[MS-OXPHISH]:
Phishing Warning Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Copyright © 2010 Microsoft Corporation.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability.
04/25/2008	0.2		Revised and updated property names and other technical content.
06/27/2008	1.0		Initial Release.
08/06/2008	1.01		Revised and edited technical content.
09/03/2008	1.02		Updated references.
12/03/2008	1.03		Updated IP notice.
02/04/2009	1.04		Revised and edited technical content.
03/04/2009	1.05		Revised and edited technical content.
04/10/2009	2.0		Updated applicable product releases.
07/15/2009	3.0	Major	Revised and edited for technical content.
11/04/2009	3.0.1	Editorial	Revised and edited the technical content.
02/10/2010	4.0.0	Major	Updated and revised the technical content.

Copyright © 2010 Microsoft Corporation.

Table of Contents

Copyright © 2010 Microsoft Corporation.

1 Introduction

This document specifies the **Phishing** Warning protocol that is used by the client to identify and mark e-mail **messages** that are designed to trick recipients into divulging sensitive information (such as passwords and/or other personal information) to a non-trustworthy source.

1.1 Glossary

The following terms are defined in [MS-OXGLOS]:

big-endian
GUID
handle
little-endian
message
Message object
named property
phishing
phishing message
property
property ID
remote operation (ROP)

The following terms are specific to this document:

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-OXCDATA] Microsoft Corporation, "Data Structures", June 2008.

[MS-OXCMSG] Microsoft Corporation, "Message and Attachment Object Protocol Specification", June 2008.

[MS-OXCPRPT] Microsoft Corporation, "Property and Stream Object Protocol Specification", June 2008.

[MS-OXCROPS] Microsoft Corporation, "Remote Operations (ROP) List and Encoding Protocol Specification", June 2008.

[MS-OXCSPAM] Microsoft Corporation, "Spam Confidence Level Protocol Specification", June 2008.

[MS-OXGLOS] Microsoft Corporation, "Exchange Server Protocols Master Glossary", June 2008.

[MS-OXOMSG] Microsoft Corporation, "E-Mail Object Protocol Specification", June 2008.

Copyright © 2010 Microsoft Corporation.

[MS-OXOSFLD] Microsoft Corporation, "Special Folders Protocol Specification", June 2008.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt.

1.2.2 Informative References

None.

1.3 Protocol Overview

This protocol enables the client to identify and mark e-mail messages that are likely to be phishing. When an e-mail message is delivered to a messaging client, the client examines the message **properties** to determine the likelihood of it being a **phishing message**. If the examination determines that the message is likely to be phishing, the client modifies a property on the message to mark it as suspicious. A messaging client's user interface can utilize this property value to identify a potential phishing message and display a warning to the end-user.

This protocol does not specify the algorithm that determines the likelihood of a message being a phishing message; it only specifies how the **Message object** is changed to indicate the result of the algorithm.

1.4 Relationship to Other Protocols

The phishing Warning protocol uses a property on the Message object as a means of identifying and marking messages that are likely to be phishing. Therefore, this specification relies on the following:

- An understanding of the Message object, as specified in [MS-OXOMSG].
- An understanding of getting and setting properties, as specified in [MS-OXCMSG].

1.5 Prerequisites/Preconditions

This specification assumes that the client has previously logged on to the server and has acquired a **handle** to the Message for which it has to identify or designate phishing status.

1.6 Applicability Statement

A client can use this protocol to identify or mark messages that are likely to be phishing. This protocol does not specify the algorithm that determines the likelihood of a Message that is a phishing message; it only specifies how the Message object is changed to indicate the result of such analysis.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

Copyright © 2010 Microsoft Corporation.

2 Messages

2.1 Transport

Message properties are transported between client and server, as specified in [MS-OXCMSG]. This protocol references commonly used data types as defined in [MS-OXCDATA].

2.2 Message Syntax

Before sending requests to the server, the client MUST obtain a handle to the Message object used in property operations.

2.2.1 Phishing Warning Protocol Properties

The following properties are specific to the phishing Warning protocol.

2.2.1.1 PidNamePhishingStamp

Type: **PtypInteger32**

The value of this **named property** is a 32-**bit** field. The structure is specified as follows. For clarity, individual fields are presented in **big-endian** byte order, but the diagram does not reflect the actual byte ordering of the **PtypInteger32** value in the **ROP** buffer. The value of this **PtypInteger32** property is encoded in **little-endian** byte order in the ROP buffer.

E: (ENABLED bitmask 0x10000000) If the value of this field is 1, the user has enabled functionality (such as hyperlinks, reply, and attachments) within the message. The default value for this field is zero (0), which indicates that the user has not enabled functionality.

STAMP: (bitmask 0x0FFFFFFF) This field is obtained from the fifth value of the PidTagAdditionalRenEntryIds property. For more information about this property, see [MS-OXOSFLD].

x: Unused. These bits SHOULD be set to zero (0) by the client and ignored by the server.

The client SHOULD set this property if it is determined that the message is likely to be phishing. The client SHOULD use the value of this property to warn the user when a message is likely to be phishing.

Copyright © 2010 Microsoft Corporation.

3 Protocol Details

The role of the client is to determine whether a Message is phishing and to update the PidNamePhishingStamp property (as specified in 3.1.1.1) to indicate the results of such analysis. The client then checks the value of the PidNamePhishingStamp property when the Message is opened, and conveys a warning to the end user for any Message that is likely to be phishing.

3.1 Client Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with the behavior described in this document.

3.1.1.1 Setting the PidNamePhishingStamp Property

If the client determines that a message is phishing, it SHOULD then update the value of the PidNamePhishingStamp property to indicate whether the message is likely to be phishing.

The PidNamePhishingStamp property value is calculated as follows:

A query for the fifth value in the PidTagAdditionalRenEntryIds property is performed. Let the queried value be called QueriedValue_FromEntryID.

The mask (0x0FFFFFFF) is then applied to QueriedValue_FromEntryID. That is, the bitwise operation (0x0FFFFFFF AND QueriedValue_FromEntryID) is performed to produce the STAMP field of PidNamePhishingStamp.

If the user has not enabled functionality on the message, the value of the ENABLED field is zero (0) and the final property value is the same as the value of the STAMP field. If the user determines that the message is not a phishing message and indicates as such by interaction with the user interface, the final PidNamePhishingStamp property value with ENABLED field 1 is produced by applying the bitwise operation (STAMP OR 0x10000000).

3.1.2 Timers

None.

3.1.3 Initialization

Before matching the PidNamePhishingStamp on the Message, the existence of the fifth value of PidTagAdditionalRenEntryIds MUST be ensured. If it is not present, the value MUST be created (as specified in [MS-OXCSPAM] section 3.1.4.1.2).

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Client Receives a New Message

When the client receives a new Message, the client determines whether the Message is likely to be phishing. If on delivery, the client determines that the Message is likely to be phishing, the client sets the PidNamePhishingStamp property on the Message (as specified in 3.1.1.1).

Copyright © 2010 Microsoft Corporation.

3.1.4.2 End-User Opens a Message

When an end user opens a Message, the client tries to retrieve the value of the PidNamePhishingStamp property (as specified in 2.2.1). If the property is present, its **STAMP** field is compared against the fifth value of the multi-valued property PidTagAdditionalRenEntryIds. If this comparison does not result in a match, the PidNamePhishingStamp property SHOULD be ignored. If the comparison results in a match, the client considers the Message to be a phishing message. If the value of the **ENABLED** field in the PidNamePhishingStamp property is 1, the user has enabled the functionality, and the client SHOULD display the Message as a normal message. If the value of the **ENABLED** field in the PidNamePhishingStamp property is zero (0), the client SHOULD disable the functionality of the Message. The functionality that the client chooses to disable (according to the value of the **ENABLED** field in the PidNamePhishingStamp property) is implementation-dependent.

The user has the option to enable all functionality within a Message by interaction with the user interface. If the user enables functionality within a Message, the value of the **ENABLED** field of the PidNamePhishingStamp property on that Message (as specified in 2.2.1) is set to 1.

The functionality is also enabled when the PidTagJunkPhishingEnableLinks property (as specified in [MS-OXCSPAM]) is set to TRUE.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

Copyright © 2010 Microsoft Corporation.

4 Protocol Examples

4.1 Setting the PidNamePhishingStamp Property

When the client receives a new Message, the client determines whether the Message is likely to be phishing. If the client determines that the Message is likely to be phishing, the client sets the PidNamePhishingStamp property on the Message (as specified in 3.1.1.1) on Message delivery. The client calculates the PidNamePhishingStamp property value as described in the following example:

If the fifth value queried from PidTagAdditionalRenEntryIds is 0xAE241D99, the client begins calculating the PidNamePhishingStamp property by setting the STAMP field as follows: (0xAE241D99 AND 0x0FFFFFFF) = 0x0E241D99.

The value of the ENABLED field of the PidNamePhishingStamp property can be either zero (0), if the user has not enabled the functionality of the Message, or 1, if the user has enabled the functionality of the Message. If the value of the ENABLED field is zero (0), the final PidNamePhishingStamp property value is 0x0E241D99. If the value of the ENABLED field is 1, the final PidNamePhishingStamp property value is the result of the bitwise operation (0x0E241D99 OR 0x10000000) = 0x1E241D99.

4.2 Evaluating the PidNamePhishingStamp Property

For purposes of the examples in 4.2, let the fifth value queried from PidTagAdditionalRenEntryIds be called **PhishingTagValue**.

4.2.1 No PidNamePhishingStamp Property

If the PidNamePhishingStamp property is absent from a Message, the client does not consider the Message to be a phishing message.

4.2.2 PidNamePhishingStamp Property Mismatch

If the PidNamePhishingStamp property is present, the client will compare its **STAMP** field with the least significant 28 **bits** of **PhishingTagValue**. If the PidNamePhishingStamp property value is 0x0EAE2103 and **PhishingTagValue** is 0xAE241D99, the comparison does not result in a match. Therefore, the client ignores the PidNamePhishingStamp property, resulting in enabled Message functionality and no added phishing-related user interface elements.

4.2.3 PidTagJunkPhishingEnableLinks Property Set to True

If the PidTagJunkPhishingEnableLinks property is present and is set to **true**, the client will ignore the PidNamePhishingStamp property and will treat the Message as non-phishing.

4.2.4 Phishing Message Functionality Not Enabled By the User

If the PidNamePhishingStamp property is present, the client will compare its **STAMP** field with the least significant 28 **bits** of **PhishingTagValue**. If the PidNamePhishingStamp property value is 0x0E241D99, and **PhishingTagValue** is 0xAE241D99, the comparison results in a match, indicating that the Message is likely to be phishing. If the value of the **ENABLED** field of the PidNamePhishingStamp property (as specified in section 2.2.1) is zero (0), the user has not enabled functionality within the Message. Therefore, the client will disable functionality within the Message, display a warning to the user, and add phishing-related user interface elements that allow the user to enable Message functionality.

Copyright © 2010 Microsoft Corporation.

4.2.5 Phishing Message Functionality Enabled By the User

If the PidNamePhishingStamp property is present, the client will compare its **STAMP** field with the least significant 28 **bits** of **PhishingTagValue**. If the PidNamePhishingStamp property value is 0x1E241D99 and **PhishingTagValue** is 0xAE241D99, the comparison results in a match, which indicates that the Message is likely to be phishing. Because the value of the **ENABLED** field of the PidNamePhishingStamp property is 1, the user has enabled functionality within the Message. Therefore, the client will treat the Message as non-phishing.

4.3 Sample Properties on a Phishing Message

The following is a description of what a client does to stamp the Message that has been identified as phishing and the responses that a server returns. The ROP input and responses are summarized in this section; for a complete explanation of how to set properties by using RopSetProperties, see [MS-OXCPRPT].

Because the PidNamePhishingStamp property is a named property, the client MUST ask the server to perform mapping from named properties to **property IDs**, by using RopGetPropertyIDsFromNames, as specified in [MS-OXCROPS].

property	property set GUID	Name or ID
PidNamePhishingStamp	{00020329-0000-0000-C000-000000000046}	HTTP://schemas.microsoft.com/outlook/phishingstamp

The server returns the following property IDs in response to RopGetPropertyIDsFromNames.

property	property ID
PidNamePhishingStamp	0x831F

After determining the value of the property, the client uses RopSetProperties to transmit the data to the server.

property	property ID	property type	Value
PidNamePhishingStamp	0x831F	0x0003(PT_LONG)	0x0A73AE09

If the user enables the functionality of the phishing message, the property value is changed and the client uses RopSetProperties to transmit the new value to the server.

property	property ID	property type	Value
PidNamePhishingStamp	0x831F	0x0003(PT_LONG)	0x1A73AE09

The client then uses RopSaveChangesMessage to commit the properties to the server.

Copyright © 2010 Microsoft Corporation.

5 Security

5.1 Security Considerations for Implementers

On delivery of the Message, the presence of the PidNamePhishingStamp with a successful match of the **STAMP** field signals the client that the Message has already been evaluated for phishing and does not have to be filtered again. Therefore, care has to be taken while setting the PidNamePhishingStamp property on the Message and all precautions for evaluation of the fifth value of PidTagAdditionalRenEntryIds have to be followed (as specified in [MS-OXCMSG]).

5.2 Index of Security Parameters

None.

Copyright © 2010 Microsoft Corporation.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following product versions. References to product versions include released service packs.

- Microsoft Office Outlook 2007
- Microsoft Exchange Server 2007
- Microsoft Outlook 2010
- Microsoft Exchange Server 2010

Exceptions, if any, are noted below. If a service pack number appears with the product version, behavior changed in that service pack. The new behavior also applies to subsequent service packs of the product unless otherwise specified.

Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that product does not follow the prescription.

Copyright © 2010 Microsoft Corporation.

7 Change Tracking

This section identifies changes made to [MS-OXPHISH] protocol documentation between November 2009 and February 2010 releases. Changes are classed as major, minor, or editorial.

Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- A protocol is deprecated.
- The removal of a document from the documentation set.
- Changes made for template compliance.

Minor changes do not affect protocol interoperability or implementation. Examples are updates to fix technical accuracy or ambiguity at the sentence, paragraph, or table level.

Editorial changes apply to grammatical, formatting, and style issues.

No changes means that the document is identical to its last release.

Major and minor changes can be described further using the following revision types:

- New content added.
- Content update.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.

Copyright © 2010 Microsoft Corporation.

- Content removed for template compliance.
- Obsolete document removed.

Editorial changes always have the revision type "Editorially updated."

Some important terms used in revision type descriptions are defined as follows:

Protocol syntax refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.

Protocol revision refers to changes made to a protocol that affect the bits that are sent over the wire.

Changes are listed in the following table. If you need further information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Revision Type
1.2.1 Normative References	Removed line item for unused reference [MS-DTYP].	N	Content removed.
1.2.1 Normative References	48768 Added information for reference [MS-OXCDATA].	N	New content added.
2.1 Transport	48768 Added information about where data types are defined.	N	New content added.
2.2.1.1 PidNamePhishingStamp	48775 Updated guidance for the 'x' field in the PidNamePhishingStamp structure.	N	Content update.
3.1.3 Initialization	48771 Added information about how to create the fifth value of PidTagAdditionalRenEntryIds.	Y	Content update.
4.3 Sample Properties on a Phishing Message	48766 Changed term "property identifiers" to "property IDs" so that it's consistent with the glossary.	N	Content update.

Copyright © 2010 Microsoft Corporation.

8 Index

Copyright © 2010 Microsoft Corporation.