

The PPP Encryption Control Protocol (ECP)

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

The Point-to-Point Protocol (PPP) [1] provides a standard method for transporting multi-protocol datagrams over point-to-point links. PPP also defines an extensible Link Control Protocol.

This document defines a method for negotiating data encryption over PPP links.

Conventions

The following language conventions are used in the items of specification in this document:

- o MUST -- the item is an absolute requirement of the specification. MUST is only used where it is actually required for interoperation, not to try to impose a particular method on implementors where not required for interoperability.
- o SHOULD -- the item should be followed for all but exceptional circumstances.
- o MAY or optional -- the item is truly optional and may be followed or ignored according to the needs of the implementor.

The words "should" and "may" are also used, in lower case, in their more ordinary senses.

Table of Contents

1. Introduction	2
2. Encryption Control Protocol (ECP)	2
2.1 Sending Encrypted Datagrams	3
3. Additional Packets	4
3.1 Reset-Request and Reset-Ack	5
4. ECP Configuration Options	6
4.1 Proprietary Encryption OUI	7
4.2 Publicly Available Encryption Types	8
4.3 Negotiating an Encryption Algorithm	9
5. Security Considerations	10

1. Introduction

In order to establish communications over a PPP link, each end of the link must first send LCP packets to configure and test the data link during Link Establishment phase. After the link has been established, optional facilities may be negotiated as needed.

One such facility is data encryption. A wide variety of encryption methods may be negotiated, although typically only one method is used in each direction of the link.

A different encryption algorithm may be negotiated in each direction, for speed, cost, memory or other considerations.

2. Encryption Control Protocol (ECP)

The Encryption Control Protocol (ECP) is responsible for configuring and enabling data encryption algorithms on both ends of the point-to-point link.

ECP uses the same packet exchange mechanism as the Link Control Protocol (LCP). ECP packets may not be exchanged until PPP has reached the Network-Layer Protocol phase. ECP packets received before this phase is reached should be silently discarded.

The Encryption Control Protocol is exactly the same as LCP [1] with the following exceptions:

Frame Modifications

The packet may utilise any modifications to the basic frame format which have been negotiated during the Link Establishment phase.

Data Link Layer Protocol Field

Exactly one ECP packet is encapsulated in the PPP Information field, where the PPP Protocol field indicates type hex 8053 (Encryption Control Protocol).

When individual link data encryption is used in a multiple link connection to a single destination [2], the PPP Protocol field indicates type hex 8055 (Individual link Encryption Control Protocol).

Code field

ECP uses (decimal) codes 1 through 7 (Configure-Request, Configure-Ack, Configure-Nak, Configure-Reject, Terminate-Request, Terminate-Ack and Code-Reject); And may also use code 14 (Reset-Request) and code 15 (Reset-Ack). Other codes should be treated as unrecognised and should result in Code-Rejects.

Negotiation

ECP packets may not be exchanged until PPP has reached the Network-Layer Protocol phase. An implementation should be prepared to wait for Authentication and Link Quality Determination to finish before timing out waiting for a Configure-Ack or other response.

An implementation MUST NOT transmit data until ECP negotiation has completed successfully. If ECP negotiation is not successful the link SHOULD be brought down.

Configuration Option Types

ECP has a distinct set of Configuration Options.

2.1 Sending Encrypted Datagrams

Before any encrypted packets may be communicated, PPP must reach the Network-Layer Protocol phase, and the Encryption Control Protocol must reach the Opened state.

An encrypted packet is encapsulated in the PPP Information field, where the PPP Protocol field indicates type hex 0053 (Encrypted datagram).

When using multiple PPP links to a single destination [2], there are two methods of employing data encryption:

- o The first method is to encrypt the data prior to sending it out through the multiple links.

The PPP Protocol field MUST indicate type hex 0053.

- o The second is to treat each link as a separate connection, that may or may not have encryption enabled.

On links which have negotiated encryption, the PPP Protocol field MUST be type hex 0055 (Individual link encrypted datagram).

Only one encryption algorithm in each direction is in use at a time, and that is negotiated prior to sending the first encrypted frame. The PPP Protocol field of the encrypted datagram indicates that the frame is encrypted, but not the algorithm with which it was encrypted.

The maximum length of an encrypted packet transmitted over a PPP link is the same as the maximum length of the Information field of a PPP encapsulated packet. If the encryption algorithm is likely to increase the size of the message beyond that, multilink should also be negotiated to allow fragmentation of the frames (even if only using a single link).

If the encryption algorithm carries history between frames, the encryption algorithm must supply a way of determining if it is passing data reliably, or it must require the use of a reliable transport such as LAPB [3].

Compression may also be negotiated using the Compression Control Protocol [5]. To ensure interoperability, plain text MUST be:

- o First compressed.
- o Then encrypted.

This order has been chosen since it should result in smaller output and more secure encryption.

3. Additional Packets

The Packet format and basic facilities are already defined for LCP [1].

Up-to-date values of the ECP Code field are specified in the most recent "Assigned Numbers" RFC [4]. This specification concerns the following values:

14	Reset-Request
15	Reset-Ack

3.1 Reset-Request and Reset-Ack

Description

ECP includes Reset-Request and Reset-Ack Codes in order to provide a mechanism for indicating a decryption failure in one direction of a decrypted link without affecting traffic in the other direction. Some encryption algorithms may not require this mechanism.

Individual algorithms need to specify a mechanism for determining how to detect a decryption failure. On initial detection of a decryption failure, an ECP implementation SHOULD transmit an ECP packet with the Code field set to 14 (Reset-Request). The Data field may be filled with any desired data.

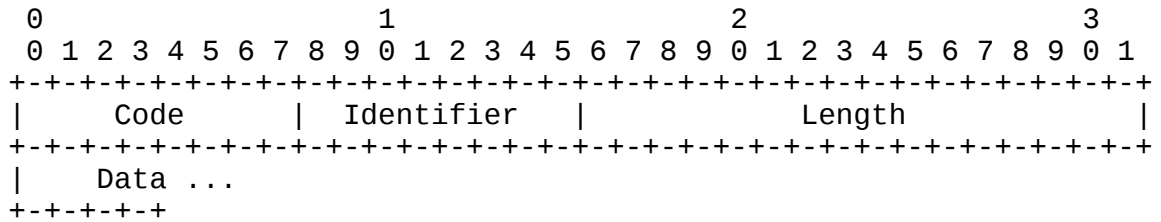
Once a Reset-Request has been sent, any encrypted packets received are discarded. Further Reset-Requests MAY be sent with the same Identifier, until a valid Reset-Ack is received.

When the link is busy, one decryption error is usually followed by several more before the Reset-Ack can be received. It is undesirable to transmit Reset-Requests more frequently than the round-trip-time of the link, since this will result in redundant Reset-Requests and Reset-Acks being transmitted and processed. The receiver MAY elect to limit transmission of Reset-Requests (to say one per second) while a Reset-Ack is outstanding.

Upon reception of a Reset-Request, the transmitting encrypter is reset to an initial state. An ECP packet MUST be transmitted with the Code field set to 15 (Reset-Ack), the Identifier field copied from the Reset-Request packet, and the Data field filled with any desired data.

On receipt of a Reset-Ack, the receiving decrypter is reset to an initial state. Since there may be several Reset-Acks in the pipe, the decrypter MUST be reset for each Reset-Ack which matches the currently expected identifier.

A summary of the Reset-Request and Reset-Ack packet formats is shown below. The fields are transmitted from left to right.



Code

14 for Reset-Request;

15 for Reset-Ack.

Identifier

On transmission, the Identifier field MUST be changed whenever the content of the Data field changes, and whenever a valid reply has been received for a previous request. For retransmissions, the Identifier SHOULD remain unchanged.

On reception, the Identifier field of the Reset-Request is copied into the Identifier field of the Reset-Ack packet.

Data

The Data field is zero or more octets and contains uninterpreted data for use by the sender. The data may consist of any binary value and may be of any length from zero to the peer's established MRU minus four.

4. ECP Configuration Options

ECP Configuration Options allow negotiation of encryption algorithms and their parameters. ECP uses the same Configuration Option format defined for LCP [1], with a separate set of Options.

Configuration Options, in this protocol, indicate algorithms that the receiver is willing or able to use to decrypt data sent by the sender. Systems may offer to accept several algorithms, and negotiate a single one that will be used.

Up-to-date values of the ECP Option Type field are specified in the most recent "Assigned Numbers" RFC [4]. Current values are assigned as follows:

ECP Option	Encryption type
0	OUI
1	DESE

All compliant ECP implementations SHOULD implement ECP option 1 - the PPP DES Encryption Protocol (DESE) [6].

Vendors who want to use proprietary encryption MAY use the OUI mechanism to negotiate these without recourse to requesting an assigned option number from the Internet Assigned Numbers Authority. All other encryption options are registered by IANA. At the time of writing only DESE (option 1) is registered. Other registered options may be found by referring to future versions of the Assigned Numbers RFC.

4.1 Proprietary Encryption OUI

Description

This Configuration Option provides a way to negotiate the use of a proprietary encryption protocol.

Vendor's encryption protocols are distinguished from each other by means of an Organisationally Unique Identifier (OUI), namely the first three octets of a Vendor's Ethernet address assigned by IEEE 802.

Since the first matching encryption will be used, it is recommended that any known OUI encryption options be transmitted first, before the common options are used.

Before accepting this option, the implementation must verify that the OUI identifies a proprietary algorithm that the implementation can decrypt, and that any vendor specific negotiation values are fully understood.

A summary of the Proprietary Encryption OUI Configuration Option format is shown below. The fields are transmitted from left to right.

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      Type      |      Length      |      OUI ...
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|      OUI      |      Subtype      |      Values...
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Type

0

Length

>= 6

IEEE OUI

The IEEE OUI is the most significant three octets of an Ethernet Physical Address, assigned to the vendor by IEEE 802. This identifies the option as being proprietary to the indicated vendor. The bits within the octet are in canonical order, and the most significant octet is transmitted first.

Subtype

This field is specific to each OUI, and indicates an encryption type for that OUI. There is no standardisation for this field. Each OUI implements its own values.

Values

This field is zero or more octets, and contains additional data as determined by the vendor's encryption protocol.

4.2 Publicly Available Encryption Types

Description

These Configuration Options provide a way to negotiate the use of a publicly defined encryption algorithm.

These protocols should be made available to interested parties, but may have certain licencing or export restrictions associated with them. For additional information, refer to the encryption protocol documents that define each of the encryption types.

A summary of the Encryption Type Configuration Option format is shown below. The fields are transmitted from left to right.

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|      Type      |      Length      | Values...
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

```

Type

1 to 254, indicating the encryption protocol option being negotiated. DESE [6] is option type 1. Refer to the latest Assigned Numbers RFC for other encryption protocols.

Length

>= 2

Values

This field is zero or more octets, and contains additional data as determined by the encryption protocol.

4.3 Negotiating an Encryption Algorithm

ECP uses LCP option negotiation techniques to negotiate encryption algorithms. In contrast with most other LCP or NCP negotiation of multiple options, ECP negotiation is expected to converge on a single mutually agreeable option (encryption algorithm) - or none. Encryption SHOULD be negotiated in both directions, but the algorithms MAY be different.

An implementation willing to decrypt using a particular encryption algorithm (or one of a set of algorithms) offers the algorithm(s) as an option (or options) in an ECP Configure-Request - call this end the Decrypter; call the peer the Encrypter.

A Decrypter supporting more than one encryption algorithm may send a Configure-Request containing either:

- o an ordered list of options, with the most-preferred encryption algorithm coming first.
- o Or may just offer its preferred (not already Rejected) option.

An Encrypter wishing to accept the first option (of several) MAY Configure-Ack ALL Options to indicate complete acceptance of the first-listed, preferred, algorithm.

Otherwise, if the Encrypter does not recognise - or is unwilling to support - an option it MUST send a Configure-Reject for that option. Where more than one option is offered, the Encrypter SHOULD Configure-Reject all but a single preferred option.

If the Encrypter Configure-Rejects all offered ECP options - and the Decrypter has no further (non-rejected) options it can offer in a Configure-Request - the Encrypter SHOULD take the link down.

If the Encrypter recognises an option, but it is not acceptable due to values in the request (or optional parameters not in the request), it MUST send a Configure-Nak with the option modified appropriately. The Configure-Nak MUST contain only those options that will be acceptable. The Decrypter SHOULD send a new Configure-Request with only the single preferred option, adjusted as specified in the Configure-Nak.

5. Security Considerations

Negotiation of encryption using PPP is designed to provide protection against eavesdropping on that link. The strength of the protection is dependent on the encryption algorithm used and the care with which any 'secret' used by the encryption algorithm is protected.

It must be recognised that complete security can only be obtained through end-to-end security between hosts.

References

- [1] Simpson, W., Editor; "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, Daydreamer, July 1994.
- [2] Sklower, K., Lloyd, B., McGregor, G. and and D. Carr, "The PPP Multilink Protocol (MP)", RFC 1717, University of California, Berkeley, November 1994.
- [3] Rand, D., "PPP Reliable Transmission", RFC 1663, Novell, July 1994.
- [4] Reynolds, J., and Postel, J.; "ASSIGNED NUMBERS", STD 2, RFC 1700, USC/Information Sciences Institute, October 1994.
- [5] Rand, D., "The PPP Compression Control Protocol (CCP)", RFC 1962, Novell, June 1996.

- [6] Sklower, K., and G. Meyer, "The PPP DES Encryption Protocol (DESE)", RFC 1969, University of California, Berkeley, June 1996.

Acknowledgements

The style and approach of this proposal owes much to the work on the Compression CP [5].

Chair's Address

The working group can be contacted via the current chair:

Karl Fox
Ascend Communications
3518 Riverside Drive, Suite 101
Columbus, Ohio 43221

EMail: karl@ascend.com

Author's Address

Gerry Meyer
Spider Systems
Stanwell Street
Edinburgh EH6 5NG
Scotland, UK

Phone: (UK) 131 554 9424
Fax: (UK) 131 554 0649
EMail: gerry@spider.co.uk

