MQTT Version 3.1.1

OASIS 标准

OASIS 结构化信息标准促进组织（Organization for the Advancement of Structured Information Standards）

2014 年 10 月 29 日

Specification URIs

This version:

http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.doc (Authoritative)
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf

Previous version:

http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/cos01/mqtt-v3.1.1-cos01.doc (Authoritative)
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/cos01/mqtt-v3.1.1-cos01.html
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/cos01/mqtt-v3.1.1-cos01.pdf

Latest version:

http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.doc (Authoritative)
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html
http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.pdf

Technical Committee:

OASIS Message Queuing Telemetry Transport (MQTT) TC

Chairs:

Raphael J Cohn (raphael.cohn@stormmq.com), Individual
Richard J Coppen (coppen@uk.ibm.com), IBM

Editors:

Andrew Banks (Andrew_Banks@uk.ibm.com), IBM
Rahul Gupta (rahul.gupta@us.ibm.com), IBM

Related work:

This specification is related to:

- *MQTT and the NIST Cybersecurity Framework Version 1.0*. Edited by Geoff Brown and Louis-Philippe Lamoureux. Latest version: http://docs.oasis-open.org/mqtt/mqtt-nist-cybersecurity/v1.0/mqtt-nist-cybersecurity-v1.0.html.

Abstract:

MQTT 是一个终端和服务器之间发布/订阅消息的传输协议。它是重量轻，开放，简单，设计，以便易于实现。这些特点使它非常适合使用在许多情况下，包括受限的环境中，如通信在机器对机器（M2M）和物联网（IoT）的背景下，需要代码占用空间非常小并且带宽需要额外费用，带宽的使用量又非常大的场景。

MQTT is a Client Server publish/subscribe messaging transport protocol. It is light weight, open, simple, and designed so as to be easy to implement. These characteristics make it ideal for use in many situations, including constrained environments such as for communication in Machine to Machine (M2M) and Internet of Things (IoT) contexts where a small code footprint is required and/or network bandwidth is at a premium.

该协议运行在 TCP / IP，或在其他的网络协议，提供有序，无损，双向连接。其特点包括：

1、 使用发布/订阅消息模式，该模式提供一对多消息分发，并且与应用解耦。

2、 消息的传输，与消息体里面包含的内容无关.

3、 对于消息传输有三种 QoS：

"At most once"（最多一次），按照工作环境的最好效果，进行消息传输。例如环境传感器数据，不需要关心一个单次的阅读是否丢失，因为下一次发布很快就会到来。

"At least once"（至少一次），消息确保到达，但是重发会产生。

"Exactly once"（完全正确一次），消息确保正确到达一次。例如计费系统。

4、 传输开销小，交换协议最小化，降低网络占用。

5、 当一个异常发生时，会有机制通知利害关系方。

Status:

Citation format:

When referencing this specification the following citation format should be used:

[mqtt-v3.1.1]

MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta. 29 October 2014. OASIS Standard. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.

Notices

Table of Contents

1 Introduction 9
1.1 Organization of MQTT 9
1.2 Terminology ..... 错误!未定义书签。
1.3 Normative references 12
1.4 Non normative references 12
1.5 Data representations 14
1.5.1 Bits 15
1.5.2 Integer data values 15
1.5.3 UTF-8 encoded strings 15
1.6 Editing conventions 17
2 MQTT Control Packet format 18
2.1 Structure of an MQTT Control Packet 18
2.2 Fixed header 18
2.2.1 MQTT Control Packet type 18
2.2.2 Flags 19
2.2.3 Remaining Length 20
2.3 Variable header 22
2.3.1 Packet Identifier 22
2.4 Payload 23
3 MQTT Control Packets 25
3.1 CONNECT – Client requests a connection to a Server 25
3.1.1 Fixed header 25
3.1.2 Variable header 25
3.1.3 Payload 33
3.1.4 Response 35
3.2 CONNACK – Acknowledge connection request 35
3.2.1 Fixed header 35
3.2.2 Variable header 36
3.2.3 Payload 37
3.3 PUBLISH – Publish message 37
3.3.1 Fixed header 37
3.3.2 Variable header 43
3.3.3 Payload 44
3.3.4 Response 45
3.3.5 Actions 45
3.4 PUBACK – Publish acknowledgement 45
3.4.1 Fixed header 45
3.4.2 Variable header 46
3.4.3 Payload 46
3.4.4 Actions 46
3.5 PUBREC – Publish received (QoS 2 publish received, part 1) 46
3.5.1 Fixed header 46

3.5.2 Variable header 46
3.5.3 Payload 47
3.5.4 Actions 47
3.6 PUBREL – Publish release (QoS 2 publish received, part 2) 47
3.6.1 Fixed header 47
3.6.2 Variable header 47
3.6.3 Payload 48
3.6.4 Actions 48
3.7 PUBCOMP – Publish complete (QoS 2 publish received, part 3) 48
3.7.1 Fixed header 48
3.7.2 Variable header 48
3.7.3 Payload 49
3.7.4 Actions 49
3.8 SUBSCRIBE - Subscribe to topics 49
3.8.1 Fixed header 49
3.8.2 Variable header 49
3.8.3 Payload 50
3.8.4 Response 51
3.9 SUBACK – Subscribe acknowledgement 52
3.9.1 Fixed header 52
3.9.2 Variable header 53
3.9.3 Payload 53
3.10 UNSUBSCRIBE – Unsubscribe from topics 54
3.10.1 Fixed header 54
3.10.2 Variable header 54
3.10.3 Payload 54
3.10.4 Response 55
3.11 UNSUBACK – Unsubscribe acknowledgement 56
3.11.1 Fixed header 56
3.11.2 Variable header 56
3.11.3 Payload 56
3.12 PINGREQ – PING request 57
3.12.1 Fixed header 57
3.12.2 Variable header 57
3.12.3 Payload 57
3.12.4 Response 57
3.13 PINGRESP – PING response 57
3.13.1 Fixed header 58
3.13.2 Variable header 58
3.13.3 Payload 58
3.14 DISCONNECT – Disconnect notification 58
3.14.1 Fixed header 58
3.14.2 Variable header 58
3.14.3 Payload 58
3.14.4 Response 59

4 Operational behavior 60
4.1 Storing state 60
4.1.1 Non normative example 61
4.2 Network Connections 61
4.3 Quality of Service levels and protocol flows 62
4.3.1 QoS 0: At most once delivery 62
4.3.2 QoS 1: At least once delivery 63
4.3.3 QoS 2: Exactly once delivery 64
4.4 Message delivery retry 66
4.5 Message receipt 67
4.6 Message ordering 67
4.7 Topic Names and Topic Filters 68
4.7.1 Topic wildcards 68
4.7.2 Topics beginning with $ 70
4.7.3 Topic semantic and usage 71
4.8 Handling errors 72
5 Security 74
5.1 Introduction 74
5.2 MQTT solutions: security and certification 75
5.3 Lightweight cryptography and constrained devices 75
5.4 Implementation notes 75
5.4.1 Authentication of Clients by the Server 76
5.4.2 Authorization of Clients by the Server 76
5.4.3 Authentication of the Server by the Client 76
5.4.4 Integrity of Application Messages and Control Packets 77
5.4.5 Privacy of Application Messages and Control Packets 77
5.4.6 Non-repudiation of message transmission 78
5.4.7 Detecting compromise of Clients and Servers 78
5.4.8 Detecting abnormal behaviors 78
5.4.9 Other security considerations 79
5.4.10 Use of SOCKS 80
5.4.11 Security profiles 80
6 Using WebSocket as a network transport 82
6.1 IANA Considerations 82
7 Conformance 83
7.1 Conformance Targets 83
7.1.1 MQTT Server 83
7.1.2 MQTT Client 84
Appendix A. Acknowledgements (non normative) ..... 错误!未定义书签。
Appendix B. Mandatory normative statements (non normative) ..... 错误!未定义书签。
Appendix C. Revision history (non normative) ..... 错误!未定义书签。

Table of Figures and Tables

Figure 1.1 Structure of UTF-8 encoded strings 15
Figure 1.2 UTF-8 encoded string non normative example....14
Figure 2.1 – Structure of an MQTT Control Packet 18
Figure 2.2 - Fixed header format....18
Table 2.1 - Control packet types 18
Table 2.2 - Flag Bits....20
Table 2.4 Size of Remaining Length field....21
Figure 2.3 - Packet Identifier bytes....22
Table 2.5 - Control Packets that contain a Packet Identifier....22
Table 2.6 - Control Packets that contain a Payload 23
Figure 3.1 – CONNECT Packet fixed header....25
Figure 3.2 - Protocol Name bytes....25
Figure 3.3 - Protocol Level byte 26
Figure 3.4 - Connect Flag bits....26
Figure 3.5 Keep Alive bytes 31
Figure 3.6 - Variable header non normative example 32
Figure 3.7 - Password bytes 34
Figure 3.8 – CONNACK Packet fixed header 35
Figure 3.9 – CONNACK Packet variable header....36
Table 3.1 – Connect Return code values 37
Figure 3.10 – PUBLISH Packet fixed header 37
Table 3.2 - QoS definitions....38
Table 3.3 - Publish Packet non normative example 44
Figure 3.11 - Publish Packet variable header non normative example 44
Table 3.4 - Expected Publish Packet response....45
Figure 3.12 - PUBACK Packet fixed header 45
Figure 3.13 – PUBACK Packet variable header....46
Figure 3.14 – PUBREC Packet fixed header 46
Figure 3.15 – PUBREC Packet variable header 47
Figure 3.16 – PUBREL Packet fixed header 47
Figure 3.17 – PUBREL Packet variable header 47
Figure 3.18 – PUBCOMP Packet fixed header 48
Figure 3.19 – PUBCOMP Packet variable header 48
Figure 3.20 – SUBSCRIBE Packet fixed header....49
Figure 3.21 - Variable header with a Packet Identifier of 10, Non normative example 50
Figure 3.22 – SUBSCRIBE Packet payload format....50
Table 3.5 - Payload non normative example....50
Figure 3.23 - Payload byte format non normative example....51
Figure 3.24 – SUBACK Packet fixed header....52
Figure 3.25 – SUBACK Packet variable header....53
Figure 3.26 – SUBACK Packet payload format....53
Table 3.6 - Payload non normative example....53
Figure 3.27 - Payload byte format non normative example....54
Figure 3.28 – UNSUBSCRIBE Packet Fixed header 54
Figure 3.29 – UNSUBSCRIBE Packet variable header....54
Table3.7 - Payload non normative example....55

Figure 3.30 - Payload byte format non normative example......55
Figure 3.31 – UNSUBACK Packet fixed header......56
Figure 3.32 – UNSUBACK Packet variable header......56
Figure 3.33 – PINGREQ Packet fixed header......57
Figure 3.34 – PINGRESP Packet fixed header......58
Figure 3.35 – DISCONNECT Packet fixed header......58
Figure 4.1 – QoS 0 protocol flow diagram, non normative example......62
Figure 4.2 – QoS 1 protocol flow diagram, non normative example......64
Figure 4.3 – QoS 2 protocol flow diagram, non normative example......65
Figure 6.1 - IANA WebSocket Identifier......82

1 Introduction

1.1 Organization of MQTT

This specification is split into seven chapters:

- Chapter 1 - Introduction
- Chapter 2 - MQTT Control Packet format
- Chapter 3 - MQTT Control Packets
- Chapter 4 - Operational behavior
- Chapter 5 - Security
- Chapter 6 - Using WebSocket as a network transport
- Chapter 7 - Conformance Targets

1.2 术语

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119 [RFC2119].

1. MUST（必须：规范的绝对要求。） This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.

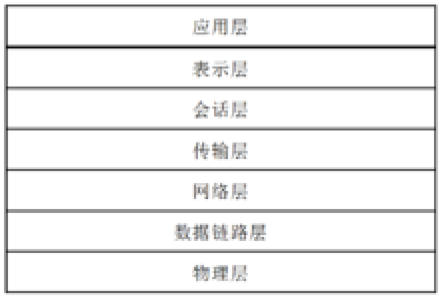
2. MUST NOT （不得：绝对禁止的规范。） This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.

3. SHOULD (应该：推荐) This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

4. SHOULD NOT （不应该:不推荐） This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.

Network Connection:（网络连接）

A construct provided by the underlying transport protocol that is being used by MQTT.
由 MQTT 提供构建由底层的传输协议。



应用层
表示层
会话层
传输层
网络层
数据链路层
物理层

TCP/IP	OSI
应用层	应用层 表示层 会话层
主机到主机层（TCP）（又称传输层）	传输层
网络层（IP）(又称互联层)	网络层
网络接口层（又称链路层）	数据链路层
	物理层

- It connects the Client to the Server.（连接终端到服务器）
- It provides the means to send an ordered, lossless, stream of bytes in both directions.（它提供了在两个方向上发送一个有序的、无损的、流的字节的方法）

应用消息：（**Application Message**）

通过 **MQTT** 协议传输的数据用于应用。当应用消息通过 **MQTT** 传输，消息都会有 **QoS** 和 Topic Name（抬头？）

终端（**Client**）**:**

一个程序或者器件使用 **MQTT** 的。终端时钟会建立网络链接到服务器。

他可以

1、发布其他终端需要的应用消息

2、订阅需要的应用消息。

3、通过退订，来去除应用消息的请求

4、从服务器断开

Server（服务器）:

一个程序或者器件作为一个中介，在两个终端之间，一个发布应用消息，一个订阅应用消息。

1、 接受一个来自终端的网络连接
2、 接受一个终端发布的应用消息。
3、 处理终端订阅和退订的请求。
4、 转发应用消息到终端订阅。

订阅（Subscription:）

订阅包含一个主题筛选器和一个最大 QoS。一个订阅与一个会话关联。一个会话可以包含一个订阅甚至更多。每个订阅在一个会话内，有一个不同的主题筛选器。

主题名称（Topic Name）:

这个应用消息附带的标签，对应的是服务器能够识别的订阅。服务器发送一个应用消息的拷贝到每个终端去匹配订阅。

主题过滤（Topic Filter）

一个订阅中包含的一个表达式，指示一个或者多个主题的关心内容。一个主题过滤器可以包含通配符。

通配符是一种特殊语句，主要有星号(*)和问号(?)，用来模糊搜索文件。当查找文件夹时，可以使用它来代替一个或多个真正字符；当不知道真正字符或者懒得输入完整名字时，常常使用通配符代替一个或多个真正的字符。 实际上用“*Not?paOd”可以对应 Notpad\MyNotpad【*可以代表任何文字】;Notpad\Notepad【?仅代表单个字】;Notepad\Notepod【ao 代表 a 与 o 里二选一】，其余以此类推。

通配符是竞价排名广告的一项高级功能，当我们在广告创意中使用了这项功能之后，使用不同搜索字词的用户将看到不同的广告创意(虽然我们只制作了一个广告)。这将大大提高我们广告的相关性和实用性，从而提高广告的点击率，同时也大大提高了我们的工作效率。

会话（Session）:

一个终端和服务器之间的状态交互。一些会话与网络链接时间同长，其他一些可以跨越多个服务器和 终端之间的链接 。

MQTT 控制包（MQTT Control Packet）:

一个发送在网络链接的信息包。MQTT 规范定义了十四种不同种类的控制包。用于传递应用消息。

1.3 Normative references（引用规范）

[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

[RFC3629]

Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003
http://www.ietf.org/rfc/rfc3629.txt

[RFC5246]

Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

http://www.ietf.org/rfc/rfc5246.txt

[RFC6455]

Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, December 2011.

http://www.ietf.org/rfc/rfc6455.txt

[Unicode]

The Unicode Consortium. The Unicode Standard.

http://www.unicode.org/versions/latest/

1.4 Non normative references（非规范引用）

[RFC793]
Postel, J. Transmission Control Protocol. STD 7, IETF RFC 793, September 1981.
http://www.ietf.org/rfc/rfc793.txt

[AES]

Advanced Encryption Standard (AES) (FIPS PUB 197).
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[DES]

Data Encryption Standard (DES).
http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[FIPS1402]

Security Requirements for Cryptographic Modules (FIPS PUB 140-2)
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

[IEEE 802.1AR]

IEEE Standard for Local and metropolitan area networks - Secure Device Identity
http://standards.ieee.org/findstds/standard/802.1AR-2009.html

[ISO29192]

ISO/IEC 29192-1:2012 Information technology -- Security techniques -- Lightweight cryptography -- Part 1: General
http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=56425

[MQTT NIST]

MQTT supplemental publication, MQTT and the NIST Framework for Improving Critical Infrastructure Cybersecurity
http://docs.oasis-open.org/mqtt/mqtt-nist-cybersecurity/v1.0/mqtt-nist-cybersecurity-v1.0.html

[MQTTV31]

MQTT V3.1 Protocol Specification.

http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html

[NISTCSF]

Improving Critical Infrastructure Cybersecurity Executive Order 13636
http://www.nist.gov/itl/upload/preliminary-cybersecurity-framework.pdf

[NIST7628]

NISTIR 7628 Guidelines for Smart Grid Cyber Security
http://www.nist.gov/smartgrid/upload/nistir-7628_total.pdf

[NSAB]

NSA Suite B Cryptography
http://www.nsa.gov/ia/programs/suiteb_cryptography/

[PCIDSS]

PCI-DSS Payment Card Industry Data Security Standard
https://www.pcisecuritystandards.org/security_standards/

[RFC1928]

Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and L. Jones, "SOCKS Protocol Version 5", RFC 1928, March 1996.

http://www.ietf.org/rfc/rfc1928.txt

[RFC4511]

Sermersheim, J., Ed., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006.

http://www.ietf.org/rfc/rfc4511.txt

[RFC5077]

Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.

http://www.ietf.org/rfc/rfc5077.txt

[RFC5280]

Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

http://www.ietf.org/rfc/rfc5280.txt

[RFC6066]

Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

http://www.ietf.org/rfc/rfc6066.txt

[RFC6749]

Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

http://www.ietf.org/rfc/rfc6749.txt

[RFC6960]

Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, June 2013.
http://www.ietf.org/rfc/rfc6960.txt

[SARBANES]

Sarbanes-Oxley Act of 2002.
http://www.gpo.gov/fdsys/pkg/PLAW-107publ204/html/PLAW-107publ204.htm

[USEUSAFEHARB]

U.S.-EU Safe Harbor
http://export.gov/safeharbor/eu/eg_main_018365.asp

1.5 Data representations（数据表达）

1.5.1 Bits（字节）

字节在一个“字”中标识为“bit7”~“bit0”。Bit7 是字节中的最高位，Bit0 是字节中的最低位。

1.5.2 Integer data values(整型数据值)

整数型数值，16bit 大端排序：高阶字节先于低阶字节。这个意味着，一个 16bit 的字，在网络中呈现为：“MSB”，跟随一个“LSB”。

1.5.3 UTF-8 encoded strings(UTF-8 解码字节)

控制包中的分组文本，被描述为“UTF-8 字节”。UTF-8 是一个高效编解码，是基于优化 ASCII 码特性，支持文本为基础的通信。

每一个字符串，都有两个子节点长度的空间作为前缀，描述 UTF-8 编码字符串自身的字节个数，通过下图 Figure1.1 UTF-8 编码字节结构描述。因此，传输字符串的大小有限制。传输的字符串大小不能超过 65535 bytes。

Each of these strings is prefixed with a two byte length field that gives the number of bytes in a UTF-8 encoded string itself, as illustrated in Figure 1.1 Structure of UTF-8 encoded strings below. Consequently there is a limit on the size of a string that can be passed in one of these UTF-8 encoded string components; you cannot use a string that would encode to more than 65535 bytes.

除非另有说明，否则所有 UTF-8 编码的字符串可以在范围 0 到 65535 字节的任何长度。

Figure 1.1 Structure of UTF-8 encoded strings

Bit	7	6	5	4	3	2	1	0
byte 1	String length MSB							
byte 2	String length LSB							
byte 3 ….	UTF-8 Encoded Character Data, if length > 0.							

UTF-8 编码字符串中的字符数据，必须是良好的 UTF-8 定义的统一编码规范。

特别是这个数据不包括 U + D800 和 U+ DFFF 间编码点的编码。如果一个服务器和终端接收到一个控制包的内容是一个错误格式的 UTF-8，必须关闭网络链接。

一个 UTF-8 编码字符串不可以包含一个空字符 U+000 的字符。如果接收到了一个这样的控制包，则必须关闭网络链接。

The character data in a UTF-8 encoded string MUST be well-formed UTF-8 as defined by the Unicode specification [Unicode] and restated in RFC 3629 [RFC3629]. In particular this data MUST NOT include encodings of code points between U+D800 and U+DFFF. If a Server or Client receives a Control Packet containing ill-formed UTF-8 it MUST close the Network Connection [MQTT-1.5.3-1].

A UTF-8 encoded string MUST NOT include an encoding of the null character U+0000. If a receiver (Server or Client) receives a Control Packet containing U+0000 it MUST close the Network Connection [MQTT-1.5.3-2].

数据不应该包含 Unicode 的编码，列在下面的。

U+0001..U+001F control characters
U+007F..U+009F control characters

如果一个接受者（终端或者服务器）接收一个控制包，包含他们（U+0001..U+001F、U+007F..U+009F），将关闭网络链接。

编码指示 定义在 Unicode 规范，中的非字符（例如：U+0FFFF）

Code points defined in the Unicode specification [Unicode] to be non-characters (for example U+0FFFF)

一个 UTF-8 编码序列“0xEF 0xBB 0xBF”一直被解读为“U+FEFF ("ZERO WIDTH NO-BREAK SPACE")”，无论他出现在哪一个字符串，都禁止被忽略，或者被一个包接收给剥离。

A UTF-8 encoded sequence 0xEF 0xBB 0xBF is always to be interpreted to mean U+FEFF ("ZERO WIDTH NO-BREAK SPACE") wherever it appears in a string and MUST NOT be skipped over or stripped off by a packet receiver [MQTT-1.5.3-3].

1.5.3.1 Non normative example（实例）

例如，一个字符 A，大写拉丁字母 A，后面跟一个 U+2A6D4(表示一个汉字扩展符 B)被解码为如下：

For example, the string A which is LATIN CAPITAL Letter A followed by the code point U+2A6D4 (which represents a CJK IDEOGRAPH EXTENSION B character) is encoded as follows:

Figure 1.2 UTF-8 encoded string non normative example

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	String Length MSB (0x00)							
	0	0	0	0	0	0	0	0
byte 2	String Length LSB (0x05)							
	0	0	0	0	0	1	0	1
byte 3	‘A’ (0x41)							
	0	1	0	0	0	0	0	1
byte 4	(0xF0)							
	1	1	1	1	0	0	0	0
byte 5	(0xAA)							
	1	0	1	0	1	0	1	0
byte 6	(0x9B)							
	1	0	0	1	1	0	1	1
byte 7	(0x94)							

	1	0	0	1	0	1	0	0

1.6 Editing conventions （编辑规范）

文字用黄色高亮在这个规范中，是标识一致性语句。每一个一致性语句都标识成这样的格式。

Text highlighted in Yellow within this specification identifies conformance statements. Each conformance statement has been assigned a reference in the format [MQTT-x.x.x-y].

2 MQTT Control Packet format（MQTT 控制包格式）

2.1 Structure of an MQTT Control Packet（MQTT 控制包的结构）

MQTT 协议工作通过交换一系列 MQTT 控制包，用已经定义好的方式。这个章节描述了这些包的格式。
一个 MQTT 控制包由三个部分组成，一直用下面顺序进行说明

Figure 2.1 – Structure of an MQTT Control Packet

Fixed header, present in all MQTT Control Packets 固定的头，展现在所有的 MQTT 控制包里面。
Variable header, present in some MQTT Control Packets 可变的头，有些 MQTT 控制包里面会出现。
Payload, present in some MQTT Control Packets 消息体（有效荷载），出现在有些 MQTT 控制包里面。

2.2 Fixed header（固定头）

Each MQTT Control Packet contains a fixed header. Figure 2.2 - Fixed header format illustrates the fixed header format.

每个 MQTT 控制包都包含一个固定头。

Figure 2.2 - Fixed header format（固定头格式）

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type MQTT 控制包类型				Flags specific to each MQTT Control Packet type 具体到每个 MQTT 控制包类型			
byte 2…	Remaining Length（剩余长度）							

2.2.1 MQTT Control Packet type(MQTT 控制包类型)

位置: byte 1, bits 7-4.

作为一个 4bit 无符号的值，这些值列下下面：

Table 2.1 - Control packet types

Name（名称）	**Value（值）**	**Direction of flow（数据流方向）**	**Description（描述）**
Reserved	0	Forbidden（禁止）	Reserved（保留）
CONNECT	1	Client to Server（终端	Client request to connect to Server（终端请求一个连接到服务器）

		到服务器）	
CONNACK	2	Server to Client （服务器到终端）	Connect acknowledgment(连接确认)
PUBLISH	3	Client to Server or Server to Client 双向	Publish message（发布消息）
PUBACK	4	Client to Server or Server to Client 双向	Publish acknowledgment（发布确认）
PUBREC	5	Client to Server or Server to Client	Publish received (assured delivery part 1) 发布接收（确定的传输第 1 部分）
PUBREL	6	Client to Server or Server to Client	Publish release (assured delivery part 2) 发布释放（确定的传输第 2 部分）
PUBCOMP	7	Client to Server or Server to Client	Publish complete (assured delivery part 3) 发布完成（确定的传输第 3 部分）
SUBSCRIBE	8	Client to Server	Client subscribe request 终端订阅请求
SUBACK	9	Server to Client	Subscribe acknowledgment 订阅确认
UNSUBSCRIBE	10	Client to Server	Unsubscribe request 退订请求
UNSUBACK	11	Server to Client	Unsubscribe acknowledgment 退订确认
PINGREQ	12	Client to Server	PING request PING 请求
PINGRESP	13	Server to Client	PING response PING 响应
DISCONNECT	14	Client to Server	Client is disconnecting 终端断开连接
Reserved	15	Forbidden	Reserved

2.2.2 Flags（标志）

Byte 1 的保留 bits[3-0]在固定消息头内容，详细描述每个 MQTT 控制包类型。在表格 2.2 中。

表格 2.2 中的“保留”值，是必须按照表格中设置，用于预留给未来使用的。如果无效的标志被接收了，接收者必须关闭网络连接。

查看章节 4.8，详细描述处理的错误。

Table 2.2 - Flag Bits

Control Packet	Fixed header flags	Bit 3	Bit 2	Bit 1	Bit 0
CONNECT	Reserved	0	0	0	0
CONNACK	Reserved	0	0	0	0
PUBLISH	Used in MQTT 3.1.1	DUP[1]	QoS[2]	QoS[2]	RETAIN[3]
PUBACK	Reserved	0	0	0	0
PUBREC	Reserved	0	0	0	0
PUBREL	Reserved	0	0	1	0
PUBCOMP	Reserved	0	0	0	0
SUBSCRIBE	Reserved	0	0	1	0
SUBACK	Reserved	0	0	0	0
UNSUBSCRIBE	Reserved	0	0	1	0
UNSUBACK	Reserved	0	0	0	0
PINGREQ	Reserved	0	0	0	0
PINGRESP	Reserved	0	0	0	0
DISCONNECT	Reserved	0	0	0	0

DUP[1] = Duplicate delivery of a PUBLISH Control Packet（一个“发布”控制包的重复传输）

QoS[2] = PUBLISH Quality of Service（“发布”的 QoS）

RETAIN[3] = PUBLISH Retain flag （“发布”的保留标志）

查看章节 3.3.1，对于 DUP、QoS 和 RETAIN 标识有详细描述。

2.2.3 Remaining Length（剩余长度）

Position: starts at byte 2.

位置：从 byte2 开始。

“剩余长度”（Remaining Length）是当前消息包中剩余的字节的个数，包含数据在“可变消息头”和消息体（payload）。剩余长度不包含用于编码剩余长度的字节（剩余长度，标称的是后续可变消息头和消息体的长度，不包括他自己的长度）。(The Remaining Length does not include the bytes used to encode the Remaining Length.)

剩余长度被编码，用一个变量长度编码方案，使用一个单字节值，达到 127。

更大的值，被处理成跟随的编码。每一个字节编码的最少的七位数据，并且最高 bit 是用来表示表达式中是否有后续的字节。（？）。因此每个字节编码都是 128 个值，和一个续位的值。剩余长度，最多的字节数是 4 个字节。

Non normative comment（举例）

例如，一个数字 64（十进制）用于编码成为一个单字节（十六进制是 0x40）。十进制数字 321（=65+2*128）被编码为两个字节。

第一个字节，是 65+128，标识为 193，注意最高位用于指示至少一个字节跟随；第二个字节标识为 2；

Non normative comment（举例）

允许应用发送控制包的大小达到 256MB（268,435,455）的尺寸。表达式是：0xFF, 0xFF, 0xFF, 0x7F

Table 2.4 表示了：剩余长度值标识字节数的增加。

Table 2.4 Size of Remaining Length field

Digits	From	To
1	0 (0x00)	127 (0x7F)
2	128 (0x80, 0x01)	16 383 (0xFF, 0x7F)
3	16 384 (0x80, 0x80, 0x01)	2 097 151 (0xFF, 0xFF, 0x7F)
4	2 097 152 (0x80, 0x80, 0x80, 0x01)	268 435 455 (0xFF, 0xFF, 0xFF, 0x7F)

Non normative comment（举例）

编码算法，对于一个无极性整数（X），变成一个长度编码：

```
    do
          encodedByte = X MOD 128   （求余）
          X = X DIV 128             （除法）
         // if there are more data to encode, set the top bit of this byte
         if ( X > 0 )               //////（X 大于 128）
             encodedByte = encodedByte OR 128   /////////（高位）
         endif
             'output' encodedByte
     while ( X > 0 )
```

MOD 求模运算，求余(% C 语言),

DIV 整数除(/ C 语言)

OR 按位或 (| C 语言).

Non normative comment

编码剩余长度的算法：

```
    multiplier = 1
    value = 0
    do
         encodedByte = 'next byte from stream'
         value += (encodedByte AND 127) * multiplier
```

```
                multiplier *= 128
                if (multiplier > 128*128*128)
                   throw Error(Malformed Remaining Length)
            while ((encodedByte AND 128) != 0)

```

AND 按位与运算(& C 语言).

当算法结束，value 包含剩余长度的值。

2.3 Variable header（可变头文件）

MQTT 控制包的一些类型包含可变消息头。它处于固定消息头和消息体的中间。可变头的内容取决于包类型。包定义可变头的区域在几个包类型中很常见。

The Packet Identifier field of variable header is common in several packet types.

2.3.1 Packet Identifier（包标识符）

Figure 2.3 - Packet Identifier bytes

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB（包标识高位）							
byte 2	Packet Identifier LSB（包标识低位）							

可变消息头包含很多控制包类型，包含 2 字节的包标识。这些控制包包含 2 字节包标识。这些控制包是：PUBLISH (where QoS > 0), PUBACK, PUBREC, PUBREL, PUBCOMP, SUBSCRIBE, SUBACK, UNSUBSCRIBE, UNSUBACK.

SUBSCRIBE, UNSUBSCRIBE, and PUBLISH (in cases where QoS > 0)控制包必须包含非零 16bit 包标识。每次一个终端发送一个新的数据包时，它必须将它分配给当前未使用的数据包标识符。

如果一个终端重发一个特殊的控制包，他必须使用相同的包标识，在订阅重发的包中。

在终端处理了相应的应答数据包之后，该数据包标识符可用于重用。

在 QoS1PUBLISH 的情况下，对应的是 PUBACK；在 QoS 2 情况下，是 PUBCOMP。对于 SUBSCRIBE 和退订，对应的是 SUBACK 或者 UNSUBACK。

同样的情况也适用于服务器，当他发送 PUBLISH，QoS>0 时。

如果一个 QoS 值为 0 的 PUBLISH 包，禁止包含一个包标识。

A PUBACK, PUBREC 或者 PUBREL 包必须包含一个相同的包标识，跟之前的 PUBLISH 包之前发送的一样。同样：SUBACK 和 UNSUBACK 必须包含包标识，与响应 SUBSCRIBE 和 UNSUBSCRIBE 也需要包标识相同。

控制包包含包标识的情况：

Table 2.5 - Control Packets that contain a Packet Identifier

控制包	包标识
CONNECT	NO
CONNACK	NO

PUBLISH	YES (If QoS > 0)
PUBACK	YES
PUBREC	YES
PUBREL	YES
PUBCOMP	YES
SUBSCRIBE	YES
SUBACK	YES
UNSUBSCRIBE	YES
UNSUBACK	YES
PINGREQ	NO
PINGRESP	NO
DISCONNECT	NO

终端和服务器独立地分配分组标识符。其结果是，终端-服务器对可以使用相同的分组标识符来参与并发信息交换。

Non normative comment（举例）

It is possible for a Client to send a PUBLISH Packet with Packet Identifier 0x1234 and then receive a different PUBLISH with Packet Identifier 0x1234 from its Server before it receives a PUBACK for the PUBLISH that it sent.

允许 Client 在收到自己发出的 Packet Identifier 0x1234 的 PUBACK 之前先接受一个服务器发来的 Packet Identifier 0x1234

```
Client                        Server
PUBLISH Packet Identifier=0x1234---→
←--PUBLISH Packet Identifier=0x1234
PUBACK Packet Identifier=0x1234---→
←--PUBACK Packet Identifier=0x1234
```

2.4 Payload（消息体）

一些 MQTT 控制包包含消息体，作为包的最后的部分，第三章会详细描述。这些情况下，PUBLISH 包是应用消息。

表格 2.6 列出了需要消息体的控制包。

Table 2.6 - Control Packets that contain a Payload

Control Packet	Payload
CONNECT	Required
CONNACK	None
PUBLISH	Optional
PUBACK	None

PUBREC	None
PUBREL	None
PUBCOMP	None
SUBSCRIBE	Required
SUBACK	Required
UNSUBSCRIBE	Required
UNSUBACK	None
PINGREQ	None
PINGRESP	None
DISCONNECT	None

381

3 MQTT Control Packets（MQTT 控制包）

3.1 CONNECT – Client requests a connection to a Server （CONNECT-终端发起一个连接到服务器）

在一个网络连接之后，第一个从客户端发送到服务器的数据包必须是一个 CONNECT 数据包。

一个终端只能发送一次 CONNECT 包。服务器如果收到第二次 CONNECT，作为一个协议违规，断开终端。查看章节 4.8 信息，关于处理错误的详细操作。

消息体包含一个或者多个编码空间。他们制定一个终端，制定了单一的终端标识，Will topic, Will Message, User Name and Password。

但是一个终端标识是可选的，他们的存在基于可变消息头的 FLAG 进行检查。

3.1.1 Fixed header（固定消息头）

Figure 3.1 – CONNECT Packet fixed header（CONNECT 包的固定消息头）

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (1)				Reserved			
	0	0	0	1	0	0	0	0
byte 2…	Remaining Length							

剩余长度位置 Remaining Length field

Remaining Length 是可变消息头（10 字节）的长度加上消息体的长度。编码方式在章节 2.2.3 中描述。

3.1.2 Variable header（可变消息头）

连接数据包的可变头包括以下顺序的四个字段：协议名称、协议级别、连接标志和保持活力。

3.1.2.1 Protocol Name（协议名称）

Figure 3.2 - Protocol Name bytes

	Description	7	6	5	4	3	2	1	0
Protocol Name									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (4)	0	0	0	0	0	1	0	0
byte 3	‘M’	0	1	0	0	1	1	0	1
byte 4	‘Q’	0	1	0	1	0	0	0	1
byte 5	‘T’	0	1	0	1	0	1	0	0

byte 6	‘T’	0	1	0	1	0	1	0	0

The Protocol Name is a UTF-8 encoded string that represents the protocol name “MQTT”, capitalized as shown. The string, its offset and length will not be changed by future versions of the MQTT specification.

协议名称是 UTF-8 编码字节，表现为协议名称“MQTT”，如图所示。这个字节，他的偏移量和长度不会随着未来 MQTT 的版本变化而变化。

如果协议名称错误，服务器可能会断开终端， 或者也可能用其他协议持续处理 CONNECT。在后一种情况，服务器不能继续处理与本规范相一致的 CONNECT 数据包。（这句有点不懂）

If the protocol name is incorrect the Server MAY disconnect the Client, or it MAY continue processing the CONNECT packet in accordance with some other specification. In the latter case, the Server MUST NOT continue to process the CONNECT packet in line with this specification [MQTT-3.1.2-1].

Non normative comment （举例）

Packet inspectors, such as firewalls, could use the Protocol Name to identify MQTT traffic.

分组检查，如防火墙，可以使用的协议名称识别 MQTT 交通。

3.1.2.2 Protocol Level（协议等级）

Figure 3.3 - Protocol Level byte

	Description	7	6	5	4	3	2	1	0
Protocol Level									
byte 7	Level(4)	0	0	0	0	0	1	0	0

8bit 无极性值，展现终端使用的协议修订级别。3.1.1 协议的数值是 4（0x04）。**如果服务器不支持该协议版本，Server 必须响应 CONNECT 包，回发一个 CONNACK 包，编码为 0x01（不可接受的协议级别），则服务器会断开链接。**

3.1.2.3 Connect Flags(链接标识)

链接标识字节包含大量参数用于描述 MQTT 的链接行为。他也指示这些在消息体中是否存在。

The Connect Flags byte contains a number of parameters specifying the behavior of the MQTT connection. It also indicates the presence or absence of fields in the payload.

Figure 3.4 - Connect Flag bits 链接标识 bits

Bit	7	6	5	4	3	2	1	0
	User Name Flag	Password Flag	Will Retain	Will QoS		Will Flag	Clean Session	Reserved
byte 8	X	X	X	X	X	X	X	0

Connect Flag，连接标识，有点像固定头部的。8 位分别代表不同的标志。第 1 个字节保留。

Clean Session,Will flag，Will Qos, Will Retain 都是相对于 CONNECT 消息来说的。

Clean Session:0 表示如果订阅的客户机断线了，那么要保存其要推送的消息，如果其重新连接时，则将这些消息推送。

1 表示消除，表示客户机是第一次连接，消息所以以前的连接信息。

Will Flag，表示如果客户机在不是在发送 DISCONNECT 消息中断，比如 IO 错误等，将些置为 1,要求重传。并且下且的 WillQos 和 WillRetain 也要设置，消息体中的 Topic 和 MessageID 也要设置，就是表示发生了错误，要重传。

Will Qos，在 CONNECT 非正常情况下设置，一般如果标识了 WillFlag，那么这个位置也要标识。

Will RETAIN: 同样在 CONNECT 中，如果标识了 WillFlag,那么些位也一定要标识

usename flag 和 passwordflag，用来标识是否在消息体中传递用户和密码，只有标识了，消息体中的用户名和密码才有效，只标记密码而不标记用户名是不合法的。

服务器必须验证 CONNECT 控制包的保留标识设置为零了，如果不为零，则断开该客户端

Clean Session（清除会话）

Position（位置）: bit 1 of the Connect Flags byte.（链接标识字节的 bit1）

改 bit 指示会话状态的处理。

客户端和服务器可以存储会话状态，以便在网络连接序列中继续进行可靠的消息传递。此位用于控制会话状态的寿命。

如果 CleanSession 被设为 0，则服务器必须保持与终端的通信，基于当前会话状态（由客户端标识符确定）。如果没有与客户端标识符相关联的会话，服务器必须创建一个新的会话。终端和服务器必须存储会话，在断开链接之后。在断开链接之后，服务器把 CleanSession 设置为 0.服务器必须存储另外 QoS1 和 QoS2 消息的匹配任何订阅，在断开的时间作为会话状态的一部分客户。也可能存储 QoS0 消息，用于相同的标准。

如果 CleanSession 设置为 1，则终端和服务器必须丢弃之前的会话，开始一个新的。Session 跟网络连接持续一样长时间。Session 状态数据禁止在订阅的 Session 中被重新使用。

If CleanSession is set to 1, the Client and Server MUST discard any previous Session and start a new one. This Session lasts as long as the Network Connection. State data associated with this Session MUST NOT be reused in any subsequent Session [MQTT-3.1.2-6].

Session 状态在在终端，由以下部分组成：

- QoS 1 and QoS 2 messages which have been sent to the Server, but have not been completely acknowledged.
- 终端发送到服务器的 QoS 1 和 QoS 2 消息，还没有被完全 “告知已经收到”。
- QoS 2 messages which have been received from the Server, but have not been completely acknowledged.
- 终端接收到的 QoS 2 消息，但是还没有被完全的告知“已经收到”

Session 状态在在终端，由以下部分组成：

- 即使是会话状态是空的，会话也可以存在。（The existence of a Session, even if the rest of the Session state is empty）
- 终端的订阅。
- 被发送到终端的 QoS 1 和 QoS 2 消息，但是还没有被确认已经接受。
- QoS 1 and QoS 2 消息正等待传输到终端
- 接受终端发过来的 QoS 2 消息，但是还没有完全确认已经被接受。
- 可选：QoS 0 消息正在传输到终端。

Retained 消息不能构成会话状态,他们禁止在会话结束后，被删除掉。

查看 4.1 章节，关于存储状态的详述和限制。

当 CleanSession 被设置为 1 时，终端和服务器不需要处理状态的自动删除

When CleanSession is set to 1 the Client and Server need not process the deletion of state atomically.

Non normative comment（举例）

To ensure consistent state in the event of a failure, the Client should repeat its attempts to connect with CleanSession set to 1, until it connects successfully.

为确保在发生故障时一致的状态，客户应重复尝试 cleansession 设置 1 连接，直到连接成功。

Non normative comment(举例)

一般情况下，终端连接一直使用 CleanSession 设置为 0，或者 CleanSession 设置为 1，不会在两个值之间变来变去。这个选择基于应用的。一个终端使用 CleanSession 设置为 1，将不会接受旧的应用消息，并且会订阅刷新标题，标题为他每次链接时新关注的。一个终端，使用 CleanSession 设置为 1,0，将接受当链接断开的时候所有发布的 QoS1 或者 QoS2 消息。因此，为

了确保你不会丢失消息，即使是链接断开的 时候，需要把 QoS1 或者 QoS2 的 CleanSession 设置为 0。

Non normative comment(举例)

当终端的 CleanSession 设置为 0 时，他相当于要求服务器在他断开之后仍然保持的他的会话状态。终端如果倾向于过一段时间会重复链接到服务器，链接时，只把 CleanSession 设置为 0。

如果后续不再链接，将会把 CleanSession 设置为 1，并且断开链接。

3.1.2.4 Will Flag（Will 标识）

位置：链接标识的 bit2

保持正常交流时，服务器特意发给客户端的消息。当客户端通过发送 DISCONNECT 消息正常断开时，Will 消息不会发送。

如果 Will Flag 设置为 1,指示 Connect 请求是否被接受，一个 Will 消息必须存储在服务器，并且与网络连接有关。当网络连接随后被关闭，Will 消息必须被发布。除非服务器响应 DISCONNECT 包，Will 消息被服务器删除。

Will 消息发布的情况包括，但不限于：

- 一个 IO 错误，或者网络失败被服务器删除。
- 在保持激活的时间内，一个终端尝试链接失败。
- 在没有先发送 DISCONNECT 包的前提下，终端关闭网络链接
- 由于协议错误，服务器关闭了网络连接。

补充：

will topic 和 will message 有点像立遗嘱。也即在连接服务器时通告：当我连接异常终止时请帮我发布这条 message 到相应的 topic。但要注意的是，will topic 和 will message 必须成对出现，并且还须设置 will flag。如果需要服务器保留这份遗嘱，则还需设置 will retain。

如果 WillFlag 标识设置为 1，则 Will Qos 和 Will Retain

If the Will Flag is set to 1, the Will QoS and Will Retain fields in the Connect Flags will be used by the Server, and the Will Topic and Will Message fields MUST be present in the payload [MQTT-3.1.2-9].

The Will Message MUST be removed from the stored Session state in the Server once it has been published or the Server has received a DISCONNECT packet from the Client [MQTT-3.1.2-10].

If the Will Flag is set to 0 the Will QoS and Will Retain fields in the Connect Flags MUST be set to zero and the Will Topic and Will Message fields MUST NOT be present in the payload [MQTT-3.1.2-11].

If the Will Flag is set to 0, a Will Message MUST NOT be published when this Network Connection ends [MQTT-3.1.2-12].

The Server SHOULD publish Will Messages promptly. In the case of a Server shutdown or failure the server MAY defer publication of Will Messages until a subsequent restart. If this happens there might be a delay between the time the server experienced failure and a Will Message being published.

3.1.2.5 Will QoS

位置：bit 4-3

Will QoS 标志用来设置当客户端异常离线时，服务器发送的 Will 消息的交付质量级别。Will 消息的内容在客户端发送的 CONNECT 消息里的有效载荷里填写。

这两个 bits 用于

Position: bits 4 and 3 of the Connect Flags.

These two bits specify the QoS level to be used when publishing the Will Message.

If the Will Flag is set to 0, then the Will QoS MUST be set to 0 (0x00) [MQTT-3.1.2-13].

If the Will Flag is set to 1, the value of Will QoS can be 0 (0x00), 1 (0x01), or 2 (0x02). It MUST NOT be 3 (0x03) [MQTT-3.1.2-14].

Position：bits 4 and 3 of the Connect flags byte.（连接标志字节的第4位和3位。）
当一个连接中的客户端被偶然的断开时为一个Will message定义在Will QoS字段中的QoS级别。Will message被定义在CONNECT message的payload中。（A connecting client specifies the QoS level in the
Will QoS field for a Will message that is sent in the event that the client is disconnected involuntarily. The Will message is defined in the payload of a CONNECT message.）
如果设置Will flag，Will QoS字段是强制性的，否则他的值将被忽略。

Will QoS 的值是 0（0x00），1（0x01），or 2（0x02）；禁止使用 3（0x03）

在当前版本的协议中这个字节的 0 位不使用。它是为将来使用保留的。

3.1.2.6 Will Retain

位置：Connect Flag（连接标识）字节的第 5 位

如果 Will 消息将要被保留，当 Will 发布时，这个 bit 就要指示。

如果 will Flag 设置为 0，则 will Retain Flag 必须设置为零。

如果 Will Flag 设置为 1：

　如果 Retain 被设置为 0，则服务器必须发布 Will Message，并且是一个 non-retained message

　如果 Retain 被设置为 1，则服务器必须发布 Will Message，并且是一个 retained message

If the Will Flag is set to 0, then the Will Retain Flag MUST be set to 0 [MQTT-3.1.2-15].

If the Will Flag is set to 1:

- If Will Retain is set to 0, the Server MUST publish the Will Message as a non-retained message [MQTT-3.1.2-16].
- If Will Retain is set to 1, the Server MUST publish the Will Message as a retained message [MQTT-3.1.2-17].

3.1.2.7 User Name Flag(用户名标识)

Position: bit 7 of the Connect Flags.

位置：Connect Flag 是的第七位；

如果 User Name Flag 设置为 0，则在消息体中，用户名禁止呈现。

如果 User Name Flag 设置为 1，则在消息体中，用户名必须呈现。

3.1.2.8 Password Flag

Position: bit 6 of the Connect Flags byte.

位置：Connect Flag 是的第 6 位；

如果 Password Flag 设置为 0，则在消息体中，密码名禁止呈现。

如果 Password Flag 设置为 1，则在消息体中，密码名必须呈现。

如果用户名标识是 0，则密码标识必须也是 0

如果设置User Name标志，User Name字段是强制性的，否则User Name的值是无效的，如果设置Password标志，Password字段是强制性的，否则Password的值是无效的。如果没有设置User Name，而提供Password是无效的。

3.1.2.9 Keep Alive

Figure 3.5 Keep Alive bytes

Bit	7	6	5	4	3	2	1	0
byte 9	Keep Alive MSB							
byte 10	Keep Alive LSB							

Keep Alive 是一个以一秒为单位的时间间隔。用 16bit 字标识，“终端完成传输一个控制包”和“开始传输下一个控制包”之间的最大的时间间隔。Keep Alive 是负责保证终端控制包发送不会超出 Keep Alive 的值。

以秒为单位，定义服务器端从客户端接收消息的最大时间间隔。一般应用服务会在业务层次检测客户端网络是否连接，不是 TCP/IP 协议层面的心跳机制(比如开启 SOCKET 的 SO_KEEPALIVE 选项)。 一般来讲，在一个心跳间隔内，客户端发送一个 PINGREQ 消息到服务器，服务器返回 PINGRESP 消息，完成一次心跳交互，继而等待下一轮。若客户端没有收到心跳反馈，会关闭掉 TCP/IP 端口连接，离线。 16 位

两个字节，可看做一个无符号的 short 类型值。最大值，2^16-1 = 65535 秒 = 18 小时。最小值可以为 0，表示客户端不断开。一般设为几分钟，比如微信心跳周期为 300 秒。

在没有发送其他控制包的时候，终端必须发送 PINGREG 包，保持心跳。该服务器得到 ping 请求后会发送一个 PINGRESP 消息。

终端可以在任意时刻发送 PINGREQ，无论 Keep Alive 值，使用 PINGRESP 去检查网络或者服务器是否在工作。

如果 Keep Alive 值是非零的，并且服务器在一个或者半个 Keep Alive 周期内不接受控制包，必须断开网络连接，认为网络连接已经失败。

如果终端在发送一个 PINGREG 之后，没有接受到 PINGRESP，他应该关闭网络连接到服务器。

保持非零 Keep Alive，会具有关闭 keep alive 机制的效果。意思是，在这种情况下，服务器不需要在闲置的情况下关闭网络。

注意：服务器检测到终端不响应，或者非活动的，则被允许断开终端。除非，终端在 KeepAlive 值内响应。

Non normative comment（范例）

Keep Alive 的实际值，是由应用程序指派的；特别是设置为几分钟。最大的值是 18 小时 12 分钟 15 秒。

3.1.2.10 Variable header non normative example（可变消息头，实例）

Figure 3.6 - Variable header non normative example（可变消息头，实例）

	Description	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Protocol Name									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (4)	0	0	0	0	0	1	0	0
byte 3	‘M’	0	1	0	0	1	1	0	1
byte 4	‘Q’	0	1	0	1	0	0	0	1
byte 5	‘T’	0	1	0	1	0	1	0	0
byte 6	‘T’	0	1	0	1	0	1	0	0
Protocol Level									
	Description	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 7	Level (4)	0	0	0	0	0	1	0	0
Connect Flags									

byte 8	User Name Flag (1) Password Flag (1) Will Retain (0) Will QoS (01) Will Flag (1) Clean Session (1) *Reserved* (0)	1	1	0	0	1	1	1	0
Keep Alive（10 秒）									
byte 9	Keep Alive MSB (0)	0	0	0	0	0	0	0	0
byte 10	Keep Alive LSB (10)（	0	0	0	0	1	0	1	0

3.1.3 Payload（消息体、有效载荷）

CONNECT 的消息体，包含一个或者多个长度前缀的空间，在可变头里面通过 Flag 进行检测。

这些内容，如果出现的话，必须通过终端 ID，Will 标题，Will 消息，用户名称，密码进行识别。

它们为客户端指定了一个唯一的标识符，Will Topic，message和User Name和密码使用

3.1.3.1 Client Identifier（终端 ID）

终端 ID（ClientId）标识，终端到服务器的唯一标识。ClientId 必须用于终端和服务器，标识装填，他们关联到终端和服务器之间的 MQTT 会话。ClientId 必须存在，必须在消息体的第一个位置。ClientId 必须是 UTF-8 编码，1.5.3 章节已经定义。服务器必须允许 ClientIDs 在 1 到 23 个字节之间，并且只包含字符 Char"0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"。服务器可能允许 ClientID 超过 23 个字节的编码。服务器可能允许 ClientID 包含字符不止以上这些。

第一个UTF编码的字符串。客户端标识符（Client ID）是介于1和23个字符长度，客户端到服务器的唯一标识。它必须在搜有客户端连接到一台服务器是唯一的，是在处理QoS级别1和2的消息ID中的关键。如果客户端ID包含23个字符，服务器响应CONNECT消息，通过一个CONNACK，返回码2：标识符被拒绝。

服务器可能允许一个终端提供一个 ClientID，长度为 0 字节，但是如果非零，则服务器必须作为一种特殊情况进行对待，同时分配一个唯一的 ClientID 给这个终端。必须处理这个独特的 ClientID 的 CONNECT 包。

如果终端提供一个 0 字节 ClientID，终端必须把 CleanSession 设置为 1。

如果终端提供一个 0 字节 ClientID，并且把 CleanSession 设置为 0，则服务器必须响应 CONNECT 一个 CONNACK 返回，返回码为 0x02（标识拒绝），同时关闭网络连接。

如果服务器拒绝 ClientId，一定是通过 CONNACK 0x02 (表示拒绝)，去响应这个 CONNECT 的

Non normative comment（实例）

一个客户端的实现可以产生一个随机 clientId 提供一个方便的方法。使用这种方法时，应积极鼓励 cleansession 设置为 0。

3.1.3.2 Will Topic

如果 Will Flag 设置为 1，则 Will Topic 就会在下一个位置。Will 标题必须是 UTF-8 编码字符串，在 1.5.3 章节已经描述了。

3.1.3.3 Will Message（Will 消息）

如果 Will 标识被设置为 1，will 消息将会出现在消息体中。Will 消息定义了 发布的应用消息。这个部分由两个字节长度组成。跟随消息体，will 消息是一个序列，0 个，或者多个字节。长度给出如下数据的字节数，不包括长度为 2 字节的字节数。

虽然在CONNECT消息中的Will Message是UTF编码的，当它发布Will Topic仅仅是消息的字节数被发送，而不是前两个字节的长度。因此，该消息必须只包含7位ASCII字符。

3.1.3.4 User Name（用户名）

如果 User Name Flag 设置为 1，则 UserName 用户名则会出现在消息体中。用户名必须是 UTF-8 编码。它可用于服务器进行身份验证和授权。

如果设置User Name标识，他是下一个UTF编码的字符串。User name标识连接的用户的名字，可用于身份验证。建议用户名为12个字符或者更少，但它不是必须的。请注意，为了兼容原来的MQTT V3规范，固定头的Remaining Length字段优先于User Name标识。
服务器必须允许实现设置User Name标识的可能性，但是User Name字符串正在缺少。这是有效的，应允许继续连接。

3.1.3.5 Password（密码）

如果密码 Flag 设置为 1，则 Password 一定出现在消息体中。Password 空间包含 0~65535 个字节，前两个字节用于描述长度。（这个长度不包含这两个标识长度的字节）

Figure 3.7 - Password bytes

Bit	7	6	5	4	3	2	1	0
byte 1	Data length MSB							
byte 2	Data length LSB							
byte 3 ….	Data, if length > 0.							

3.1.4 Response（响应）

注意，服务器可能支持多种协议（包括更早版本的 MQTT 协议）在相同 TCP 接口行，或者其他网络端点。如果服务器检查协议为 MQTT3.1.1，则他会验证尝试的链接如下。

1、如果服务器在网络建立之后一段时间内，没有接受到 CONNECT 包，则服务器应该关闭连接。

2、服务器应该验证 CONNECT 包，如果不符合要求，则不发送 CONNACK 并关闭网络。

3、服务器可能检查 CONNECT 包的内容，满足任何进一步的限制，并可进行身份验证和授权检查。如果这些检查可能失败，他可能发送一个适当的非零码的 CONNACK 响应（3.2 章节描述），并且必须关闭网络连接。

如果验证成功，服务器执行以下步骤。

1、如果 ClientID 代表一个终端，已经连接到服务器，则服务器必须断开这个已经存在的终端。

2、服务器必须处理一个 CleanSession。

3、服务器必须发送一个 CONNACK 包包含一个 0 返回码，通知终端已经收到 CONNECT 包。

4、开始消息传递，并且保持心跳监控。

终端被允许立即发送后续的控制包，在发送 CONNECT 包之后。

终端需要等待 CONNACK 包，从 Server 发过来的。如果服务器拒绝 CONNECT，他禁止处理终端发送的任何数据，在这个 CONNECT 之后的。

Non normative comment（实例）

终端一般会等待 CONNACK 包，但是如果终端在他接收到 CONNACK 之前，他利用他的自由去发送控制包。他可能简化终端实施，如同他不去监管链接状态一样。

终端接受这种情况，那些在他收到 CONNACK 包之前他发出去的数据，如果服务器拒绝链接，将不会被服务器处理。

3.2 CONNACK – Acknowledge connection request（通知收到连接请求）

CONNACK 包，是服务器发送的包，响应接收到终端的 CONNECT 包。第一个从服务器发送到终端的控制包，必须是 CONNACK 包。

如果终端没有接收到一个 CONNACK 包，在规定的时间内，终端应该关闭网络连接。一个“合理”的时间取决于应用程序和通信基础设施的类型。

3.2.1 Fixed header（固定消息头）

Figure 3.8 – CONNACK Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet Type (2)				Reserved			
	0	0	1	0	0	0	0	0
byte 2	Remaining Length (2)							

	0	0	0	0	0	0	1	0

Remaining Length field（剩余长度）

可变消息头的长度，对于 CONNACK 来说，固定是 2

3.2.2 Variable header（可变消息头）

Figure 3.9 – CONNACK Packet variable header

	Description	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Connect Acknowledge Flags		Reserved							SP[1]
byte 1		0	0	0	0	0	0	0	X
Connect Return code									
byte 2		X	X	X	X	X	X	X	X

3.2.2.1 Connect Acknowledge Flags

Byte 1 is the "Connect Acknowledge Flags". Bits 7-1 are reserved and MUST be set to 0.

Bit 0 (SP[1]) is the Session Present Flag.

字节 1 是链接应答指示，字节 7-1 保留，并且必须设置为 0.

字节 0（SP1）是会话展现标识。

3.2.2.2 Session Present （会话 出现）

Position: bit 0 of the Connect Acknowledge Flags.

位置：Connect Acknowledge Flags 的字节 0；

如果服务器接收一个连接，并且 CleanSession 设置为 1，则服务器必须在 CONNACK 设置 Session Present 为零，除此之外在 CONNACK 包中发送一个 0 返回码。

如果服务器接收一个 CleanSession 设置为 0 的连接，会话展现中设置的值取决于服务器是否存储会话状态用于提供终端 ID。如果服务器已经存储会话状态，并且他必须设置会话展现为 1，在 CONNACK 控制包中。

如果服务器不存储会话状态，则 Session Present 必须设置为 0。除了设置一个 0 返回码的 CONNACK 包。

This is in addition to setting a zero return code in the CONNACK packet

Session Present flag 使能一个终端去确认 终端和服务器之间是否有一个一致的结论，关于是否存储 Session 状态。

一旦一个 Session 的初始化设置完成，则一个 终端存储 Session 状态，将会期望服务器保持他会存储 Session 状态。在这个事件中，由终端接收的 Session Present 的值不是期望的值，终端选择处理 Session 或者断开链接。 终端可以丢弃 Session 状态，终端和服务器都断开链接。重新链接 Clean Session 设置为 1，然后再次断开。

如果服务器发送一个 CONNACK 包，包含一个非零的返回码，必须设置 Session Present 为 0。

3.2.2.3 Connect Return code（连接返回码）

在可变消息头中第二个字节。

如果一个服务器接收到一个良好格式的数据包，但是服务器由于一些原因不能处理，服务器应该尝试发送一个 CONNACK 包，非零返回码，在这个表格中。

如果服务器发送一个非零返回码，必须关闭网络连接。

Table 3.1 – Connect Return code values

Value	Return Code Response	Description
0	0x00 Connection Accepted	Connection accepted
1	0x01 Connection Refused, unacceptable protocol version	The Server does not support the level of the MQTT protocol requested by the Client
2	0x02 Connection Refused, identifier rejected	The Client identifier is correct UTF-8 but not allowed by the Server
3	0x03 Connection Refused, Server unavailable	The Network Connection has been made but the MQTT service is unavailable
4	0x04 Connection Refused, bad user name or password	The data in the user name or password is malformed
5	0x05 Connection Refused, not authorized	The Client is not authorized to connect
6-255		Reserved for future use

If none of the return codes listed in Table 3.1 – Connect Return code values are deemed applicable, then the Server MUST close the Network Connection without sending a CONNACK [MQTT-3.2.2-6].

如果没有返回码被认为是合适的，服务器将关闭网络，并且不发送 CONNACK。

3.2.3 Payload

CONNACK 没有 payload.

3.3 PUBLISH – Publish message（发布消息）

PUBLISH 控制包，由终端发送给服务器，或者服务器发送给终端，传输应用消息。

3.3.1 Fixed header（固定消息头）

Figure 3.10 – PUBLISH Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (3)				DUP flag	QoS level		RETAIN
	0	0	1	1	X	X	X	X
byte 2	Remaining Length							

3.3.1.1 DUP

Position: byte 1, bit 3.

位置：第一个字节的第三个 bit

如果 DUP 被设置为 0，则他指示：这是第一次，终端或者服务器尝试发送 MQTT PUBLISH 包。如果 DUP 设置为 1，则表示这可能是重传，或者之前传过数据包。

如果是重传，则 DUP 必须设置为 1。所有的 QoS 为 0 的消息，DUP 必须设置为 0。

当 PUBLISH 通过服务器传给订阅者，DUP 的值来自 PUBLISH 不会被传输。DUP 在外发的 PUBLISH 包中，独立于进入的 PUBLISH 包，他的值必须检查外发的 PUBLISH 包是不是重复传输。

Non normative comment（实例）

The recipient of a Control Packet that contains the DUP flag set to 1 cannot assume that it has seen an earlier copy of this packet.

控制包的接受者，包含 DUP 设置为 1，不能假定它已经看到了这个数据包的早期副本。

Non normative comment （实例）

重要内容：注意 DUP 参考控制包本身，而非控制包内含的应用消息。当使用 QoS1 时，终端发布的 PUBLISH 包里面的内容是之前他接收的，但是 DUP 标识是 0，并且带有不同的包标识。章节 2.3.1 提供更多的信息关于包标识。

3.3.1.2 QoS

位置：第一个字节，bit2-1。

这个空间指示应用消息的传输级别保证。

Table 3.2 - QoS definitions

QoS value	Bit 2	bit 1	Description
0	0	0	At most once delivery（最多一次）
1	0	1	At least once delivery（至少一次）
2	1	0	Exactly once delivery（至少正确交付一次）
-	1	1	Reserved – must not be used

PUBLISH 包禁止把所有的 QoS 的 bit 位设置为 1.如果服务器和终端接受一个 PUBLISH 包，把所有的 QoSbit 位设置为 1，会关闭网络连接。

QoS level 决定的消息流

QoS level 为 Quality of Service level 的缩写，翻译成中文，服务质量等级。

MQTT 3.1 协议在"4.1 Quality of Service levels and flows"章节中，仅仅讨论了客户端到服务器的发布流程，不太完整。因为决定消息到达率，能够提升发送质量的，应该是服务器发布 PUBLISH 消息到订阅者这一消息流方向。

QoS level 0

至多发送一次，发送即丢弃。没有确认消息，也不知道对方是否收到。

Client	Message and direction	Server
QoS = 0	PUBLISH ---------->	**Action:** Publish message to subscribers then Forget **Reception:** <=1

针对的消息不重要，丢失也无所谓。

网络层面，传输压力小。

QoS level 1

所有 QoS level 1 都要在可变头部中附加一个 16 位的消息 ID。

SUBSCRIBE 和 UNSUBSCRIBE 消息使用 QoS level 1。

针对消息的发布，**Qos level 1**，意味着消息至少被传输一次。

发送者若在一段时间内接收不到 PUBACK 消息，发送者需要打开 DUB 标记为 1，然后重新发送 PUBLISH 消息。因此会导致接收方可能会收到两次 PUBLISH 消息。针对客户端发布消息到服务器的消息流：

Client	Message and direction	Server

QoS = 1
DUP = 0
Message ID = x
Action: Store message
	PUBLISH
	---------->
		Actions:
		• Store message
		• Publish message to subscribers
		• Delete message
		Reception: >=1

Action: Discard message	PUBACK <----------	Message ID = x

针对服务器发布到订阅者的消息流：

Server	Message and direction	Subscriber

QoS = 1
DUP = 0
Message ID = x
	PUBLISH
	---------->
		Actions:
		• Store message
		• Make message available
		Reception: >=1

	PUBACK <----------	Message ID = x

发布者（客户端/服务器）若因种种异常接收不到 PUBACK 消息，会再次重新发送 PUBLISH 消息，同时设置 DUP 标记为 1。接收者以服务器为例，这可能会导致服务器收到重复消息，按照流程，broker（服务器）发布消息到订阅者（会导致订阅者接收到重复消息），然后发送一条 PUBACK 确认消息到发布者。

在业务层面，或许可以弥补 MQTT 协议的不足之处：重试的消息 ID 一定要一致接收方一定判断当前接收的消息 ID 是否已经接受过

但一样不能够完全确保，消息一定到达了。

QoS level 2

仅仅在 PUBLISH 类型消息中出现，要求在可变头部中要附加消息 ID。

级别高，通信压力稍大些，但确保了仅仅传输接收一次。

先看协议中流程图，Client -> Server 方向，会有一个总体印象：

Client	Message and direction	Server

QoS = 2
DUP = 0
Message ID = x
Action: Store message
	PUBLISH
	---------->
		Action(a) Store message
		or
		Actions(b):
		• Store message ID
		• Publish message to subscribers

	PUBREC <----------	Message ID = x
Message ID = x	PUBREL ---------->	**Actions(a):** • Publish message to subscribers • Delete message *or* **Action(b):** Delete message ID

Client	Message and direction	Server
Action: Discard message	PUBCOMP <-----------	Message ID = x

Server -> Subscriber：

Server	Message and direction	Subscriber

QoS = 2
DUP = 0
Message ID = x
	PUBLISH
	----------->
		Action: Store message

	PUBREC <-----------	Message ID = x
Message ID = x	PUBREL ----------->	**Actions:** • Make message available
	PUBCOMP <-----------	Message ID = x

Server 端采取的方案 a 和 b，都包含了何时消息有效，何时处理消息。两个方案二选一，Server 端自己决定。但无论死采取哪一种方式，都是在 QoS level 2 协议范畴下，不受影响。若一方没有接收到对应的确认消息，会从最近一次需要确认的消息重试，以便整个（QoS level 2）流程打通。

3.3.2 消息顺序

消息顺序会受许多因素的影响，但对于服务器程序，必须保证消息传递流程的每个阶段要和开始的顺序一致。例如，在 QoS level 2 定义的消息流中，PUBREL 流必须和 PUBLISH 流具有相同的顺序发送：

Client	Message and direction	Server
	PUBLISH 1 -----------> PUBLISH 2 -----------> PUBLISH 3 ----------->	
	PUBREC 1 <-----------	

Client	Message and direction	Server
	PUBREC 2 <----------	
	PUBREL 1 ---------->	
	PUBREC 3 <----------	
	PUBREL 2 ---------->	
	PUBCOMP 1 <----------	
	PUBREL 3 ---------->	
	PUBCOMP 2 <---------- PUBCOMP 3 <----------	

流动消息(in-flight messages)数量允许有一个可保证的效果：

- 在流动消息(in-flight)窗口 1 中，每个传递流在下一个流开始之前完成。这保证消息以提交的顺序传递
- 在流动消息(in-flight)大于 1 的窗口，只能在 QoS level 内被保证消息的顺序

3.3.2.1 RETAIN

Position: byte 1, bit 0.
位置：第一个字节的 bit0
retain
保持; 留在心中，记住; 雇用; 付定金保留;

如果 RETAIN 设置为 1，则 PUBLISH 包被终端发送到服务器，服务器必须存储应用消息和它的 QoS，所以他可以被传递到更远的订阅了该主题的订阅者。

当一个新的订阅被建立时，则最后保留的消息必须被发送到每个匹配的主题名称的订阅者。

如果服务器接收一个 QoS 为 0 的消息，并且 RETAIN 标识被设置为 1，则他必须抛弃所有的之前保留的主题。他应该存储新的 QoS0 消息，作为新的对应主题的保留消息，但是可能选择任何时刻丢弃他，入这个发生了，将会对于这个主题来说没有保留的消息。

查看章节 4.1，更多消息关于存储状态。

当服务器发送一个 PUBLISH 包到一个终端的时候，服务器必须设置 RETAIN 为 1，消息被作为一个新的订阅结果发送到服务器。当 PUBLISH 包被发送到终端时，必须设置 RETAIN 为 0，因为它会匹配建立一个订阅，无论他接收到的消息是如何设置的。

带有一个保留标志设置为 1 和一个包含零字节的有效载荷的发布包将被处理为正常的服务器，并发送给客户端与订阅匹配的主题名称。另外，任何具有相同主题名称的任何现有的保留消息必须被删除，并且该主题的任何未来的用户将无法接收保留消息。“正常”意味着在现有客户端接收的消息中保留标记,终端不设置。一个 0 字节的保留消息，禁止被存储。

由终端发送给服务器的 PUBLISH 包中如果 RETAIN 是 0，则服务器禁止存储消息，禁止移除或者替换已经有的保留消息。

Non normative comment（实例）

发布者用不规则的方式发送状态消息，Retained 消息是有用的。

一个新订阅者将会接收最近的状态。

Remaining Length field（保留长度）

可变消息头长度加上消息体长度。

消息的持久化

在 MQTT 协议中，PUBLISH 消息固定头部 RETAIN 标记，只有为 1 才要求服务器需要持久保存此消息，除非新的 PUBLISH 覆盖。

对于持久的、最新一条 PUBLISH 消息，服务器不但要发送给当前的订阅者，并且新的订阅者(new subscriber，同样需要订阅了此消息对应的 Topic name）会马上得到推送。

Tip：新来乍到的订阅者，只会取出最新的一个 RETAIN flag = 1 的消息推送，不是所有。

3.3.3 Variable header（可变消息头）

可变消息头包含：主题名称，包标识。

3.3.3.1 Topic Name(主题名称)

主题名称标识，负载数据发布的信息频道。（主题名称标识消息体数据发布的信息通道。）

The Topic Name identifies the information channel to which payload data is published.

消息标题名称，必须在 PUBLISH 包的可变头文件的第一个位置。他必须是 UTF-8 的编码，在章节 1.5.3 已经定义。

PUBLISH 包的主题名称禁止包含通配符字符。

PUBLISH 包的主题名称由服务器发送给订阅的终端，必须匹配订阅的主题过滤器，根据匹配关系处理。

但是，由于服务器允许主题重名，所以这个主题名称可能不是最原始 PUBLISH 包里面的相同主题名称。

3.3.3.2 Packet Identifier（包标识）

在 PUBLISH 包中，QoS 的级别是 1 或者 2，则包标识才会存在。章节 2.3.1 提供更多型信息关于包标识。

3.3.3.3 Variable header non normative example（可变消息头实例）

Table 3.3 - Publish Packet non normative example（PUBLISH 包实例）

Field	Value
Topic Name	a/b
Packet Identifier	10

Figure 3.11 - Publish Packet variable header non normative example（PUBLISH 包可变消息头实例）

	Description	7	6	5	4	3	2	1	0
Topic Name									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (3)	0	0	0	0	0	0	1	1
byte 3	‘a’ (0x61)	0	1	1	0	0	0	0	1
byte 4	‘/’ (0x2F)	0	0	1	0	1	1	1	1
byte 5	‘b’ (0x62)	0	1	1	0	0	0	1	0
Packet Identifier									
byte 6	Packet Identifier MSB (0)	0	0	0	0	0	0	0	0
byte 7	Packet Identifier LSB (10)	0	0	0	0	1	0	1	0

3.3.4 Payload（消息体）

消息体包含要发布的应用消息。数据的内容和格式是特定于应用程序的。消息体的长度可以通过的在固定头上的剩余长度字段减去可变头的长度来计算。PUBLISH 包包含一个 0 长度的消息体是有效的。

3.3.5 Response（响应）

Table 3.4 - Expected Publish Packet response

QoS Level	Expected Response
QoS 0	None
QoS 1	PUBACK Packet
QoS 2	PUBREC Packet

3.3.6 Actions（行动）

终端使用 PUBLISH 包发送应用程序到服务器，用于分发到其他匹配订阅了的终端。

服务器使用 PUBLISH 包，发送应用消息到每个匹配订阅的终端。另外，服务器可能传输更多的消息副本，

当终端订阅了话题过滤器，包含通配符，可能出现的情况是：终端的订阅可能重复，发布的消息可能匹配多个过滤器。在这种情况下，服务器必须传输消息到终端尊从所有匹配的订阅器中最大的 QoS，所有的情况都是遵从订阅者的 QoS。

收件者的动作取决于 QoS 的级别，在章节 4.3 中描述。

如果服务器不允许授权一个由终端执行的 PUBLISH；服务器没有办法通知终端。它必须是一个积极的接收确认，根据正常的 QoS 规则，或关闭网络连接。

3.4 PUBACK – Publish acknowledgement （PUBLISH 接收确认）

PUBACK 包是 QoS=1 的 PUBLISH 包应答。

3.4.1 Fixed header（固定消息头）

Figure 3.12 - PUBACK Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (4)				Reserved			
	0	1	0	0	0	0	0	0
byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

Remaining Length field（剩余长度）

This is the length of the variable header. For the PUBACK Packet this has the value 2.

可变消息头的长度，对于 PUBACK 来说，一直是 2。

3.4.2 Variable header（可变消息头）

This contains the Packet Identifier from the PUBLISH Packet that is being acknowledged.
包含包 PUBLISH 包 ID 被接收确认

Figure 3.13 – PUBACK Packet variable header

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3.4.3 Payload（消息体）

The PUBACK Packet has no payload.
PUBACK 包没有消息体。

3.4.4 Actions（行动）

This is fully described in Section 4.3.2.
完整表述在章节 4.3.2

3.5 PUBREC – Publish received (QoS 2 publish received, part 1)

PUBREC（QoS 为 2PUBLISH 消息的接收确认，第一部分）

A PUBREC Packet is the response to a PUBLISH Packet with QoS 2. It is the second packet of the QoS 2 protocol exchange.
PUBREC 包是 QoS 为 2 的 PUBLISH 包的响应。这是 QoS 为 2 的协议交换的第二包

3.5.1 Fixed header（固定消息头）

Figure 3.14 – PUBREC Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (5)				Reserved			
	0	1	0	1	0	0	0	0
byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

Remaining Length field

This is the length of the variable header. For the PUBREC Packet this has the value 2.
可变消息头的长度，对于 PUBREC 来说，一直是 2。

3.5.2 Variable header（可变消息头）

PUBLISH 被接收确认的包 ID

Figure 3.15 – PUBREC Packet variable header

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3.5.3 Payload(消息体)

没有消息体

3.5.4 Actions（动作）

完整描述在章节 4.3.3.

3.6 PUBREL – Publish release (QoS 2 publish received, part 2)

PUBREL-PUBLISH 释放（QoS=2 的 PUBLISH 接收到，第二部分）

PUBREL 包是 PUBREC 的相应。他是 QoS2 协议交换的第三个包。

3.6.1 Fixed header（固定消息头）

Figure 3.16 – PUBREL Packet fixed header

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	MQTT Control Packet type (6)				Reserved			
	0	1	1	0	0	0	1	0
byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

PUBREL 控制包的**固定消息头的 Bit3，2,1,0 被保留，并且必须设置为 0010。服务器必须处理任何其他值为畸形和关闭网络连接。**

Remaining Length field（剩余长度）

This is the length of the variable header. For the PUBREL Packet this has the value 2.

剩余长度都是 2，就是可变消息头的长度

3.6.2 Variable header（可变消息头）

The variable header contains the same Packet Identifier as the PUBREC Packet that is being acknowledged.

可变消息头包含相同的包 ID 和接收到的 PUBREC 保持一直

Figure 3.17 – PUBREL Packet variable header

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

byte 1	Packet Identifier MSB
byte 2	Packet Identifier LSB

3.6.3 Payload（消息体）

没有消息体.

3.6.4 Actions（动作）

完整描述在章节 4.3.3.

3.7 PUBCOMP – Publish complete (QoS 2 publish received, part 3)

PUBCOMP-PUBLISH 完成（QoS=2 的 PUBLISH 接收的第三部分）

The PUBCOMP Packet is the response to a PUBREL Packet. It is the fourth and final packet of the QoS 2 protocol exchange.

PUBCOMP 是响应 PUBREL 的。是 QoS=2 协议交换的第四包，也是最后一个。

3.7.1 Fixed header（固定头）

Figure 3.18 – PUBCOMP Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (7)				Reserved			
	0	1	1	1	0	0	0	0
byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

Remaining Length field

This is the length of the variable header. For the PUBCOMP Packet this has the value 2.

剩余长度就是可变消息头，对于 PUBCOMP 剩余长度一直是 2.

3.7.2 Variable header（可变消息头）

The variable header contains the same Packet Identifier as the PUBREL Packet that is being acknowledged.（可变消息头包含相同包 ID 和接收到的 PUBREL 保持一致）

Figure 3.19 – PUBCOMP Packet variable header

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3.7.3 Payload（消息体）

The PUBCOMP Packet has no payload.（没有消息体）

3.7.4 Actions

This is fully described in Section 4.3.3.（章节 4.3.3 完整描述）

3.8 SUBSCRIBE - Subscribe to topics（SUBSCRIBE-订阅主题）

The SUBSCRIBE Packet is sent from the Client to the Server to create one or more Subscriptions. Each Subscription registers a Client’s interest in one or more Topics. The Server sends PUBLISH Packets to the Client in order to forward Application Messages that were published to Topics that match these Subscriptions. The SUBSCRIBE Packet also specifies (for each Subscription) the maximum QoS with which the Server can send Application Messages to the Client.

订阅数据包从终端发送到服务器，以创建一个或多个订阅每个订阅注册一个终端的关注一个主题或者多个主题。服务器向终端发送发布数据包，以便将发布到匹配这些订阅内容的主题中的应用程序消息发送给终端。SUBSCRIBE 包对于每个订阅来说，指定最大的 QoS，服务器可以发送应用消息到终端。

3.8.1 Fixed header（固定消息头）

Figure 3.20 – SUBSCRIBE Packet fixed header

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	MQTT Control Packet type (8)				Reserved			
	1	0	0	0	0	0	1	0
byte 2	Remaining Length							

Bits 3,2,1 and 0 of the fixed header of the SUBSCRIBE Control Packet are reserved and MUST be set to 0,0,1 and 0 respectively. The Server MUST treat any other value as malformed and close the Network Connection [MQTT-3.8.1-1].

SUBSCRIBE 固定消息头的 Bit3、2、1、0 四位保留，必须设置为 0010.服务器必须把其他值认为是畸形值，并且关闭网络。

Remaining Length field（剩余长度）

This is the length of variable header (2 bytes) plus the length of the payload.

可变消息头 2 个字节，加上消息体的长度。

3.8.2 Variable header

The variable header contains a Packet Identifier. Section 2.3.1 provides more information about Packet Identifiers.

可变消息头包含包 ID，章节 2.3.1 提供更多信息关于包 ID。

3.8.2.1 Variable header non normative example（可变消息头实例）

Figure 3.21 shows a variable header with Packet Identifier set to 10.（包 ID 设置为 10）

Figure 3.21 - Variable header with a Packet Identifier of 10, Non normative example

	Description	7	6	5	4	3	2	1	0
Packet Identifier									
byte 1	Packet Identifier MSB (0)	0	0	0	0	0	0	0	0
byte 2	Packet Identifier LSB (10)	0	0	0	0	1	0	1	0

3.8.3 Payload（消息体）

SUBSCRIBE 包的消息体包含主题过滤器指示主题，终端想订阅的主题。主题过滤器在消息体中必须是 UTF-8 编码，在章节 1.5.3 定义。一个服务器应该支持带有通配符的主题过滤器，在章节 4.7.1 中定义。如果他选择不支持带通配符的主题过滤器，服务器必须拒绝所有包含通配符的过滤器。每个过滤器都遵循一个叫做 Requested QoS 的字节。Requested QoS 字节给出服务器可以发送应用消息到终端的最大的 QoS 的级别。查看 4.8 章节，查看处理错误的信息。

需要的最大的 QoS 编码在一个字节中，跟随着每个 UTF-8 编码的主题名称后面，这些 Topic Filter / QoS 对，连续打包。

Figure 3.22 – SUBSCRIBE Packet payload format

Description	7	6	5	4	3	2	1	0
Topic Filter								
byte 1	Length MSB							
byte 2	Length LSB							
bytes 3..N	Topic Filter							
Requested QoS								
	Reserved						QoS	
byte N+1	0	0	0	0	0	0	X	X

在当前的协议规范中，Requested QoS 的高 6bit 不使用。保留，留以后用；如果保留字节非零，或者 QoS 不是 0,1,2 三个数字，则服务器必须处理 SUBSCRIBE 作为畸形包，然后关闭网络连接。

3.8.3.1 Payload non normative example（消息体实例）

Table 3.5 - Payload non normative example

Topic Name	“a/b”
Requested QoS	0x01
Topic Name	“c/d”
Requested QoS	0x02

Figure 3.23 - Payload byte format non normative example

	Description	7	6	5	4	3	2	1	0
Topic Filter									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (3)	0	0	0	0	0	0	1	1
byte 3	‘a’ (0x61)	0	1	1	0	0	0	0	1
byte 4	‘/’ (0x2F)	0	0	1	0	1	1	1	1
byte 5	‘b’ (0x62)	0	1	1	0	0	0	1	0
Requested QoS									
byte 6	Requested QoS(1)	0	0	0	0	0	0	0	1
Topic Filter									
byte 7	Length MSB (0)	0	0	0	0	0	0	0	0
byte 8	Length LSB (3)	0	0	0	0	0	0	1	1
byte 9	‘c’ (0x63)	0	1	1	0	0	0	1	1
byte 10	‘/’ (0x2F)	0	0	1	0	1	1	1	1
byte 11	‘d’ (0x64)	0	1	1	0	0	1	0	0
Requested QoS									
byte 12	Requested QoS(2)	0	0	0	0	0	0	1	0

3.8.4 Response（响应）

当服务器接收到来自终端的 SUBSCRIBE 包，必须响应一个 SUBACK 包。SUBACK 包必须包含相同的包 ID 和 SUBSCRIBE 保持一致，并且他是确认收到。

服务器允许开始传输 PUBLISH 包，匹配订阅，在服务器发送 SUBACK 之前。

如果服务器接收到一个 SUBSCRIBE 包，SUBSCRIBE 包含标题过滤器，

If a Server receives a SUBSCRIBE Packet containing a Topic Filter that is identical to an existing Subscription’s Topic Filter then it MUST completely replace that existing Subscription with a new Subscription. The Topic Filter in the new Subscription will be identical to that in the previous Subscription, although its maximum QoS value could be different. Any existing retained messages matching the Topic Filter MUST be re-sent, but the flow of publications MUST NOT be interrupted [MQTT-3.8.4-3].

Where the Topic Filter is not identical to any existing Subscription’s filter, a new Subscription is created and all matching retained messages are sent.

If a Server receives a SUBSCRIBE packet that contains multiple Topic Filters it MUST handle that packet as if it had received a sequence of multiple SUBSCRIBE packets, except that it combines their responses into a single SUBACK response [MQTT-3.8.4-4].

The SUBACK Packet sent by the Server to the Client MUST contain a return code for each Topic Filter/QoS pair. This return code MUST either show the maximum QoS that was granted for that Subscription or indicate that the subscription failed [MQTT-3.8.4-5]. The Server might grant a lower maximum QoS than the subscriber requested. The QoS of Payload Messages sent in response to a Subscription MUST be the minimum of the QoS of the originally published message and the maximum QoS granted by the Server. The server is permitted to send duplicate copies of a message to a subscriber in the case where the original message was published with QoS 1 and the maximum QoS granted was QoS 0 [MQTT-3.8.4-6].

Non normative examples

If a subscribing Client has been granted maximum QoS 1 for a particular Topic Filter, then a QoS 0 Application Message matching the filter is delivered to the Client at QoS 0. This means that at most one copy of the message is received by the Client. On the other hand a QoS 2 Message published to the same topic is downgraded by the Server to QoS 1 for delivery to the Client, so that Client might receive duplicate copies of the Message.

If the subscribing Client has been granted maximum QoS 0, then an Application Message originally published as QoS 2 might get lost on the hop to the Client, but the Server should never send a duplicate of that Message. A QoS 1 Message published to the same topic might either get lost or duplicated on its transmission to that Client.

Non normative comment

Subscribing to a Topic Filter at QoS 2 is equivalent to saying "I would like to receive Messages matching this filter at the QoS with which they were published". This means a publisher is responsible for determining the maximum QoS a Message can be delivered at, but a subscriber is able to require that the Server downgrades the QoS to one more suitable for its usage.

3.9 SUBACK – Subscribe acknowledgement

A SUBACK Packet is sent by the Server to the Client to confirm receipt and processing of a SUBSCRIBE Packet.

A SUBACK Packet contains a list of return codes, that specify the maximum QoS level that was granted in each Subscription that was requested by the SUBSCRIBE.

3.9.1 Fixed header

Figure 3.24 – SUBACK Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (9)				Reserved			
	1	0	0	1	0	0	0	0
byte 2	Remaining Length							

Remaining Length field

This is the length of variable header (2 bytes) plus the length of the payload.

3.9.2 Variable header

The variable header contains the Packet Identifier from the SUBSCRIBE Packet that is being acknowledged. Figure 3.25 - variable header format below illustrates the format of the variable header.

Figure 3.25 – SUBACK Packet variable header

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3.9.3 Payload

The payload contains a list of return codes. Each return code corresponds to a Topic Filter in the SUBSCRIBE Packet being acknowledged. The order of return codes in the SUBACK Packet MUST match the order of Topic Filters in the SUBSCRIBE Packet [MQTT-3.9.3-1].

Figure 3.26 - Payload format below illustrates the Return Code field encoded in a byte in the Payload.

Figure 3.26 – SUBACK Packet payload format

Bit	7	6	5	4	3	2	1	0
	Return Code							
byte 1	X	0	0	0	0	0	X	X

Allowed return codes:

0x00 - Success - Maximum QoS 0
0x01 - Success - Maximum QoS 1
0x02 - Success - Maximum QoS 2
0x80 - Failure

SUBACK return codes other than 0x00, 0x01, 0x02 and 0x80 are reserved and MUST NOT be used [MQTT-3.9.3-2].

3.9.3.1 Payload non normative example

Figure 3.27 - Payload byte format non normative example shows the payload for the SUBACK Packet briefly described in Table 3.6 - Payload non normative example.

Table 3.6 - Payload non normative example

Success - Maximum QoS 0	0
Success - Maximum QoS 2	2
Failure	128

Figure 3.27 - Payload byte format non normative example

	Description	7	6	5	4	3	2	1	0
byte 1	Success - Maximum QoS 0	0	0	0	0	0	0	0	0
byte 2	Success - Maximum QoS 2	0	0	0	0	0	0	1	0
byte 3	Failure	1	0	0	0	0	0	0	0

3.10 UNSUBSCRIBE – Unsubscribe from topics

An UNSUBSCRIBE Packet is sent by the Client to the Server, to unsubscribe from topics.

3.10.1 Fixed header

Figure 3.28 – UNSUBSCRIBE Packet Fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (10)				Reserved			
	1	0	1	0	0	0	1	0
byte 2	Remaining Length							

Bits 3,2,1 and 0 of the fixed header of the UNSUBSCRIBE Control Packet are reserved and MUST be set to 0,0,1 and 0 respectively. The Server MUST treat any other value as malformed and close the Network Connection [MQTT-3.10.1-1].

Remaining Length field

This is the length of variable header (2 bytes) plus the length of the payload.

3.10.2 Variable header

The variable header contains a Packet Identifier. Section 2.3.1 provides more information about Packet Identifiers.

Figure 3.29 – UNSUBSCRIBE Packet variable header

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3.10.3 Payload

The payload for the UNSUBSCRIBE Packet contains the list of Topic Filters that the Client wishes to unsubscribe from. The Topic Filters in an UNSUBSCRIBE packet MUST be UTF-8 encoded strings as defined in Section 1.5.3, packed contiguously [MQTT-3.10.3-1].

The Payload of an UNSUBSCRIBE packet MUST contain at least one Topic Filter. An UNSUBSCRIBE packet with no payload is a protocol violation [MQTT-3.10.3-2]. See section 4.8 for information about handling errors.

3.10.3.1 Payload non normative example

Figure 3.30 - Payload byte format non normative example show the payload for the UNSUBSCRIBE Packet briefly described in Table3.7 - Payload non normative example.

Table3.7 - Payload non normative example

Topic Filter	“a/b”
Topic Filter	“c/d”

Figure 3.30 - Payload byte format non normative example

	Description	7	6	5	4	3	2	1	0
Topic Filter									
byte 1	Length MSB (0)	0	0	0	0	0	0	0	0
byte 2	Length LSB (3)	0	0	0	0	0	0	1	1
byte 3	‘a’ (0x61)	0	1	1	0	0	0	0	1
byte 4	‘/’ (0x2F)	0	0	1	0	1	1	1	1
byte 5	‘b’ (0x62)	0	1	1	0	0	0	1	0
Topic Filter									
byte 6	Length MSB (0)	0	0	0	0	0	0	0	0
byte 7	Length LSB (3)	0	0	0	0	0	0	1	1
byte 8	‘c’ (0x63)	0	1	1	0	0	0	1	1
byte 9	‘/’ (0x2F)	0	0	1	0	1	1	1	1
byte 10	‘d’ (0x64)	0	1	1	0	0	1	0	0

3.10.4 Response

The Topic Filters (whether they contain wildcards or not) supplied in an UNSUBSCRIBE packet MUST be compared character-by-character with the current set of Topic Filters held by the Server for the Client. If any filter matches exactly then its owning Subscription is deleted, otherwise no additional processing occurs [MQTT-3.10.4-1].

If a Server deletes a Subscription:

- It MUST stop adding any new messages for delivery to the Client [MQTT-3.10.4-2].
- It MUST complete the delivery of any QoS 1 or QoS 2 messages which it has started to send to the Client [MQTT-3.10.4-3].
- It MAY continue to deliver any existing messages buffered for delivery to the Client.

The Server MUST respond to an UNSUBSUBCRIBE request by sending an UNSUBACK packet. The UNSUBACK Packet MUST have the same Packet Identifier as the UNSUBSCRIBE Packet [MQTT-3.10.4-4]. Even where no Topic Subscriptions are deleted, the Server MUST respond with an UNSUBACK [MQTT-3.10.4-5].

If a Server receives an UNSUBSCRIBE packet that contains multiple Topic Filters it MUST handle that packet as if it had received a sequence of multiple UNSUBSCRIBE packets, except that it sends just one UNSUBACK response [MQTT-3.10.4-6].

3.11 UNSUBACK – Unsubscribe acknowledgement（退订确认）

The UNSUBACK Packet is sent by the Server to the Client to confirm receipt of an UNSUBSCRIBE Packet.

服务器发送给终端，确认一个 UNSUBSCRIBE 包已经被接收。

3.11.1 Fixed header

Figure 3.31 – UNSUBACK Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (11)				Reserved			
	1	0	1	1	0	0	0	0
byte 2	Remaining Length (2)							
	0	0	0	0	0	0	1	0

Remaining Length field

This is the length of the variable header. For the UNSUBACK Packet this has the value 2.

剩余长度为 2，就是可变消息头的长度。

3.11.2 Variable header（可变消息头）

The variable header contains the Packet Identifier of the UNSUBSCRIBE Packet that is being acknowledged. 可变消息头包含接收确认的 UNSUBSCRIBE 的包 ID

Figure 3.32 – UNSUBACK Packet variable header

Bit	7	6	5	4	3	2	1	0
byte 1	Packet Identifier MSB							
byte 2	Packet Identifier LSB							

3.11.3 Payload

The UNSUBACK Packet has no payload.没有消息体。

3.12 PINGREQ – PING request（PING 请求）

The PINGREQ Packet is sent from a Client to the Server. It can be used to:
PINGREQ 是终端发给服务器，可以用于：

1. Indicate to the Server that the Client is alive in the absence of any other Control Packets being sent from the Client to the Server（在没有其他控制包要发给服务器的时候，告诉服务器，终端还活着，）
2. Request that the Server responds to confirm that it is alive.（请求服务器响应，确认服务器还活着）
3. Exercise the network to indicate that the Network Connection is active.（网络可用指示）

This Packet is used in Keep Alive processing, see Section 3.1.2.9 for more details.
Keep Alive 处理，详见 3.1.2.10

3.12.1 Fixed header（固定头）

Figure 3.33 – PINGREQ Packet fixed header

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type (12)				Reserved			
	1	1	0	0	0	0	0	0
byte 2	Remaining Length (0)							
	0	0	0	0	0	0	0	0

3.12.2 Variable header

The PINGREQ Packet has no variable header.（没有可变头）

3.12.3 Payload

The PINGREQ Packet has no payload.（没有消息体）

3.12.4 Response

The Server MUST send a PINGRESP Packet in response to a PINGREQ Packet [MQTT-3.12.4-1].
服务器必须发送一个 PINGRESP 响应 PINGREQ

3.13 PINGRESP – PING response（PING 响应）

A PINGRESP Packet is sent by the Server to the Client in response to a PINGREQ Packet. It indicates that the Server is alive.
PINGRESP 服务器发送给终端，是 PINGREQ 包的响应，标识服务器还活着。
This Packet is used in Keep Alive processing, see Section 3.1.2.9 for more details.

3.13.1 Fixed header（固定头）

Figure 3.34 – PINGRESP Packet fixed header

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	MQTT Control Packet type (13)				Reserved			
	1	1	0	1	0	0	0	0
byte 2	Remaining Length (0)							
	0	0	0	0	0	0	0	0

3.13.2 Variable header

The PINGRESP Packet has no variable header. （PINGRESP 没有可变消息头）

3.13.3 Payload

The PINGRESP Packet has no payload.（PINGRESP 没有消息体）

3.14 DISCONNECT – Disconnect notification（断开链接通知）

The DISCONNECT Packet is the final Control Packet sent from the Client to the Server. It indicates that the Client is disconnecting cleanly.

DISCONNECT 是终端发送给服务器的控制包。他指示终端正在断开服务器

3.14.1 Fixed header（固定消息头）

Figure 3.35 – DISCONNECT Packet fixed header

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
byte 1	MQTT Control Packet type (14)				Reserved			
	1	1	1	0	0	0	0	0
byte 2	Remaining Length (0)							
	0	0	0	0	0	0	0	0

The Server MUST validate that reserved bits are set to zero and disconnect the Client if they are not zero [MQTT-3.14.1-1].

3.14.2 Variable header

The DISCONNECT Packet has no variable header.没有可变消息头

3.14.3 Payload

The DISCONNECT Packet has no payload.没有消息体

3.14.4 Response（响应）

After sending a DISCONNECT Packet the Client:在发送一个 DISCONNECT 包之后，终端：

- MUST close the Network Connection [MQTT-3.14.4-1].
- 必须关系网络连接
- MUST NOT send any more Control Packets on that Network Connection [MQTT-3.14.4-2].
- 禁止在网络上发送任何控制包。

On receipt of DISCONNECT the Server:接收到 DISCONNECT 之后，服务器：

必须抛弃任何 Will 消息，跟当前连接有关，并且没有发布它，如章节 3.1.2.5
如果终端还没有完成这样干，应该关闭网络连接。

4 Operational behavior 操作性的行为

4.1 Storing state

It is necessary for the Client and Server to store Session state in order to provide Quality of Service guarantees. The Client and Server MUST store Session state for the entire duration of the Session [MQTT-4.1.0-1]. A Session MUST last at least as long it has an active Network Connection [MQTT-4.1.0-2].

为了提供 QoS 保证，终端和服务器有必要存储会话状态。终端与服务器必须为整个会话存储会话状态。

会话必须持续直到网络连接是激活的。

Retained messages do not form part of the Session state in the Server. The Server SHOULD retain such messages until deleted by a Client.

保留信息不能组成服务器的会话状态。服务器应保留这些信息直到被终端删除。

Non normative comment

实例

The storage capabilities of Client and Server implementations will of course have limits in terms of capacity and may be subject to administrative policies such as the maximum time that Session state is stored between Network Connections. Stored Session state can be discarded as a result of an administrator action, including an automated response to defined conditions. This has the effect of terminating the Session. These actions might be prompted by resource constraints or for other operational reasons. It is prudent to evaluate the storage capabilities of the Client and Server to ensure that they are sufficient.

终端和服务器的储存能力在履行时有所限制并服从于诸如网络连接间会话状态的最大时间。存储会话状态。经管理员操作可删除存储会话状态，包括对限制条件的自动回复。该行为会对终止会话产生影响。资源限制或其他操作性原因导致此类行为发生。需要对终端和服务器的存储能力进行谨慎评估以确保存储充足。

Non normative comment

It is possible that hardware or software failures may result in loss or corruption of Session state stored by the Client or Server.

硬件和软件的损坏可能会导致终端和服务器的存储会话状态丢失或损坏。

Non normative comment

Normal operation of the Client of Server could mean that stored state is lost or corrupted because of administrator action, hardware failure or software failure. An administrator action could be an automated response to defined conditions. These actions might be prompted by resource constraints or for other operational reasons. For example the server might determine that based on external knowledge, a message or messages can no longer be delivered to any current or future client.

终端和服务器的一般性行为意味着由于管理性操作或者硬件软件损坏导致存储状态丢失或损坏。管理性操作可作为对限制条件的自动回复。资源限制或者其他操作性原因促使该类操作发生。例如，服务器必须基于外部认识作出决断，信息不再能够传递到任何当前或者之后的终端上。

Non normative comment

An MQTT user should evaluate the storage capabilities of the MQTT Client and Server implementations to ensure that they are sufficient for their needs.

MQTT 用户应对 MQTT 终端和服务器存储能力进行评估，从而确保他们能够充分满足要求。

4.1.1 Non normative example

实例

For example, a user wishing to gather electricity meter readings may decide that they need to use QoS 1 messages because they need to protect the readings against loss over the network, however they may have determined that the power supply is sufficiently reliable that the data in the Client and Server can be stored in volatile memory without too much risk of its loss.

比方说，有个用户希望收集电表读数，于是决定利用 QoS 信息，因为他们需要保护读数以免在网上丢失。然而，他们需要确保有充足的动力储存好终端和服务器上的数据，以防丢失。

Conversely a parking meter payment application provider might decide that there are no circumstances where a payment message can be lost so they require that all data are force written to non-volatile memory before it is transmitted across the network.

相反，一个停车收费 app 的开发者要保证在任何情况下都不能丢失收付信息。于是他们要求在传送到网上之前，强制录入所有数据。

4.2 Network Connections

The MQTT protocol requires an underlying transport that provides an ordered, lossless, stream of bytes from the Client to Server and Server to Client.

MQTT 协议需要一个能够完成终端和服务器之间的传输的基本传输方式。

Non normative comment

The transport protocol used to carry MQTT 3.1 was TCP/IP as defined in [RFC793]. TCP/IP can be used for MQTT 3.1.1. The following are also suitable:

- TLS [RFC5246]
- WebSocket [RFC6455]

在 RFC793 中规定，被用作传送 MQTT3.1 的传输协议是 TCP/IP。除了 Tcp/IP 可以传输 MQTT3.1.1.还有这些

- TLS [RFC5246]
- WebSocket [RFC6455]

Non normative comment

TCP ports 8883 and 1883 are registered with IANA for MQTT TLS and non TLS communication respectively.

TCP ports 8883 和 1883 分别注册为 IANA MQTT TLS 和非 TLS

Connectionless network transports such as User Datagram Protocol (UDP) are not suitable on their own because they might lose or reorder data.

断网传送方式比如 UDP 对他们就不适合，因为这种方法会丢失或弄乱数据。

4.3 Quality of Service levels and protocol flows

MQTT delivers Application Messages according to the Quality of Service (QoS) levels defined here. The delivery protocol is symmetric, in the description below the Client and Server can each take the role of either Sender or Receiver. The delivery protocol is concerned solely with the delivery of an application message from a single Sender to a single Receiver. When the Server is delivering an Application Message to more than one Client, each Client is treated independently. The QoS level used to deliver an Application Message outbound to the Client could differ from that of the inbound Application Message.

The non-normative flow diagrams in the following sections are intended to show possible implementation approaches.

MQTT 基于 QoS 层次定义传送应用信息。其传输协议是对称的，在描述中，终端和服务器都可作为发送者和接受者。发送者和接受者传递一个应用信息只和传输协议有关。当一个服务器在传递一个应用信息给多个终端时，每个终端都是作为独立的。以前，QoS 传递一个向外的信息给终端与传递一个向内信息是不一样的。

在之后环节中的非规范流程图是为了指出可能的实施办法。

4.3.1 QoS 0: At most once delivery

The message is delivered according to the capabilities of the underlying network. No response is sent by the receiver and no retry is performed by the sender. The message arrives at the receiver either once or not at all.

基于基础网络能力的信息传送。接受者不发任何回应，发送者也不进行重试。信息一次性传送到接受者要不就传不到。

In the QoS 0 delivery protocol, the Sender

在 QoS 传输协议中，发送者

- MUST send a PUBLISH packet with QoS=0, DUP=0 [MQTT-4.3.1-1].

必须发送 PUBLISH packet with QoS=0, DUP=0 [MQTT-4.3.1-1]

In the QoS 0 delivery protocol, the Receiver

在 QoS 传输协议中，接受者

- Accepts ownership of the message when it receives the PUBLISH packet.

收到 PUBLISH packet 后就获得信息的所有权。

Figure 4.1 – QoS 0 protocol flow diagram, non normative example

Sender Action	Control Packet	Receiver Action
PUBLISH QoS 0, DUP=0		
	---------->	

		Deliver Application Message to appropriate onward recipient(s) 传递应用信息给前方适当的接受者

4.3.2 QoS 1: At least once delivery 至少一次传递

This quality of service ensures that the message arrives at the receiver at least once. A QoS 1 PUBLISH Packet has a Packet Identifier in its variable header and is acknowledged by a PUBACK Packet. Section 2.3.1 provides more information about Packet Identifiers.

服务质量需保证一次性将信息传送给接收者。QoS PUBLISH packet 能识别和认知 packet。2.3.1 部分具体描述关于 packet 识别。

In the QoS 1 delivery protocol, the Sender

在 QoS 传输协议中，发送者

- MUST assign an unused Packet Identifier each time it has a new Application Message to publish.
- 每推送新的应用信息时必须分配一个未使用的 packet 识别器
- MUST send a PUBLISH Packet containing this Packet Identifier with QoS=1, DUP=0.
- 必须发送包含有 packet 识别器（QoS=1, DUP=0）的 publish packet
- MUST treat the PUBLISH Packet as “unacknowledged” until it has received the corresponding PUBACK packet from the receiver. See Section 4.4 for a discussion of unacknowledged messages.
- 必须将 PUBLISH 包作为未知直到其从发送者处收到相关的 PUBACK 包。4.4 将对未知信息进行讨论

[MQTT-4.3.2-1].

The Packet Identifier becomes available for reuse once the Sender has received the PUBACK Packet.

一旦发送者收到 PUBACK 包时，PACKET 识别能够再次使用。

Note that a Sender is permitted to send further PUBLISH Packets with different Packet Identifiers while it is waiting to receive acknowledgements.

注意，发送者在等候接受确认时，有资格发送后续的带有不同包 ID 的 PUBLISH 包。

In the QoS 1 delivery protocol, the Receiver

在 QoS 传输协议中，接受者

- MUST respond with a PUBACK Packet containing the Packet Identifier from the incoming PUBLISH Packet, having accepted ownership of the Application Message
- 必须对包含从传入的 publish 包（取得应用信息所有权）里来的包 id 的 PUBACK 包进行回应，
- After it has sent a PUBACK Packet the Receiver MUST treat any incoming PUBLISH packet that contains the same Packet Identifier as being a new publication, irrespective of the setting of its DUP flag.
- 在发送 PUBACK 包之后，不管其 DUP 标志的设置，接受者必须将传送中包含有相同包 ID 的 PUBLISH 包作为新发布者。

[MQTT-4.3.2-2].

Figure 4.2 – QoS 1 protocol flow diagram, non normative example

Sender Action	Control Packet	Receiver action
Store message		
Send PUBLISH QoS 1, DUP 0, <Packet Identifier>	---------->	
		Initiate onward delivery of the Application Message[1]
	<----------	Send PUBACK <Packet Identifier>
Discard message		

[1] The receiver is not required to complete delivery of the Application Message before sending the PUBACK. When its original sender receives the PUBACK packet, ownership of the Application Message is transferred to the receiver.

在发送 PUBACK 之前，接收者无需完善应用信息的传递。当初始发送者收到 PUBACK 包时，也将应用信息所有权传给了接收者。

4.3.3 QoS 2: Exactly once delivery 至少一次完全正确的传递

This is the highest quality of service, for use when neither loss nor duplication of messages are acceptable. There is an increased overhead associated with this quality of service.

在使用时能够既不丢失信息也不复制信息，这是最高端的服务。这种高质量服务会不断增加开销。

A QoS 2 message has a Packet Identifier in its variable header. Section 2.3.1 provides more information about Packet Identifiers. The receiver of a QoS 2 PUBLISH Packet acknowledges receipt with a two-step acknowledgement process.

QoS 2 信息有个包 id。2.3.1 部分对包 id 进行更多描述。Qos2 PUBLISH 包的接受者承认收到两步式认证过程。

In the QoS 2 delivery protocol, the Sender

在 QoS2 传送协议中，发送者

- MUST assign an unused Packet Identifier when it has a new Application Message to publish.
- 当其有一个新的应用信息要发布时，必须分配一个未使用的包 ID.
- MUST send a PUBLISH packet containing this Packet Identifier with QoS=2, DUP=0.
- 必须发送一个包含有包 ID（带有 QoS=2, DUP=0)的 PUBLISH 包
- MUST treat the PUBLISH packet as “unacknowledged” until it has received the corresponding PUBREC packet from the receiver. See Section 4.4 for a discussion of unacknowledged messages.
- 必须将 PUBLISH 包作为未认证的，直到从接收者出接收到相关 PUBERC 包。4.4 部分将对未认证进行讨论。

- MUST send a PUBREL packet when it receives a PUBREC packet from the receiver. This PUBREL packet MUST contain the same Packet Identifier as the original PUBLISH packet.
- 当从接收者处收到 PUBERC 包时，必须发布 PUBREL 包，其中必须包含有与初始 PUBLSH 包相同的包 ID.
- MUST treat the PUBREL packet as “unacknowledged” until it has received the corresponding PUBCOMP packet from the receiver.
- 必须将 PUBREL 包作为未认证的直到其从接收者处收到相关 PUBCOMP 包。
- MUST NOT re-send the PUBLISH once it has sent the corresponding PUBREL packet.
- 一旦发了相关的 PUBREL 包就禁止重新发送 PUBLSH.

[MQTT-4.3.3-1].

The Packet Identifier becomes available for reuse once the Sender has received the PUBCOMP Packet.

一旦发送者收到PUBCOMP包，包ID就能够再次使用。

Note that a Sender is permitted to send further PUBLISH Packets with different Packet Identifiers while it is waiting to receive acknowledgements.

注意，发送者在等候接受确认时，有资格发送后续的带有不同包 ID 的 PUBLISH 包。

In the QoS 2 delivery protocol, the Receiver

在 QoS2 传送协议中，接收者

- MUST respond with a PUBREC containing the Packet Identifier from the incoming PUBLISH Packet, having accepted ownership of the Application Message.
- 必须对包含从传入的 publish 包（取得应用信息所有权）里来的包 id 的 PUBREC 包进行回应，
-
- Until it has received the corresponding PUBREL packet, the Receiver MUST acknowledge any subsequent PUBLISH packet with the same Packet Identifier by sending a PUBREC. It MUST NOT cause duplicate messages to be delivered to any onward recipients in this case.
- 接收者必须通过发送 PUBREC 来认证之后有着相同包 ID 的 PUBLISH 包，直到其收到相关 PUBREL 包。禁止造成复制信息传递到任何前方的接收者处。
- MUST respond to a PUBREL packet by sending a PUBCOMP packet containing the same Packet Identifier as the PUBREL.
- 必须通过发送包含与 PUBREL 相同 的包 ID 的 PUBCOMP 包来对 PUBREL 包进行回应。
- After it has sent a PUBCOMP, the receiver MUST treat any subsequent PUBLISH packet that contains that Packet Identifier as being a new publication.

[MQTT-4.3.3-2].

发送 PUBCOMP 之后，接收者必须将之后那些包含包 ID 的 PUBLISH 包作为新的发布者。

Figure 4.3 – QoS 2 protocol flow diagram, non normative example

Sender Action	Control Packet	Receiver Action
Store message		
PUBLISH QoS 2, DUP 0 <Packet Identifier>		

	---------->	
		Method A, Store message or Method B, Store <Packet Identifier> then Initiate onward delivery of the Application Message[1]
		PUBREC <Packet Identifier>
	<----------	
Discard message, Store PUBREC received <Packet Identifier>		
PUBREL <Packet Identifier>		
	---------->	
		Method A, Initiate onward delivery of the Application Message[1] then discard message or Method B, Discard <Packet Identifier>
		Send PUBCOMP <Packet Identifier>
	<----------	
Discard stored state		

[1] The receiver is not required to complete delivery of the Application Message before sending the PUBREC or PUBCOMP. When its original sender receives the PUBREC packet, ownership of the Application Message is transferred to the receiver.

在发送 PUBREC 或者 PUBCOMP 前，接收者无需完善应用信息的传送。当初始发送者收到 PUBREC 包时，应用信息的所有权便转给了接收者。

Figure 4.3 shows that there are two methods by which QoS 2 can be handled by the receiver. They differ in the point within the flow at which the message is made available for onward delivery. The choice of Method A or Method B is implementation specific. As long as an implementation chooses exactly one of these approaches, this does not affect the guarantees of a QoS 2 flow.

图表 4.3 指出接收者可通过两种方法来解决 QoS2.他们的不同在于如何使得信息能够向前传送。两种方法的选择在应用时是明确的。只要选择其中一种方法，便不会影响 QoS2 流动的保证。

4.4 Message delivery retry 信息传递重试

When a Client reconnects with CleanSession set to 0, both the Client and Server MUST re-send any unacknowledged PUBLISH Packets (where QoS > 0) and PUBREL Packets using their original Packet Identifiers [MQTT-4.4.0-1]. This is the only circumstance where a Client or Server is REQUIRED to redeliver messages.

当终端与会话（设置为 0）重新连接时，终端与服务器都必须重新发送未认证的 PUBLISH 包（QoS>0）PUBREL 包使用其初始包 ID [MQTT-4.4.0-1]. 这是终端或者服务器有资格重新传递信息的唯一情况。

Non normative comment

Historically retransmission of Control Packets was required to overcome data loss on some older TCP networks. This might remain a concern where MQTT 3.1.1 implementations are to be deployed in such environments.

以前，在老的 TCP 网络协议中，重新传送控制包需要克服数据丢失。其中可包含 MQTT3.1.1 在一些环境中应用的协议。

4.5 Message receipt

When a Server takes ownership of an incoming Application Message it MUST add it to the Session state of those clients that have matching Subscriptions. Matching rules are defined in Section 4.7 [MQTT-4.5.0-1].

当服务器取得应用信息的所有权时，必须将其加入符合协议的终端的会话状态中去。4.7 部分详细说明匹配原则。

Under normal circumstances Clients receive messages in response to Subscriptions they have created. A Client could also receive messages that do not match any of its explicit Subscriptions. This can happen if the Server automatically assigned a subscription to the Client. A Client could also receive messages while an UNSUBSCRIBE operation is in progress. The Client MUST acknowledge any Publish Packet it receives according to the applicable QoS rules regardless of whether it elects to process the Application Message that it contains [MQTT-4.5.0-2].

正常状况下，终端接收信息作为对他们所建立的协议的回应。终端也可接受那些不符合其明确协议的信息。如果服务器给终端自动分配了协议，这种情况就能够发生。当过程中发生取消操作时，终端同样也能接受信息。根据 QoS 规则，终端必须承认任何其接收的 PUBLISH 包，无论他是否处理其中包含的应用信息。

4.6 Message ordering

A Client MUST follow these rules when implementing the protocol flows defined elsewhere in this chapter:

这章节对实施协议进行定义，终端必须遵守这些条例

- When it re-sends any PUBLISH packets, it MUST re-send them in the order in which the original PUBLISH packets were sent (this applies to QoS 1 and QoS 2 messages) [MQTT-4.6.0-1]
- 当重新发送任何 PUBLISH 包时，初始 PUBLISH 包已发送时，必须按序重新发送。（这应用于 QoS1 和 QoS2 信息）
- It MUST send PUBACK packets in the order in which the corresponding PUBLISH packets were received (QoS 1 messages) [MQTT-4.6.0-2]
- 收到相关 PUBLISH 包时，必须顺序发送 puback 包（QoS 1 信息）
- It MUST send PUBREC packets in the order in which the corresponding PUBLISH packets were received (QoS 2 messages) [MQTT-4.6.0-3]
- 收到相关 PUBLISH 包时，必须顺序发送 pubrec 包（QoS 2 信息）
-
- It MUST send PUBREL packets in the order in which the corresponding PUBREC packets were received (QoS 2 messages) [MQTT-4.6.0-4]
- 收到相关 PUBLISH 包时，必须顺序发送 puback 包（QoS 2 信息）

A Server MUST by default treat each Topic as an "Ordered Topic". It MAY provide an administrative or other mechanism to allow one or more Topics to be treated as an "Unordered Topic" [MQTT-4.6.0-5].

When a Server processes a message that has been published to an Ordered Topic, it MUST follow the rules listed above when delivering messages to each of its subscribers. In addition it MUST send PUBLISH packets to consumers (for the same Topic and QoS) in the order that they were received from any given Client [MQTT-4.6.0-6].

服务器必须默认每个主题是有序的。可能提供管理或其他机制来允许一个或多个主题作为无序主题

当服务器对发布给一个有序主题的信息进行加工时，必须遵从传递信息给订阅者所列出的各种规则。此外，必须将他们从终端收到 PUBLISH 包顺序发给消费者（同一主题和 QoS)

Non normative comment

The rules listed above ensure that when a stream of messages is published and subscribed to with QoS 1, the final copy of each message received by the subscribers will be in the order that they were originally published in, but the possibility of message duplication could result in a re-send of an earlier message being received after one of its successor messages. For example a publisher might send messages in the order 1,2,3,4 and the subscriber might receive them in the order 1,2,3,2,3,4.

以上所列的规则为确保当发布和订阅 QoS1 一系列信息时 ，最终复制的每条初始信息将有序发送给订阅者，但是复制的信息可能导致重新发送先前收到的一条信息。比如发布者以 1234 的顺序发布，订阅者可能收到的信息顺序为 123234

If both Client and Server make sure that no more than one message is “in-flight” at any one time (by not sending a message until its predecessor has been acknowledged), then no QoS 1 message will be received after any later one - for example a subscriber might receive them in the order 1,2,3,3,4 but not 1,2,3,2,3,4. Setting an in-flight window of 1 also means that order will be preserved even if the publisher sends a sequence of messages with different QoS levels on the same topic.

如果终端和服务器确保某个时刻不超过一条信息在发送中（直到先前的被认证前不发送任何信息），则之后也不会收到 QoS1 信息。比如，一个订阅者收到的顺序为 12334 不是 123234.。设立 1 的发送窗口意味着该顺序将被保存，即使订阅者发送同一主题不同 QoS 的信息序列。

4.7 Topic Names and Topic Filters

主体名称和主题过滤

4.7.1 Topic wildcards 主题通配符

The topic level separator is used to introduce structure into the Topic Name. If present, it divides the Topic Name into multiple “topic levels”.

主体分隔符被用于介绍主题名称的结构。如今，它将主体名称分成多元化主题层次

A subscription’s Topic Filter can contain special wildcard characters, which allow you to subscribe to multiple topics at once.

一个协议的主题过滤包含特殊通配符，可允许你一次订阅多元主题。

The wildcard characters can be used in Topic Filters, but MUST NOT be used within a Topic Name [MQTT-4.7.1-1].

通配符可应用于主题过滤，不能用于主题名称。

4.7.1.1 Topic level separator

The forward slash ('/' U+002F) is used to separate each level within a topic tree and provide a hierarchical structure to the Topic Names. The use of the topic level separator is significant when either of the two wildcard characters is encountered in Topic Filters specified by subscribing Clients. Topic level separators can appear anywhere in a Topic Filter or Topic Name. Adjacent Topic level separators indicate a zero length topic level.

前面的斜线（/ U+002F)用于分离每个层次，通过主题树的形式，并为主题名称建立等级制度。当在主题过滤中遇上订阅终端指定的两个通配符时，主题分隔符的使用具有重要意义。主题分隔符在主题过滤和主题名称中无处不在。临近的主题分隔符表明零长度主题级别。

4.7.1.2 Multi-level wildcard 多级通配符

The number sign ('#' U+0023) is a wildcard character that matches any number of levels within a topic. The multi-level wildcard represents the parent and any number of child levels. The multi-level wildcard character MUST be specified either on its own or following a topic level separator. In either case it MUST be the last character specified in the Topic Filter [MQTT-4.7.1-2].

数字标志（'#' U+0023）为一个能够与主题任何级别的数字匹配的通配符。多级通配符代表母体和各种子体级别数字。必须自主或者遵从主体分隔符指定多级通配符。无论发生何种情况，在主题过滤中必须最后对其进行规定。

Non normative comment

For example, if a Client subscribes to "sport/tennis/player1/#", it would receive messages published using these topic names:

比如，如果一个终端这样描述"sport/tennis/player1/#"，它将接收到使用以下名称的信息发布。

- "sport/tennis/player1"
- "sport/tennis/player1/ranking"
- "sport/tennis/player1/score/wimbledon"

Non normative comment

- "sport/#" also matches the singular "sport", since # includes the parent level.
- "sport/#" 同样适合 "sport",因为#包括母体级别
- "#" is valid and will receive every Application Message
- "#" 正确 可以收到任何应用消息
- "sport/tennis/#" is valid
- 合规
- "sport/tennis#" is not valid
- 不合规
- "sport/tennis/#/ranking" is not valid
- 不合规

4.7.1.3 Single level wildcard 单独等级通配符

The plus sign ('+' U+002B) is a wildcard character that matches only one topic level.

加号也是一个通配符，其只适用于一个主题等级

The single-level wildcard can be used at any level in the Topic Filter, including first and last levels. Where it is used it MUST occupy an entire level of the filter [MQTT-4.7.1-3]. It can be used at more than one level in the Topic Filter and can be used in conjunction with the multilevel wildcard.

单独等级通配符适用于主题过滤的各个等级，包括第一级和最后一级。使用时必须 占据过滤器的整个等级 [MQTT-4.7.1-3]。在主题过滤器中其可适用于多个级别以及与多元等级通配符的连接中。

Non normative comment

For example, “sport/tennis/+” matches “sport/tennis/player1” and “sport/tennis/player2”, but not “sport/tennis/player1/ranking”. Also, because the single-level wildcard matches only a single level, “sport/+” does not match “sport” but it does match “sport/”.

比如，“sport/tennis/+” 适用 “sport/tennis/player1” 和“sport/tennis/player2”，但非 “sport/tennis/player1/ranking”。而且，因为单独等级通配符只适合唯一的等级。“sport/+” 不适 “sport” 但却适用于 “sport/”.

Non normative comment

- “+” is valid
- 合规
- “+/tennis/#” is valid
- 合规
- “sport+” is not valid
- 不合规
- “sport/+/player1” is valid
- 合规
- “/finance” matches “+/+” and “/+”, but not “+”

4.7.2 Topics beginning with $ 主题开头的$

The Server MUST NOT match Topic Filters starting with a wildcard character (# or +) with Topic Names beginning with a $ character [MQTT-4.7.2-1]. The Server SHOULD prevent Clients from using such Topic Names to exchange messages with other Clients. Server implementations MAY use Topic Names that start with a leading $ character for other purposes.

服务器不能与主题开头有$的通配符的主题过滤器相配。服务器应防止终端使用此类主题名称来与其他终端交换消息。为了其他目的，服务器运行会使用主题名称开头为$。

Non normative comment

- $SYS/ has been widely adopted as a prefix to topics that contain Server-specific information or control APIs
- $SYS/ 被广泛认作为主题前缀，包含服务器专用信息或控制 APIs
- Applications cannot use a topic with a leading $ character for their own purposes

  应用不能为其自身目的而使用开头$的主题。

Non normative comment

- A subscription to “#” will not receive any messages published to a topic beginning with a $
- #的订阅将不会接收到任何发布给开头为$的主题的信息
- A subscription to “+/monitor/Clients” will not receive any messages published to “$SYS/monitor/Clients”
- “+/monitor/Clients”的订阅将不会接收到发布给“$SYS/monitor/Clients”的信息
- A subscription to “$SYS/#” will receive messages published to topics beginning with “$SYS/”
- “$SYS/#” 的订阅将接收到发布给主题开头为“$SYS/”的信息
- A subscription to “$SYS/monitor/+” will receive messages published to “$SYS/monitor/Clients”
- “$SYS/monitor/+” 的订阅将收到发布给“$SYS/monitor/Clients”的信息
- For a Client to receive messages from topics that begin with $SYS/ and from topics that don’t begin with a $, it has to subscribe to both “#” and “$SYS/#”
- 对于终端从开头为$SYS/的主题和不以$开头的主题处收到的信息，必须标注“#” 和 “$SYS/#”

4.7.3 Topic semantic and usage 主题语意和用法

The following rules apply to Topic Names and Topic Filters:

以下规则适用于主题名称和主题过滤

- All Topic Names and Topic Filters MUST be at least one character long [MQTT-4.7.3-1]
- 所有主题名称和主题过滤必须至少一个字符长
- Topic Names and Topic Filters are case sensitive
- 主题名称和主题过滤区分大小写
- Topic Names and Topic Filters can include the space character
- 主题名称和主题过滤可包括空格字符
- A leading or trailing ‘/’ creates a distinct Topic Name or Topic Filter
- 开头或结尾 ‘/’ 创建了不同的主题名称或主题过滤
- A Topic Name or Topic Filter consisting only of the ‘/’ character is valid
- 只有包含‘/’ 字符的主题名称或主题过滤器是有效的
- Topic Names and Topic Filters MUST NOT include the null character (Unicode U+0000) [Unicode] [MQTT-4.7.3-2]
- 主题名称和主题过滤器不能包括空字符 (Unicode U+0000)
- Topic Names and Topic Filters are UTF-8 encoded strings, they MUST NOT encode to more than 65535 bytes [MQTT-4.7.3-3]. See Section 1.5.3
- 主题名称和主题过滤为 UTF-8 编码字符串，不能超过 65535 bytes [MQTT-4.7.3-3] 详见 1.5.3

There is no limit to the number of levels in a Topic Name or Topic Filter, other than that imposed by the overall length of a UTF-8 encoded string.

主题名称和主题过滤的级别数没有限制，除此之外，为 UTF-8 编码字符串所加。

When it performs subscription matching the Server MUST NOT perform any normalization of Topic Names or Topic Filters, or any modification or substitution of unrecognized characters [MQTT-4.7.3-4]. Each non-wildcarded level in the Topic Filter has to match the corresponding level in the Topic Name character for character for the match to succeed.

演示时，服务器协议不能演示任何主题名称或主题过滤的正常形式，或未识别字符的任何替代体或仿体。

每个主题过滤中的非通配符级别需与主题名称中的相应级别匹配。

Non normative comment

The UTF-8 encoding rules mean that the comparison of Topic Filter and Topic Name could be performed either by comparing the encoded UTF-8 bytes, or by comparing decoded Unicode characters

UTF-8 编码规则意味着通过比对 UTF-8 bytes 或比对解码的单一码字符可显示出主题过滤与主题名称的不同。

Non normative comment

- “ACCOUNTS” and “Accounts” are two different topic names
- “ACCOUNTS” 与 “Accounts”为两种不同主题名称
- “Accounts payable” is a valid topic name
- Accounts payable 有效
- “/finance” is different from “finance”
- “/finance” 有别于“finance”
-

An Application Message is sent to each Client Subscription whose Topic Filter matches the Topic Name attached to an Application Message. The topic resource MAY be either predefined in the Server by an administrator or it MAY be dynamically created by the Server when it receives the first subscription or an Application Message with that Topic Name. The Server MAY also use a security component to selectively authorize actions on the topic resource for a given Client.

应用信息发送给每个主题过滤器与主题名称相符的订阅终端上。主题资源可能会被服务器的操作者预定义，若当他收到第一个订阅或主题名称的应用信息时，主题资源也可被服务器随机创建。服务器也会利用安全控件选择性授权给终端的主题资源。

4.8 Handling errors 错误处理

Unless stated otherwise, if either the Server or Client encounters a protocol violation, it MUST close the Network Connection on which it received that Control Packet which caused the protocol violation [MQTT-4.8.0-1].

除非特别声明，若服务器或终端碰上违反协议，必须关掉那个收到导致违反协议的控制包的网络连接。

A Client or Server implementation might encounter a Transient Error (for example an internal buffer full condition) that prevents successful processing of an MQTT packet.

服务器或终端操作时可能碰上暂时性错误（比如内部缓存已满）而阻碍 MQTT 包的进程。

If the Client or Server encounters a Transient Error while processing an inbound Control Packet it MUST close the Network Connection on which it received that Control Packet [MQTT-4.8.0-2]. If a Server detects a Transient Error it SHOULD NOT disconnect or have any other effect on its interactions with any other Client.

若终端或服务器在运行内部控制包时碰上暂时性错误时，必须关闭那个收到控制包的网络连接。若服务器检测到一个暂时性错误，不能断开或者做出影响与其他终端连接的行为。

5 Security 安全

5.1 Introduction

This Chapter is provided for guidance only and is **Non Normative**. However, it is strongly recommended that Server implementations that offer TLS [RFC5246] SHOULD use TCP port 8883 (IANA service name: secure-mqtt).

本章只提供非规范性指导意见，但是，强烈建议 TLS [RFC5246]服务器操作应利用 TCP port 8883 (IANA service name: secure-mqtt。

There are a number of threats that solution providers should consider. For example:

应考虑以下一系列威胁的解决方案

- Devices could be compromised
- 设备可能受破坏
- Data at rest in Clients and Servers might be accessible
- 终端上的空闲数据和服务器可能会被访问
- Protocol behaviors could have side effects (e.g. “timing attacks”)
- 协议行为可能产生副作用（如 “timing attacks”)
- Denial of Service (DoS) attacks
- DoS 攻击
- Communications could be intercepted, altered, re-routed or disclosed
- 通讯可能被截获 修改 改道 或关闭
- Injection of spoofed Control Packets

- 欺诈控制包乱入

MQTT solutions are often deployed in hostile communication environments. In such cases, implementations will often need to provide mechanisms for:

在不良网络连接环境中，经常部署 MQTT 解决方案。一些情况下，操作也要提供机制。

- Authentication of users and devices
- 用户和设备的身份验证
- Authorization of access to Server resources
- 访问服务器的授权
- Integrity of MQTT Control Packets and application data contained therein
- MQTT 控制包和包含在其中的应用数据的健全
- Privacy of MQTT Control Packets and application data contained therein
- MQTT 控制包和包含其中的应用数据的隐私

As a transport protocol, MQTT is concerned only with message transmission and it is the implementer’s responsibility to provide appropriate security features. This is commonly achieved by using TLS [RFC5246].

作为运输协议，MQTT 只管信息运输，操作者有责任提供合适安全措施。一般通过使用 TLS 实现。

In addition to technical security issues there could also be geographic (e.g. U.S.-EU SafeHarbor [USEUSAFEHARB]), industry specific (e.g. PCI DSS [PCIDSS]) and regulatory considerations (e.g. Sarbanes-Oxley [SARBANES]).

技术安全问题上还会存在地理性问题（比如 e.g. U.S.-EU SafeHarbor [USEUSAFEHARB]),特定行业 (e.g. PCI DSS [PCIDSS]) 监管问题(e.g. Sarbanes-Oxley [SARBANES]).

5.2 MQTT solutions: security and certification MQTT 解决方案：安全和认证

An implementation might want to provide conformance with specific industry security standards such as NIST Cyber Security Framework [NISTCSF], PCI-DSS [PCIDSS]), FIPS-140-2 [FIPS1402] and NSA Suite B [NSAB].

一个操作想要遵从行业安全标准（比如 NIST Cyber Security Framework [NISTCSF], PCI-DSS [PCIDSS]),FIPS-140-2 [FIPS1402] and NSA Suite B [NSAB].

Guidance on using MQTT within the NIST Cyber Security Framework [NISTCSF] can be found in the MQTT supplemental publication, MQTT and the NIST Framework for Improving Critical Infrastructure Cybersecurity [MQTT NIST]. The use of industry proven, independently verified and certified technologies will help meet compliance requirements.

在 MQTT 再版中可找到 NIST 网络安全框架中 MQTT 的使用指导，还有为改进网络安全设施 MQTT 和 NIST 框架。行业证明的使用，独立的技术认证能够帮助达到合规要求。

5.3 Lightweight cryptography and constrained devices 轻量加密和设备受限

Advanced Encryption Standard [AES] and Data Encryption Standard [DES] are widely adopted.

广泛接受的标准是 AES 和 DES

ISO 29192 [ISO29192] makes recommendations for cryptographic primitives specifically tuned to perform on constrained “low end” devices.

为运行受限的低端设备而调整密码算法时，建议使用 ISO 29192 [ISO29192]行业标准

5.4 Implementation notes 操作注意

There are many security concerns to consider when implementing or using MQTT. The following section should not be considered a “check list”.

操作或使用 MQTT 时需考虑很多安全问题，以下部分不能作为“检查单”

An implementation might want to achieve some, or all, of the following:
一项操作要达到以下其中一些甚至全部

5.4.1 Authentication of Clients by the Server 服务器认证终端

The CONNECT Packet contains Username and Password fields. Implementations can choose how to make use of the content of these fields. They may provide their own authentication mechanism, use an external authentication system such as LDAP [RFC4511] or OAuth [RFC6749] tokens, or leverage operating system authentication mechanisms.
连接包里包含了用户名和密码的内容。操作时可选择如何利用这些字段的内容。他们可能拥有自己的授权机制，有诸如 LDAP 和 OAuth 这样的外部授权系统，或者利用操作系统的授权机制。

Implementations passing authentication data in clear text, obfuscating such data elements or requiring no authentication data should be aware this can give rise to Man-in-the-Middle and replay attacks. Section 5.4.5 introduces approaches to ensure data privacy.
操作以明文形式通过授权，模糊处理数据或是不需要认证数据应当谨慎这种行为会产生中间人和重复攻击。5.4.5 部分介绍如何保证数据的私密性

A Virtual Private Network (VPN) between the Clients and Servers can provide confidence that data is only being received from authorized Clients.
服务器与终端间的 VPN 能有效使得数据只能从被授权的终端处取得

Where TLS [RFC5246] is used, SSL Certificates sent from the Client can be used by the Server to authenticate the Client.
当使用 TLS 时，终端将发送 SSL 证书，服务器利用该证书来授权终端。

An implementation might allow for authentication where the credentials are sent in an Application Message from the Client to the Server.
当终端给服务器发送带证书的应用信息时，可给予操作授权。

5.4.2 Authorization of Clients by the Server 服务器授权终端

An implementation may restrict access to Server resources based on information provided by the Client such as User Name, Client Identifier, the hostname/IP address of the Client, or the outcome of authentication mechanisms.
由于授权机制，操作限制进入基于终端所提供的信息（如用户名 终端 ID 主机名和 IP 地址 ）的服务器。

5.4.3 Authentication of the Server by the Client 终端授权服务器

The MQTT protocol is not trust symmetrical: it provides no mechanism for the Client to authenticate the Server.
MQTT 协议不是绝对对称的，它没有建立终端授权服务器的机制

Where TLS [RFC5246] is used, SSL Certificates sent from the Server can be used by the Client to authenticate the Server. Implementations providing MQTT service for multiple hostnames from a single IP address should be aware of the Server Name Indication extension to TLS defined in section 3 of RFC 6066 [RFC6066].This allows a Client to tell the Server the hostname of the Server it is trying to connect to.

当使用 TLS 时，服务器发送 SSL 证书给终端以此来授权服务器。同一 ip 多个主机使用的 MQTT 协议应注意第三部分中 RFC 所定义的服务器名称指示拓展 6066[RFC6066].它允许终端将服务器上试图连接的主机名称告知服务器。

An implementation might allow for authentication where the credentials are sent in an Application Message from the Server to the Client.

当服务器将带有证书的应用信息发给终端时，可给予操作授权。

A VPN between Clients and Servers can provide confidence that Clients are connecting to the intended Server.

服务器与终端间的 VPN 能有效使得终端与服务器相连。

5.4.4 Integrity of Application Messages and Control Packets 应用信息完整性和控制包

Applications can independently include hash values in their Application Messages. This can provide integrity of the contents of Publish Control Packets across the network and at rest.

应用能独立包括存在于应用信息中的 hash 值。这保证了发布控制包的完整性。

TLS [RFC5246] provides hash algorithms to verify the integrity of data sent over the network.

TLS [RFC5246]利用 hash 算法来证明利用网络发送的数据是完整的

The use of VPNs to connect Clients and Servers can provide integrity of data across the section of the network covered by a VPN.

连接终端和服务器的 VPN 保证了 VPN 覆盖的网络发送的数据的完整性。

5.4.5 Privacy of Application Messages and Control Packets

应用信息的私密性和控制包

TLS [RFC5246] can provide encryption of data sent over the network. There are valid TLS cipher suites that include a NULL encryption algorithm that does not encrypt data. To ensure privacy Clients and Servers should avoid these cipher suites.

TLS [RFC5246] 提供网络发送的数据加密。存在有效的 TLS 加密套件包括一个空的不能加密数据的加密算法。为保证客户端和服务器的隐私，应避免这些加密套件。

An application might independently encrypt the contents of its Application Messages. This could provide privacy of the Application Message both over the network and at rest. This would not provide privacy for other properties of the Application Message such as Topic Name.

一个应用能独立对应用信息进行加密。无论连接与否都能保护应用信息的私密性。但不能保护诸如主题名称的应用信息中的其他内容。

Client and Server implementations can provide encrypted storage for data at rest such as Application Messages stored as part of a Session.

终端和服务器操作能在脱机状态下建立加密储存，比如将应用信息作为一部分会话储存。

The use of VPNs to connect Clients and Servers can provide privacy of data across the section of the network covered by a VPN.

用于连接终端和服务器的 VPN 能保护其下的数据隐私。

5.4.6 Non-repudiation of message transmission 信息传输的不可抵赖性

Application designers might need to consider appropriate strategies to achieve end to end non-repudiation.

App 开发者需要想出合适的策略来实现端到端的不可抵赖性

5.4.7 Detecting compromise of Clients and Servers 客户端与服务器的检测协议

Client and Server implementations using TLS [RFC5246] should provide capabilities to ensure that any SSL certificates provided when initiating a TLS [RFC5246] connection are associated with the hostname of the Client connecting or Server being connected to.

当 与连接中的终端的主机或服务器进行 TLS 连接时，利用 TLS [RFC5246]**操作的客户端和服务器应想方设法确保建立 SSL 证书。**

Client and Server implementations using TLS [RFC5246] can choose to provide capabilities to check Certificate Revocation Lists (CRLs [RFC5280]) and Online Certificate Status Protocol (OSCP) [RFC6960] to prevent revoked certificates from being used.

利用 TLS 的终端和服务器能够选择检查 CRLs 和 OSCP 以此避免使用撤销的证书。

Physical deployments might combine tamper-proof hardware with the transmission of specific data in Application Messages. For example a meter might have an embedded GPS to ensure it is not used in an unauthorized location. [IEEE 802.1AR] is a standard for implementing mechanisms to authenticate a device's identity using a cryptographically bound identifier.

物理部署应包含应用信息中特殊数据传输的防篡改硬件。比如一米可能有一个内置 GPS 来确保它未在一个未授权位置使用。[IEEE 802.1AR] 是一个操作机制标准，利用密码绑定识别来授权设备 ID。

5.4.8 Detecting abnormal behaviors 异常行为检测

Server implementations might monitor Client behavior to detect potential security incidents. For example:

服务器操作能监测终端行为以此检测潜在安全威胁，如：

- Repeated connection attempts
- 反复尝试连接
- Repeated authentication attempts
- 反复尝试授权
- Abnormal termination of connections
- 异常的终止连接
-

- Topic scanning (attempts to send or subscribe to many topics)
- 主题扫描（尝试发送或描述主题）
- Sending undeliverable messages (no subscribers to the topics)
- 发送无法投递的信息（主题无描述）
- Clients that connect but do not send data
- 连接的终端不发送数据

Server implementations might disconnect Clients that breach its security rules.
服务器可终止不合安全的终端的连接

Server implementations detecting unwelcome behavior might implement a dynamic block list based on identifiers such as IP address or Client Identifier.
服务器检测到不善行为可建立基于诸如 ip 地址或终端 id 的动态阻止列表。

Deployments might use network level controls (where available) to implement rate limiting or blocking based on IP address or other information.
部署可使用网络级别控制（若有的话）来设立基于 ip 地址或其他信息的等级限制或阻止。

5.4.9 Other security considerations 其他安全隐患

If Client or Server SSL certificates are lost or it is considered that they might be compromised they should be revoked (utilizing CRLs [RFC5280] and/or OSCP [RFC6960]).
终端或服务器 SSL 证书丢失或者被认作受到威胁，他们应当予以吊销。

Client or Server authentication credentials, such as User Name and Password, that are lost or considered compromised should be revoked and/or reissued.
终端或服务器授权证书，比如用户名和密码丢失或是被认作受到威胁应给予吊销或补发

In the case of long lasting connections:
在持续连接下

- Client and Server implementations using TLS [RFC5246] should allow for session renegotiation to establish new cryptographic parameters (replace session keys, change cipher suites, change authentication credentials).
- 利用 TLS 的终端和服务器应允许会话重新协商来建立新的加密参数（更换会话密钥，更换密码套件 更换授权证书）
- Servers may disconnect Clients and require them to re-authenticate with new credentials.
- 服务器可以中断与终端的连接并要求他们重新授权新的证书

Constrained devices and Clients on constrained networks can make use of TLS session resumption [RFC5077], in order to reduce the costs of reconnecting TLS [RFC5246] sessions.
为减少重新连接 TLS 的花费，受限设备和终端的网络连接可利用 TLS 会话恢复

Clients connected to a Server have a transitive trust relationship with other Clients connected to the same Server and who have authority to publish data on the same topics.

与服务器连接的终端与同一服务器的其他终端，和有权对同一主题发布数据的服务器存在可信传递关系。

5.4.10 Use of SOCKS socks 应用

Implementations of Clients should be aware that some environments will require the use of SOCKSv5 [RFC1928] proxies to make outbound Network Connections. Some MQTT implementations could make use of alternative secured tunnels (e.g. SSH) through the use of SOCKS. Where implementations choose to use SOCKS, they should support both anonymous and user-name password authenticating SOCKS proxies. In the latter case, implementations should be aware that SOCKS authentication might occur in plain-text and so should avoid using the same credentials for connection to a MQTT Server.

终端操作时应注意一些情况下会要求利用 SOCKSv5 代理来进行外部的网络连接。一些 MQTT 操作在使用 SOCKs 过程中会利用备选固定通道（如 SSH)。选择使用 SOCKS 时，他们应支持匿名和用户名密码授权 SOCKS 代理两种形式。后一种情况下，操作时需注意纯文本中可能出现的 SOCKS 授权，应避免使用同样的证书来连接 MQTT 服务器。

5.4.11 Security profiles 安全性配置文件

Implementers and solution designers might wish to consider security as a set of profiles which can be applied to the MQTT protocol. An example of a layered security hierarchy is presented below.

操作者和方案策划者不妨考虑将安全性作为一种配置文件应用于 MQTT 协议。以下是一个层次安全等级的例子

5.4.11.1 Clear communication profile 明确通讯协议

When using the clear communication profile, the MQTT protocol runs over an open network with no additional secure communication mechanisms in place.

当使用明确通信协议时，MQTT 协议在一个没有其他安全通信机制的开放网络中运行。

5.4.11.2 Secured network communication profile 安全网络通讯协议

When using the secured network communication profile, the MQTT protocol runs over a physical or virtual network which has security controls e.g., VPNs or physically secure network.

当使用安全网络通讯协议时，MQTT 协议在物理的或虚拟的拥有安全控制网络中运行比如 VPN 或物理保护网络。

5.4.11.3 Secured transport profile 传输安全协议

When using the secured transport profile, the MQTT protocol runs over a physical or virtual network and using TLS [RFC5246] which provides authentication, integrity and privacy.

当使用传输安全协议时，MQTT 协议在物理或虚拟的网络中运行并使用授权的完整的私密的 TLS。

TLS [RFC5246] Client authentication can be used in addition to – or in place of – MQTT Client authentication as provided by the Username and Password fields.

TLS 终端授权可替代用户名和密码给 MQTT 终端授权

5.4.11.4 Industry specific security profiles 行业专用安全配置文件

It is anticipated that the MQTT protocol will be designed into industry specific application profiles, each defining a threat model and the specific security mechanisms to be used to address these threats. Recommendations for specific security mechanisms will often be taken from existing works including:

预计 MQTT 协议将作为特定行业应用规范，定义威胁模型和特定安全机制以此解决这些威胁。专用安全机制建议来源于现存工作包括：

[NISTCSF] NIST Cyber Security Framework

[NIST7628] NISTIR 7628 Guidelines for Smart Grid Cyber Security

[FIPS1402] Security Requirements for Cryptographic Modules (FIPS PUB 140-2)

[PCIDSS] PCI-DSS Payment Card Industry Data Security Standard

[NSAB] NSA Suite B Cryptography

6 Using WebSocket as a network transport 用 WEBSOCKET 作为网络传输

If MQTT is transported over a WebSocket [RFC6455] connection, the following conditions apply:

若 MQTT 通过 WEBSocket 连接传输，应用于以下状况

- MQTT Control Packets MUST be sent in WebSocket binary data frames. If any other type of data frame is received the recipient MUST close the Network Connection [MQTT-6.0.0-1].
- MQTT 控制包必须在 WEBSOCKET 二进制数据框发送。若果收到其他类型 的数据框就，接受者必须关闭网络连接
- A single WebSocket data frame can contain multiple or partial MQTT Control Packets. The receiver MUST NOT assume that MQTT Control Packets are aligned on WebSocket frame boundaries [MQTT-6.0.0-2].、

- 单一 WEBSOKET 数据框可包含多个或部分 MQTT 控制包。接受者不能假定 MQTT 控制包与 WEBSOCKET 框架边界对齐。
- The client MUST include “mqtt” in the list of WebSocket Sub Protocols it offers [MQTT-6.0.0-3].
- 终端的 Websocket 子协议名单中必须包括 MQTT
- The WebSocket Sub Protocol name selected and returned by the server MUST be “mqtt” [MQTT-6.0.0-4].
- 服务器的 WEBSOCKET 子协议名称选择和退还必须为 MQTT
- The WebSocket URI used to connect the client and server has no impact on the MQTT protocol.
- 曾连接服务器和终端的 WEBSOCKET URI 对 MQTT 协议无影响

6.1 IANA Considerations

This specification requests IANA to register the WebSocket MQTT sub-protocol under the “WebSocket Subprotocol Name” registry with the following data:

该规范要求 IANA 以以下数据在 websocket 子协议名称下注册

Figure 6.1 - IANA WebSocket Identifier

Subprotocol Identifier	mqtt
Subprotocol Common Name	mqtt
Subprotocol Definition	http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html

7 Conformance 一致性

The MQTT specification defines conformance for MQTT Client implementations and MQTT Server implementations.

MQTT 规范定义了 MQTT 终端与服务器的一致性

An MQTT implementation MAY conform as both an MQTT Client and MQTT Server implementation. A Server that both accepts inbound connections and establishes outbound connections to other Servers MUST conform as both an MQTT Client and MQTT Server [MQTT-7.0.0-1].

MQTT 操作必须与 MQTT 客户端和服务器一致。一个服务器不仅接收内部连接还能建立与其他服务器的外部连接必须达到 MQTT 服务器和终端的一致性。

Conformant implementations MUST NOT require the use of any extensions defined outside of this specification in order to interoperate with any other conformant implementation [MQTT-7.0.0-2].

为实现与其他操作的互动，禁止使用规范定义外的任何扩展

7.1 Conformance Targets

7.1.1 MQTT Server

An MQTT Server conforms to this specification only if it satisfies all the statements below:

只要符文以下情况，MQTT 服务器遵从该规范

1. The format of all Control Packets that the Server sends matches the format described in Chapter 2 and Chapter 3. 服务器发送的控制包的格式与第二第三张描述的格式相符

2. It follows the Topic matching rules described in Section 4.7.遵循与 4.7 中描述的规则相符的主题

3. It satisfies all of the MUST level requirements in the following chapters that are identified except for those that only apply to the Client: 除了那些仅适用于终端，满足以下认证的章节中所有等级要求

- Chapter 1 - Introduction
- Chapter 2 - MQTT Control Packet format
- Chapter 3 - MQTT Control Packets
- Chapter 4 - Operational behavior
- Chapter 6 - (if MQTT is transported over a WebSocket connection)
- Chapter 7 - Conformance Targets

A conformant Server MUST support the use of one or more underlying transport protocols that provide an ordered, lossless, stream of bytes from the Client to Server and Server to Client [MQTT-7.1.1-1]. However conformance does not depend on it supporting any specific transport protocols. A Server MAY support any of the transport protocols listed in Section 4.2, or any other transport protocol that meets the requirements of [MQTT-7.1.1-1].

一个一致性服务器必须支持一个或多个能够有序无丢失的实现服务器与终端传送的基本传送协议。然而一致性不取决于他支持的任何特定传输协议。服务器可支持任何列在 4.2 里的传送协议，或任何达到要求的传送协议。

7.1.2 MQTT Client MQTT 终端

An MQTT Client conforms to this specification only if it satisfies all the statements below:

达到以下状况时，MQTT 终端遵循该规范

1. The format of all Control Packets that the Client sends matches the format described in Chapter 2 and Chapter 3. 终端发送的所有控制包的格式与第二第三张描述的格式相符

2. It satisfies all of the MUST level requirements in the following chapters that are identified except for those that only apply to the Server: 除了那些仅适用于服务器的，满足以下认证的章节中所有等级要求

- Chapter 1 - Introduction
- Chapter 2 - MQTT Control Packet format
- Chapter 3 - MQTT Control Packets
- Chapter 4 - Operational behavior
- Chapter 6 - (if MQTT is transported over a WebSocket connection)
- Chapter 7 - Conformance Targets

A conformant Client MUST support the use of one or more underlying transport protocols that provide an ordered, lossless, stream of bytes from the Client to Server and Server to Client [MQTT-7.1.2-1]. However conformance does not depend on it supporting any specific transport protocols. A Client MAY support any of the transport protocols listed in Section 4.2, or any other transport protocol that meets the requirements of [MQTT-7.1.2-1].

一个一致性终端必须支持一个或多个能够有序无丢失的实现服务器与终端传送的基本传送协议。然而一致性不取决于他支持的任何特定传输协议。终端可支持任何列在 4.2 里的传送协议，或任何达到要求的传送协议。

欢迎关注硬件十万个为什么