[MS-XLOGIN]:

Simple Mail Transfer Protocol (SMTP) AUTH LOGIN Extension

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	New	Initial Availability.
6/27/2008	1.0	Major	Initial Release.
8/6/2008	1.01	Minor	Revised and edited technical content.
9/3/2008	1.02	Minor	Updated references.
12/3/2008	1.03	Minor	Updated IP notice.
4/10/2009	2.0	Major	Updated applicable product releases.
7/15/2009	3.0	Major	Revised and edited for technical content.
11/4/2009	4.0.0	Major	Updated and revised the technical content.
2/10/2010	4.1.0	Minor	Updated the technical content.
5/5/2010	4.1.1	Editorial	Revised and edited the technical content.
8/4/2010	5.0	Major	Significantly changed the technical content.
11/3/2010	5.0	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	5.0	None	No changes to the meaning, language, or formatting of the technical content.
8/5/2011	5.1	Minor	Clarified the meaning of the technical content.
10/7/2011	5.1	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	6.0	Major	Significantly changed the technical content.
4/27/2012	6.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	6.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	7.0	Major	Significantly changed the technical content.
2/11/2013	8.0	Major	Significantly changed the technical content.
7/26/2013	9.0	Major	Significantly changed the technical content.
11/18/2013	9.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	9.0	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	9.0	None	No changes to the meaning, language, or formatting of the

Date	Revision History	Revision Class	Comments
			technical content.
3/16/2015	10.0	Major	Significantly changed the technical content.
5/26/2015	10.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2015	11.0	Major	Significantly changed the technical content.
6/13/2016	12.0	Major	Significantly changed the technical content.
9/14/2016	12.0	None	No changes to the meaning, language, or formatting of the technical content.
3/16/2017	13.0	Major	Significantly changed the technical content.
9/15/2017	14.0	Major	Significantly changed the technical content.
12/12/2017	15.0	Major	Significantly changed the technical content.
7/24/2018	16.0	Major	Significantly changed the technical content.
10/1/2018	17.0	Major	Significantly changed the technical content.
4/7/2021	18.0	Major	Significantly changed the technical content.
4/22/2021	19.0	Major	Significantly changed the technical content.
6/25/2021	20.0	Major	Significantly changed the technical content.
8/17/2021	20.0	None	No changes to the meaning, language, or formatting of the technical content.
4/16/2024	21.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Simple Mail Transfer Protocol (SMTP) AUTH LOGIN Extension is an authentication mechanism that provides an easily implemented method for clients to authenticate to **SMTP** servers over a standard SMTP connection. This extension uses the SMTP Service Extension for Authentication, as specified in [RFC4954], to extend SMTP.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [RFC4648].

NT LAN Manager (NTLM) Authentication Protocol: A protocol using a challenge-response mechanism for authentication in which clients are able to verify their identities without sending a password to the server. It consists of three messages, commonly referred to as Type 1 (negotiation), Type 2 (challenge) and Type 3 (authentication).

SASL: The Simple Authentication and Security Layer, as described in [RFC2222]. This is an authentication mechanism used by the Lightweight Directory Access Protocol (LDAP).

Simple Mail Transfer Protocol (SMTP): A member of the TCP/IP suite of protocols that is used to transport Internet messages, as described in [RFC5321].

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. TLS supports server and, optionally, client authentication by using X.509 certificates (as specified in [X509]). TLS is standardized in the IETF TLS working group.

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

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

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.

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006, https://www.rfc-editor.org/info/rfc4648

[RFC4954] Siemborski, R., and Melnikov, A., Eds., "SMTP Service Extension for Authentication", RFC 4954, July 2007, http://www.rfc-editor.org/rfc/rfc4954.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, https://www.rfc-editor.org/info/rfc5234

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, October 2008, http://rfc-editor.org/rfc/rfc5321.txt

1.2.2 Informative References

[MS-OXPROTO] Microsoft Corporation, "Exchange Server Protocols System Overview".

[RFC4346] Dierks, T., and Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006, https://www.rfc-editor.org/info/rfc4346

1.3 Overview

Client applications use **SMTP** to transfer mail to a server for submission. Client applications that connect to an SMTP server can use a number of different authentication mechanisms. In some scenarios, clients can use existing authentication mechanisms to authenticate with the SMTP server, such as the **NT LAN Manager (NTLM) Authentication Protocol**. However, in other scenarios, existing authentication mechanisms are unavailable or clients may not implement them. This extension provides an authentication mechanism for SMTP clients that is simple to implement.

The SMTP Service Extension for Authentication, as specified in [RFC4954], defines a service extension to SMTP, as specified in [RFC5321], where a client specifies an authentication method to the server and performs an authentication protocol exchange. This extension is one such authentication method for SMTP. It allows clients to authenticate to SMTP servers over a standard SMTP connection by passing authentication information in SMTP commands and responses.

1.4 Relationship to Other Protocols

This extension uses the methods provided by the SMTP Service for Authentication, as specified in [RFC4954], to extend **SMTP**, as specified in [RFC5321], by providing a new authentication method. This extension relies on SMTP to provide the transport for the authentication commands and responses.

For conceptual background information and overviews of the relationships and interactions between this and other protocols, see [MS-OXPROTO].

1.5 Prerequisites/Preconditions

This extension conforms to all of the prerequisites and preconditions of **SMTP**, as specified in [RFC5321], and the extension to SMTP provided by the SMTP Service for Authentication, as specified in [RFC4954].

1.6 Applicability Statement

This extension is used by clients to support authentication to **SMTP** servers that implement the AUTH LOGIN extension. This extension is used by SMTP servers to provide an authentication method to control access to the SMTP service.

Since this extension does not encrypt the password sent over the network, it is only applicable to environments where a secure channel exists under the SMTP connection, such as **Transport Layer Security (TLS)**, as specified in [RFC4346].

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

This extension defines a **SASL** mechanism for use with the SMTP Service Extension for Authentication.

Parameter	Value	Reference
SASL mechanism	LOGIN	http://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xml

2 Messages

2.1 Transport

This extension does not change the base transport specified by [RFC5321], or its extension specified by [RFC4954].

2.2 Message Syntax

2.2.1 SASL Mechanism Name

The **SASL** mechanism name for this extension is defined as "LOGIN".

2.2.2 Command and Response ABNF Grammar

This section uses **Augmented Backus-Naur Form (ABNF)** (as specified in [RFC5234]) to define the format of commands and responses used by this extension, where CRLF, SP, and CHAR are specified in [RFC5234]. Note that the values of *username* and *password* MUST be encoded using **base64 encoding**, as specified in [RFC4648], before being transmitted.

```
username                        = 1*CHAR            ; Base64-encoded username
password                        = 1*CHAR            ; Base64-encoded password

auth_login_command              = "AUTH LOGIN" [SP username] CRLF
auth_login_username_challenge   = "334 VXNlcm5hbWU6" CRLF
auth_login_username_response    = username CRLF
auth_login_password_challenge   = "334 UGFzc3dvcmQ6" CRLF
auth_login_password_response    = password CRLF
```

The auth_login_command ABNF rule is consistent with the AUTH command syntax specified in [RFC4954], where the mechanism parameter is "LOGIN" and the initial-response parameter is the base64-encoded username.

3 Protocol Details

This extension defines both a client and server role. The choice of which roles to support is implementation-specific.<1>

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 that described in this document.

The client maintains the following state for a given connection to an **SMTP** server:

Username: The username provided by the application or higher-layer protocol.

Password: The password provided by the application or higher-layer protocol.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Initiating Authentication

When the client initiates authentication, it MUST compose an AUTH command that conforms to the auth_login_command **ABNF** rule, as specified in section 2.2.2. The client SHOULD include the **Username** (encoded with **base64 encoding**) in the command. It MAY<2> instead omit the **Username**.

3.1.5 Message Processing Events and Sequencing Rules

This extension does not change the message processing events or sequencing rules of messages specified in [RFC4954].

3.1.5.1 Receiving a Server Challenge

When the client receives a 334 response, as specified in [RFC4954] section 6, it SHOULD check whether the response matches the format specified by the auth_login_username_challenge or auth_login_password_challenge **ABNF** rules, as specified in section 2.2.2. If the response does not match either format, it SHOULD cancel the authentication, as specified in [RFC4954]. The client MAY<3> instead simply assume that the server challenges are in the proper format, according to the following rules:

- If the client omits the **Username** in the auth_login_command, the client assumes that the first server challenge matches the auth_login_username_challenge ABNF rule and any subsequent

server challenge matches the auth_login_password_challenge ABNF rule. The client MAY cancel the authentication if a third server challenge is received.

- If the client includes the **Username** in the auth_login_command, the client assumes that the first server challenge matches the auth_login_password_challenge ABNF rule. The client MAY cancel the authentication if a second server challenge is received.

In response to a challenge that matches the auth_login_username_challenge ABNF rule, the client MUST send a response that conforms to the auth_login_username_response ABNF rule with the **Username**, as specified in section 2.2.2.

In response to a challenge that matches the auth_login_password_challenge ABNF rule, the client MUST send a response that conforms to the auth_login_password_response ABNF rule with the **Password**, as specified in section 2.2.2.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The following state machine diagram illustrates the states used in the authentication process.

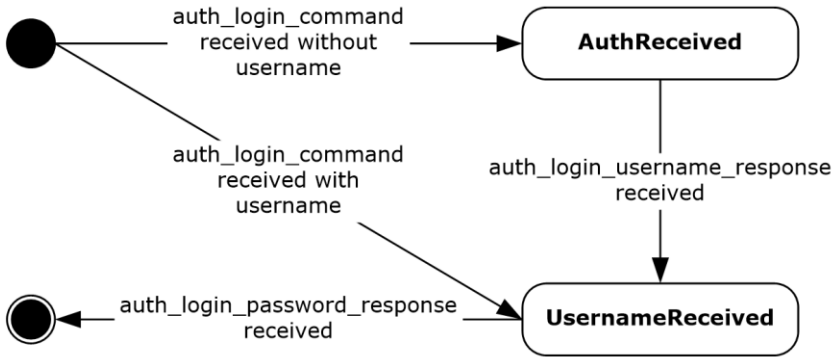

Figure 1: Server state machine diagram

3.2.1 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 that described in this document.

The server maintains the following global state:

List of SASL Mechanisms: The list of **SASL** mechanisms names to be returned in an EHLO response, as specified in [RFC5321].

For each connection from an **SMTP** client, the server has access to a set of authorized credentials consisting of a username and password. In addition, the server maintains the following state for each connection:

Substate: The state of the authentication, which can be either AuthReceived or UsernameReceived.

Username: The base64-encoded username value provided by the client.

3.2.2 Timers

None.

3.2.3 Initialization

When the server is initialized, it MUST place "LOGIN" in its **List of SASL Mechanisms** abstract data model element.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Processing AUTH LOGIN

When the server receives an AUTH command that conforms to the auth_login_command **ABNF** rule specified in section 2.2.2, it MUST respond according to the following rules.

1. If the username is not included in the command, the server MUST set the **Substate** to AuthReceived and send a response that conforms to the auth_login_username_challenge ABNF rule.
2. If the username is included in the command, the server MUST save the username in the **Username** associated with the connection, set the **Substate** to UsernameReceived, and send a response that conforms to the auth_login_password_challenge ABNF rule.

3.2.5.2 Processing a Request in the AuthReceived State

When the server receives a request in the AuthReceived state, the server MUST attempt to parse it according to the auth_login_username_response **ABNF** rule specified in section 2.2.2. The server MUST save the username in the **Username** associated with the connection, set the **Substate** to UsernameReceived, and send a response that conforms to the auth_login_password_challenge ABNF rule..

3.2.5.3 Processing a Request in the UsernameReceived State

When the server receives a request in the UsernameReceived state, the server MUST attempt to parse it according to the auth_login_password_response ABNF rule specified in section 2.2.2. The server MUST attempt to base64-decode the **Username** associated with the connection and the password included in the request and check that the **Username** corresponds to a valid user and that the password is a valid password for that user. The process of validating the **Username** and password is implementation-specific.

If the username and password are valid, the server MUST end the authentication by responding with a 235 response, as specified in [RFC4954] section 6. If the username or password is invalid, the server MUST end the authentication by responding with a 535 response, as specified in [RFC4954] section 6.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Example

The following is an example of the use of the AUTH LOGIN extension. The example demonstrates **SMTP** authentication using the AUTH LOGIN extension. In this example, the user name is "Charlie" and the password is "password". The following diagram illustrates the sequence of events following the client's initial connection to the SMTP server.

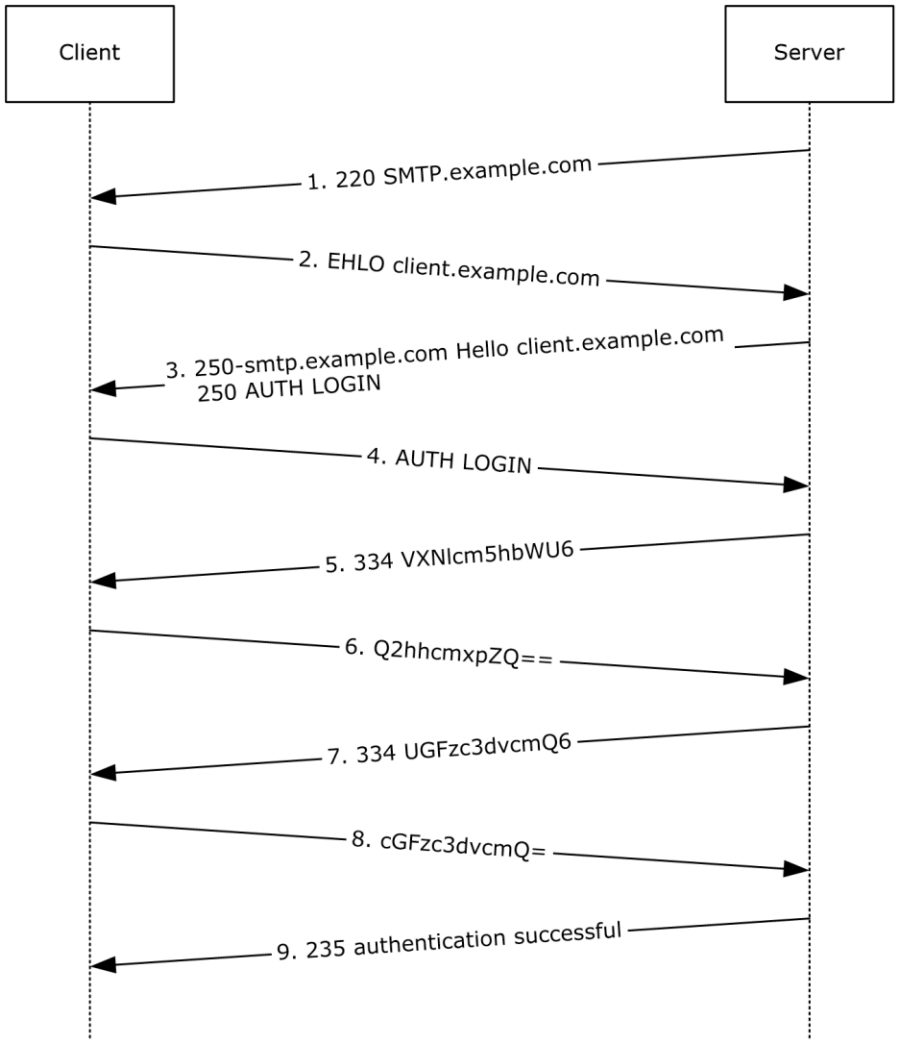

Figure 2: Example Authentication Exchange

1. The initial response by the SMTP server ("220 SMTP.example.com") is the greeting by the server as specified in [RFC5321].
2. The client sends the EHLO command.
3. The server responds with, among other things, an indication of support for AUTH LOGIN.
4. The client then issues the AUTH LOGIN command. In this example, the client omits the username in the AUTH LOGIN command.
5. The server responds with the username challenge.

6. The client responds with "Q2hhcmxpZQ==", which is the username "Charlie", encoded with **base64 encoding**.

7. The server stores the value "Q2hhcmxpZQ==" then issues the password challenge.

8. The client responds with "cGFzc3dvcmQ=", which is the password "password", encoded with base64 encoding.

9. The server base64-decodes the username and password and verifies that the username "Charlie" and the password "password" are valid credentials. The server then responds with "235 authentication successful".

5 Security

5.1 Security Considerations for Implementers

This extension offers no inherent security mechanisms to protect user credentials during authentication. Because of this, it is extremely important to only use this extension when also using a secure communication channel such as **Transport Layer Security (TLS)**, as specified in [RFC4346].

In environments where the use of TLS or other external security is mandated, it is strongly recommended that the AUTH LOGIN advertisement be suppressed until a secure channel is negotiated. TLS in particular exhibits this behavior where the **SMTP** session is restarted after TLS is negotiated.

5.2 Index of Security Parameters

Security parameter	Section
SASL mechanism name	section 2.2.1
Username	section 3.1.1
Password	section 3.1.1

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Microsoft Exchange Server 2003
- Microsoft Exchange Server 2007
- Microsoft Exchange Server 2010
- Microsoft Exchange Server 2013
- Microsoft Exchange Server 2016
- Microsoft Exchange Server 2019
- Microsoft Office Outlook 2003
- Microsoft Office Outlook 2007
- Microsoft Outlook 2010
- Microsoft Outlook 2013
- Microsoft Outlook 2016
- Microsoft Outlook 2019
- Microsoft .NET Framework 2.0
- Microsoft .NET Framework 3.5
- Microsoft .NET Framework 4
- Microsoft .NET Framework 4.5
- Microsoft .NET Framework 4.7
- Microsoft .NET Framework 4.8
- Windows 2000 Professional operating system
- Windows XP operating system
- Windows Vista operating system
- Windows 7 operating system
- Windows 8 operating system
- Windows 8.1
- Windows 10 operating system
- Windows 2000 Server operating system
- Windows Server 2003 operating system
- Windows Server 2008 operating system
- Windows Server 2012 operating system

- Windows Server 2012 R2
- Windows Server 2016 operating system
- Windows Server 2019 operating system
- Windows Server 2022 operating system
- Microsoft Outlook 2021
- Windows 11 operating system
- Microsoft Outlook 2024 Preview
- Windows Server 2025 operating system

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is 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 the product does not follow the prescription.

<1> Section 3: Exchange 2003, Exchange 2007, Exchange 2010, Exchange 2013, Exchange 2016, and Exchange 2019 only implement the server role. Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, Outlook 2016, Outlook 2019, .NET Framework 2.0, .NET Framework 3.5, .NET Framework 4, .NET Framework 4.5, Windows Vista, Windows 7, and Windows 8 only implement the client role. Windows 2000 Professional, Windows XP, Windows 2000 Server, Windows Server 2003, Windows Server 2008, and Windows Server 2012 implement both client and server roles.

<2> Section 3.1.4.1: Office Outlook 2003, Office Outlook 2007, Outlook 2010, Outlook 2013, Outlook 2016, Outlook 2019, and inetcomm.dll in Windows 2000 Professional, Windows XP, Windows Vista, Windows 7, Windows 8, Windows 2000 Server, Windows Server 2003, Windows Server 2008, and Windows Server 2012 do not include the username in the initial AUTH command.

<3> Section 3.1.5.1: .NET Framework 2.0, .NET Framework 3.5, .NET Framework 4, and .NET Framework 4.5 do not verify the syntax of 334 responses and instead keep state to remember whether it is the first server challenge or a subsequent server challenge.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Description	Revision class
6 Appendix A: Product Behavior	Updated list of supported products.	Major

8 Index