[MS-OXPHISH]: Phishing Warning Protocol Specification

Intellectual Property Rights Notice for Protocol Documentation

- **Copyrights.** This protocol 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 protocols, and may distribute portions of it in your implementations of the protocols 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 protocol documentation.
- **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 protocols. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, the protocols may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp). If you would prefer a written license, or if the protocols are not covered by the OSP, patent licenses are available by contacting protocol@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.

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. This protocol documentation is 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. A protocol specification does 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.

Revision Summary			
Author	Date	Version	Comments
Microsoft Corporation	April 4, 2008	0.1	Initial Availability.
Microsoft Corporation	April 25, 2008	0.2	Revised and updated property names and other technical content.
Microsoft Corporation	June 27, 2008	1.0	Initial Release.
Microsoft Corporation	August 6, 2008	1.01	Revised and edited technical content.
Microsoft Corporation	September 3, 2008	1.02	Updated references.
Microsoft Corporation	December 3, 2008	1.03	Updated IP notice.
Microsoft	February	1.04	Revised and edited technical content.

Corporation	4, 2009		

Table of Contents

1 Introduction ***5***
1.1 Glossary 5
1.2 References 5
1.2.1 Normative References 5
1.2.2 Informative References 6
1.3 Protocol Overview 6
1.4 Relationship to Other Protocols 6
1.5 Prerequisites/Preconditions 7
1.6 Applicability Statement 7
1.7 Versioning and Capability Negotiation 7
1.8 Vendor-Extensible Fields 7
1.9 Standards Assignments 7
2 Messages ***7***
2.1 Transport 7
2.2 Message Syntax 7
2.2.1 Phishing Warning Protocol Properties 7
2.2.1.1 PidNamePhishingStamp 7
3 Protocol Details ***8***
3.1 Client Details 8
3.1.1 Abstract Data Model 8
3.1.1.1 Setting the PidNamePhishingStamp Property 8
3.1.2 Timers 9
3.1.3 Initialization 9
3.1.4 Higher-Layer Triggered Events 9
3.1.4.1 Client Receives a New Message 9
3.1.4.2 End-User Opens a Message 9
3.1.5 Message Processing Events and Sequencing Rules 10
3.1.6 Timer Events 10
3.1.7 Other Local Events 10
4 Protocol Examples ***10***
4.1 Setting the PidNamePhishingStamp Property 10
4.2 Evaluating the PidNamePhishingStamp Property 10
4.2.1 No PidNamePhishingStamp Property 11
4.2.2 PidNamePhishingStamp Property Mismatch 11
4.2.3 PidTagJunkPhishingEnableLinks Property Set to True 11
4.2.4 Phishing Message Functionality Not Enabled By the User 11
4.2.5 Phishing Message Functionality Enabled By the User 11
4.3 Sample Properties on a Phishing Message 11
5 Security ***12***
5.1 Security Considerations for Implementers 12
5.2 Index of Security Parameters 12

6 Appendix A: Office/Exchange Behavior 13
Index 14

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 data types are defined in [MS-DTYP]:

bit
byte

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

[MS-DTYP] Microsoft Corporation, "Windows Data Types", March 2007, http://go.microsoft.com/fwlink/?LinkId=111558.

[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, Allow and Block Lists 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.

[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.

2 Messages

2.1 Transport

Message properties are transported between client and server, as specified in [MS-OXCMSG].

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.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x	x	x	E	STAMP																											

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 MUST 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.

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 section 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 a message is likely to be phishing.

The **PidNamePhishingStamp** property value is calculated as follows:

1. A query for the fifth value in the **PidTagAdditionalRenEntryIds** property is performed. Let the queried value be called QueriedValue_FromEntryID.
2. 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**.
3. 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.

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 section 3.1.1.1).

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 the section 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 section 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.

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 section 3.1.1.1) on message delivery. The client calculates the **PidNamePhishingStamp** property value as described in the following example:

1. 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.
2. 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 section 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.

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 identifiers, 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.

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 SHOULD NOT 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.

6 Appendix A: Office/Exchange Behavior

The information in this specification is applicable to the following versions of Office/Exchange:

- Office 2003 with Service Pack 3 applied
- Exchange 2003 with Service Pack 2 applied
- Office 2007 with Service Pack 1 applied
- Exchange 2007 with Service Pack 1 applied

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Office/Exchange behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies Office/Exchange does not follow the prescription.

Index

Applicability, 7
Client details, 8
Examples, 10
Fields, vendor-extensible, 7
Glossary, 5
Index of security parameters, 12
Informative references, 6
Introduction, 5
Message syntax, 7
Message transport, 7
Messages, 7
  Syntax, 7
  Transport, 7
Normative references, 5
Office/Exchange behavior, 13
Overview, 6
Preconditions, 7
Prerequisites, 7
Protocol details, 8
  Client details, 8
References, 5
  Informative references, 6
  Normative references, 5
Relationship to other protocols, 6
Security, 12
  Considerations for implementers, 12
  Index of security parameters, 12
Security considerations for implementers, 12
Standards assignments, 7
Vendor-extensible fields, 7
Versioning and capability negotiation, 7