic Provider Type Extensibility) | Specifies that a cryptographic provider [type](type.docx) which was not defined by this Office Open [XML](XML.docx) Standard has been used to generate the hash value stored with this document.This value, when present, shall be interpreted based on the value of the cryptProviderTypeExtSource attribute in order to determine the provider [type](type.docx) used, which shall be application-defined. [Rationale: This extensibility affords the fact that with exponentially increasing computing power, documents created in the future will likely need to utilize as yet undefined cryptographic provider types in order to remain secure. end rationale]If this value is present, the cryptProviderType attribute value shall be ignored in favor of the provider [type](type.docx) defined by this attribute. [Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptProviderTypeExt="00A5691D"  w:cryptProvideTypeExtSource="futureCryptography" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptProviderTypeExt attribute value of 00A5691D specifies that the provider [type](type.docx) associated with hex code A5691D shall be used as defined by the futureCryptography application. end example]The possible values for this attribute are defined by the [ST\_LongHexNumber](ST_LongHexNumber.docx) simple [type](type.docx) (§). |
| cryptProviderTypeExtSource (Provider Type Extensibility Source) | Specifies the application which defined the provider [type](type.docx) value specified by the cryptProviderTypeExt attribute.[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptProviderTypeExt="00A5691D"  w:cryptProvideTypeExtSource="futureCryptography" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptProvideTypeExtSource attribute value of futureCryptography specifies that the provider [type](type.docx) used here was published by the futureCryptography application. end example]The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§). |
| cryptSpinCount (Iterations to [Run](Run.docx) Hashing Algorithm) | Specifies the number of times the hashing function shall be iteratively run (using each iteration's [result](result.docx) as the input for the [next](next.docx) iteration) when attempting to compare a user-supplied password with the value stored in the hash attribute. [Rationale: Running the algorithm many times increases the cost of exhaustive search attacks correspondingly. Storing this value allows for the number of iterations to be increased over time to accommodate faster hardware (and hence the ability to run more iterations in less time). end rationale][Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptSpinCount="100000" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The cryptSpinCount attribute value of 100000 specifies that the hashing function shall be run one hundred thousand times to generate a hash value for comparison with the hash attribute. end example]The possible values for this attribute are defined by the [ST\_DecimalNumber](ST_DecimalNumber.docx) simple [type](type.docx) (§). |
| hash (Password Hash) | Specifies the hash value for the password stored with this document. This value shall be compared with the resulting hash value after hashing the user-supplied password using the algorithm specified by the preceding attributes and parent [XML](XML.docx) element, and if the two values match, the protection shall no longer be enforced.If this value is omitted, then no password shall be associated with the protection, and it may be turned off without supplying any password.[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:cryptAlgorithmClass="hash"  w:cryptAlgorithmType="typeAny"  w:cryptAlgorithmSid="1" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The hash attribute value of 9oN7nWkCAyEZib1RomSJTjmPpCY= specifies that the user-supplied password shall be hashed using the pre-processing defined by the parent element (if any) followed by the SHA-1 algorithm (specified via the cryptAlgorithmSid attribute value of 1) and that the resulting has value must be 9oN7nWkCAyEZib1RomSJTjmPpCY= for the protection to be disabled. end example]The possible values for this attribute are defined by the [XML](XML.docx) Schema base64Binary datatype. |
| recommended (Recommend Write Protection in User Interface) | Specifies that applications should provide user interface recommending that the user open this document in write protected state. If the user chooses to do so, the document shall be write protected, otherwise, it shall be opened fully editable.If this attribute is omitted, then user interface recommending that the user open this document in write protected state should not be provided. If the password attribute is also specified, then this setting shall be ignored.[Example: Consider a WordprocessingML document which specifies that applications shall recommend write protection to this document. This requirement would be specified using the following WordprocessingML in the document settings: <w:writeProtection w:recommended="true" /> The recommended attribute has a value of true specifying that the applications shall hash any password provided, and if it matches this hash value, may only then halt enforcement of write protection. end example]The possible values for this attribute are defined by the [ST\_OnOff](ST_OnOff.docx) simple [type](type.docx) (§). |
| salt (Salt for Password Verifier) | Specifies the salt which was prepended to the user-supplied password before it was hashed using the hashing algorithm defined by the preceding attribute values to generate the hash attribute, and which shall also be prepended to the user-supplied password before attempting to generate a hash value for comparison. A salt is a random string which is added to a user-supplied password before it is hashed in order to prevent a malicious party from pre-calculating all possible password/hash combinations and simply using those precalculated values (often referred to as a "dictionary attack").If this attribute is omitted, then no salt shall be prepended to the user-supplied password before it is hashed for comparison with the stored hash value.[Example: Consider a WordprocessingML document with the following information stored in one of its protection elements:<w:… w:salt="ZUdHa+D8F/OAKP3I7ssUnQ==" w:hash="9oN7nWkCAyEZib1RomSJTjmPpCY=" />The salt attribute value of ZUdHa+D8F/OAKP3I7ssUnQ== specifies that the user-supplied password shall have this value prepended before it is run through the specified hashing algorithm to generate a resulting hash value for comparison. end example]The possible values for this attribute are defined by the [XML](XML.docx) Schema base64Binary datatype. |

The following [XML](XML.docx) Schema fragment defines the contents of this element:

<complexType [name](name.docx)="CT\_WriteProtection">

 <attribute [name](name.docx)="recommended" [type](type.docx)="[ST\_OnOff](ST_OnOff.docx)" use="optional"/>

 <attributeGroup ref="AG\_Password"/>

</complexType>