LASAT Embedded Unit Configuration Protocol

Document Version: 1.00
Release date: 2001-03-26.
Document maintainer: Thomas Hansen tha@lasat.com

Revision history:

1.00	2001-03-26	*sip*	Initial release based on document Eucp.txt by THH
1.01	2001-03-26	*sip*	Update on TCP transfers.

Purpose:

This document describes the EUCP protocol and should be regarded as a “standard”.

Table of contents

Introduction

This document describes a method for initial configuration and maintenance for Lasat networks embedded units.

It includes a description of the configuration process, the roles of the client and the server, and an in-depth description of the Embedded Unit Configuration Protocol, EUCP.

Design goals

The EUCP protocol should provide the following:

- A simple but secure interface for initial IP configuration of an embedded unit.
- A method for upgrading the firmware of an embedded unit.
- A method of obtaining information about the IP configuration of an embedded unit.
- The protocol should be simple to implement on both the client and server side.
- Interoperability with existing devices and protocols in a mixed network environment.

Considerations

Two different approaches towards the design of a configuration protocol has been considered:

1) Using an Ethernet-level protocol with its own ether_type.
2) Using broadcast IP UDP datagrams.

Each of these options offer advantages and weaknesses, outlined here:

1) Ethernet-level protocol

Advantages:
- No need for a TCP/IP stack.
- Will not interfere with an existing TCP/IP network.
- Simplicity.
- Will only generate one broadcast packet per session.

Disadvantages:
- Broadcast packets may confuse existing equipment.
- Requires modification of the kernel, possibly this can be limited to the use of a loadable module, but it may well require further kernel patching, and it will certainly require maintenance for each new kernel upgrade.
- May prove difficult to implement in a Windows environment.

2) IP UPD based protocol.

Advantages:
- Can be implemented completely as a userspace application, utilizing the standard TCP/IP kernel interface - no kernel patching - on both Windows and Unix.
- Will work with any type of medium capable of broadcasting, not limited to Ethernet.
- Relies on a well tested code base.
- Simple implementation.

Disadvantages:
- Generates broadcast IP datagrams even after the initial handshake, but only until an IP address has been assigned to the embedded unit.
- Requires a running TCP/IP stack.

Given these facts, the UDP model seems the best choice since the use of broadcast packets will still be very limited. Also, depending on the protocol stack used, it may be possible to unicast the response packets when the destination IP address is not known (but the hardware address is), although this will definitely require additional work.

Terminology

The following terms will be used throughout this document:

Client: A host (Windows PC) running the Configuration Utility.

Server: The Lasat Embedded Unit running the EUCP server software in either Maintenance or Report mode.

EUCP Discover Request: A broadcast request sent by the client, requesting present servers to identify themselves.

EUCP Identification Message: A message sent back to a client, including model and serial number, state, software revision and primary Private Ethernet IP configuration.

EUCP Configure Request: A message sent from the client to a unit, containing an IP address and netmask to be assigned to the primary Private Ethernet port and/or the ID of a stored configuration to make active.

EUCP Transfer Request: A message sent from the client to prepare for a transfer of a Firmware Upgrade or Configuration to the unit, or sent to the client to indicate that a requested image will be sent.

EUCP Retrieve Request: A message sent from the client to request that a Firmware or Configuration image should be sent.

EUCP Confirmation Message: A message sent back to the client, indicating the result of the last Configure or Transfer request.

EUCP Image Data Packet: A message sent from the client, containing a fragment of an image to be transmitted.

Overview

The EUCP protocol is used for initial IP configuration of Lasat embedded units. It is only used for the transmission of non-sensitive information.

The EUCP messages are enclosed in standard IP UPD datagrams.

The protocol works as follow:

The Client, a software program running on a Windows PC, sends a broadcast EUCP Discover request.

Each Server - the embedded unit - sends a response indicating its serial number and model name, current status, EUCP mode, and IP address (if any).

The Client will then present the user with a summary of units found, and allow the user to select a unit to configure or read status from.

The Server listens for EUCP requests only on its Private Ethernet interface.

The Server can operate in two modes: Maintenance mode and Report mode. Maintenance mode is automatically invoked when the embedded unit is new and has not yet been configured, and can be invoked by the user by physically pushing a button on the embedded unit at boot time. Report mode is active at all other times, i.e. when the unit is running normally.

In both modes, the Client has access to the following features:

- Examine the IP configuration of the primary (private) Ethernet interface
- Examine firmware version information
- See any error messages that may prevent the use of the web-based configuration interface of the unit.

In Maintenance mode, the Client can perform the following additional functions:

- Set the IP configuration of the primary (private) Ethernet interface
- Upgrade firmware (flash) from an image file

Packet format

The EUCP protocol uses a common packet format for communication between the client and the server. The EUCP packets are enclosed in standard IP UDP datagrams as described in RFC 768. The EUCP server listens on UDP port 30292 (hex 7654), and the EUCP client listens on UDP port 30293 (hex 7655).

For initial configuration, EUCP can work only when the client and the server is attached to the same physical network because of the need of broadcast packets. After initial configuration, the Server MUST only respond to either broadcast requests or requests originating from within the local network. This restriction is imposed for security reasons.

For EUCP Discovery Requests, and for Configuration Requests to a unit that does not know its IP address, the Destination address of the IP header is set to the broadcast address 255.255.255.255. This address means "broadcast to all units on the local cable" (RFC 919). This address will be translated to the broadcast hardware address in the Ethernet frame (i.e. 0xffffffffffff) by the IP layer.

The client MUST ONLY broadcast EUCP Discovery Requests and Configuration Requests. Any other request must be sent to the Server IP address after it has been assigned an IP address.

All integer values in the EUCP packets are stored in network byte order (higher order bytes are sent first) and are unsigned unless otherwise is explicitly specified.

The common header for all EUCP packets looks like this:

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	magic			
4	sessionid			
8	seqno			
12	type	protver	reserved	
16	Type dependent data			
.	.			
.	.			

All EUCP packets start with a 32-bit integer (network byte order) containing the magic number 0x45554350 to indicate that this is a valid EUCP packet.

The maximum packet length of any EUCP packet is 1400 bytes.

The next word contains a "sessionid". The sessionid field must be initialized to zero by the client, and each responding server will assign a randomly selected session-id which will be sent in the response. All subsequent communication MUST contain this session-ID, and both client and server MUST reject packets with an invalid session-ID. The session-ID replaces the IP-address to uniquely identify an unconfigured box.

The seqno field is zero for the first packet transmitted and incremented each time a packet is sent. It can be used to check that no packets have been lost.

The type field specifies the type of the packet and determines the contents of the rest of the packet. It can have one of the following values:

0x01 - EUCP Discover Request
0x02 - EUCP Identification Message
0x03 - EUCP Configure Request
0x04 - EUCP Transfer Request
0x05 - EUCP Retrieve Request
0x06 - EUCP Confirmation Message
0x07 - EUCP Image Data Packet
0x08 - EUCP Keepalive Request
0x09 - EUCP Transfer Complete Message
0x0a - EUCP Flash message
0x0b - EUCP Configure Lasat Request

The protver field specifies the protocol version, currently 0x01.

The format of each packet type is discussed below.

Protocol operation

The Client broadcasts an EUCP Discover Request.

1) All servers respond with an EUCP Identification Message, containing a randomly selected session ID that will be used subsequently in all communications with that server.

2) The client must wait for up to 5 seconds for servers to respond, and should then present a list of responding units to the user. The session ID will remain valid for up to 30 minutes, as long as the unit is

not powered down or rebooted. After this timeout a new session-ID should be obtained by re-sending an EUCP Discover Request.

3) For servers in either Report mode or Maintenance mode, the client can now send the following, for as long as the Session-ID is valid.

EUCP Retrieve Request

This initiates a transfer of Configuration Storage or Flash Memory to the client (see below under EUCP Transfer Request for how the transfer protocol works). The server will send a EUCP Transfer Request, which must be confirmed with a Confirmation Message, following the server will send EUCP Image Data Packets containing the actual data.

For Maintenance mode servers, the client may also send one of the following:

EUCP Configure Request

This request may contain a request to set a new IP address in the current configuration, and/or instruct the unit to load a different stored configuration.

The configuration ID to load is one of the following (decimal) values:

0	Active Configuration (selecting this has no effect since it is already the operating configuration).
1-10	Stored configuration 1-10.
11	The factory default configuration.

If the request contains both an IP address and a configuration to set, then the selected configuration is first copied to the Active Configuration, and then the IP address and Netmask is set in the Active Configuration.

The configure request may also have the reboot flag, in which case the unit will reboot and resume normal operations after the confirmation message has been sent (this renders the current session-ID invalid).

When the request has been processed, a EUCP Confirmation Message will be sent back to the client, indicating success or failure of the request.

EUCP Transfer Request

This indicates that the client wishes to upload one of the following:

1. An updated Firmware image.
2. A saved configuration.

When the Transfer Request has been sent, the server will respond with a EUCP Confirmation Message. If the status code of the Confirmation Message is zero, the client may start uploading the data in EUCP Image Data Packets. When the transfer is complete, the server will respond with another EUCP Confirmation Message.

EUCP Discover Request

This packet type contains no additional data.

EUCP Identification Message

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	ipaddr			
4	ipmask			

8	confstore		mode	status
12	fwdate			
16	serlen	fwverlen	modlen	smsglen
20	serial			
.	.			
N	fwver			
.	.			
N	modname			
.	.			
N	smsg			
.	.			

Field meanings:

ipaddr	The IP address of the unit if known, otherwise zero.
ipmask	The netmask of the unit if known, otherwise zero.
confstore	Bitmask indicating which stored configuration slots contain valid configurations that can be selected with a EUCP Configure Request (bit 0 = Configuration Slot 1).
mode	EUCP operation mode: 0=Report, 1=Maintenance.
status	Status report, 0=no error, nonzero=error.
fwdate	Firmware date, as number of seconds since Jan 01, 1970.
serlen	Length of the Serial Number string, including null terminator.
fwverlen	Length of the Firmware Version string, including null terminator.
modlen	Length of the Model Name string, including terminator.
smsglen	Length of the Status Message string, including null terminator.
serial	Null terminated Serial number.
fwver	Null terminated Firmware version string.
modname	Null terminated Model name.
smsg	Status message, if nonzero status code this null. terminated text string describes the error condition.

EUCP Configure Request

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	reqtype	setconf	pwdlen	RESERVED
4	ipaddr			
8	ipmask			
12	gateway			
16	password			
	.			

Field meanings:

reqtype Request type, bit mapped:
0x01 = Set IP address+mask of current configuration.
0x02 = Select new active configuration.
0x04 = Reboot to normal mode.
0x08 = Reboot to normal mode and enable kernel boot line editing.

setconf Configuration to make active.

pwdlen Password length.

ipaddr IP address to assign.

ipmask Netmask to assign.

gateway Gateway (Currently unused).

password Is the administrators new password.

If pwdlen is zero the password isn't set or changed.

EUCP Transfer Request

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	reqtype		Selection	
4	Imgsize			
8	md5sum			
12				
16				
20				

Field meanings:

reqtype Type of image that will be transfered:
0 - Software image
1 - Software image, use TCP to transfer it[1].
A software image can be either Firmware or configuration data. The type is indicated in the datastream. For a description of software image file structures look at appendix B.

selection If the software image is firmware this is the partition to transfer to. (currently, as the firmware is not partitioned the only valid value is 0).

If the software image is configuration data this indicates which stored configuration to transfer, 0 = Active Configuration, 1-10 = Stored Configurations.

imgsize Size of firmware upgrade image.

md5sum 128-bit MD5 digest of image file.

EUCP Retrieve Request

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	reqtype		selection	

Field meanings:

reqtype Type of image to retrieve:
0 - Firmware Image.
1 - Stored Configuration.

selection If reqtype = 0, this indicates which partition of the Firmware Image to transfer (currently, as the firmware is not partitioned the only valid value is 0).

[1] See Appendix C

If reqtype = 1, this indicates which stored configuration to download, 0 = Active Configuration, 1-10 = Stored Configurations.

EUCP Confirmation Message

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	reqtype	status	msglen	RESERVED
4	optdata			
8	message			
	.			

Field meanings:

reqtype — Type of confirmation:
0x01 = Configure Request processed.
0x02 = Transfer Request processed.
0x03 = Transfer completed.
0x04 = Image saved to flash.
0x05 = Resend from.
0x06 = Abort transfer.
0x07 = Image packet received

status — Zero if succesful, nonzero if error occured. If the request was a Transfer Request, a status code of zero indicates that the unit is ready to receive the image file. For a list of error codes look at appendix A.

msglen — Length of status message including null terminator.

optdata — File position if transfer error.

message — Status message indicating the result of the operation, null terminated.

EUCP Image Data Packet

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
0	Seq	eof	size	
4	pos			
8	data			
	.			

seq — Transfer sequence number.

eof — 1 marks End-Of-File, 2 means abort file transfer.

size — Number of bytes contained in this packet.

pos — Relative file position.

data — Binary image data.

EUCP Configure Lasat

	bit 0..7	bit 8..15	bit 16..23	bit 24..31
00	reqtype	vendor	prid	macaddr[0]
04	macaddr[1..12]			
08				
0c				
10	macaddr[0..11]			
14				
18				

1c	macaddr[12]	model[0..2]

20	model[3..62]		
24			
28			
2c			
30			
34			
38			
3c			
40			
44			
48			
4c			
50			
54			
58			
5c	model[63]	print_partno[0..2]	
60	print_partno[3..10]		
64			
68	print_partno[11]	print_serial[0..2]	
6c	print_serial[3..14]		
70			
74			
78	prod_partno[0..11]		
7c			
80			
84	prod_serial[0.11]		
88			
8c			
90	prod_serial[12..14]		passwd[0]
94	passwd[1..60]		
98			
9c			
A0			
A4			
A8			
Ac			
B0			
B4			
B8			
Bc			
C0			
C4			
C8			
Cc			
D0	passwd[61..63]		factory_default_req
D4	cfg[0]		
D8	cfg[1]		
Dc	cfg[2]		
E0	ipaddr		
E4	ipmask		
E8	changed	reboot	base_model
Ec	product_id		

reqtype Request type
0x01 – setup
0x04 - reboot

vendor	Vendor id
prid	Product id (0xff for version 2 of configure lasat request)
macaddr0	MAC of first ethernet controller
macaddr1	MAC of second ethernet controller
model	Model
print_partno	Print partno
print_serial	Print serial
prod_partno	Prod partno
prod_serial	Prod serial
passwd	Password
factory_default_req	0x01 - Restore default configuration
cfg	Configuration words
ipaddr	IP address
ipmask	IP mask
changed	Bit field bits 1 – IP address changed 2 – IP mask changed 3 – password changed 4 – vendor changed 5 – configuration word 0 changed 6 – configuration word 1 changed 7 – configuration word 2 changed 8 – request configuration (if this bit is set by client, server builds Configure Lasat Request and send it to client) 9 – base model changed 10 – product id changed 11 – request configuration v2 (same as ‘request configuration’ but returns version 2 of Configure Lasat Request)
Reboot	0x01 – reboot 0x02 – reboot and enable editing of kernel boot line
base_model	Base model (only in version 2)
product_id	Product id (only in version 2)

Appendix

A. Error codes returned in Confirmation messages.

The status field of a confirmation message will hold one of the following values.

Code	Description
0	No error.
1	Timeout waiting for data.
2	Packet out of sequence.
3	Tried to configure without being in maintenance mode.
4	Tried to set configuration to illegal slot (only 0 and 11 supported at this time).
5	Selection of configuration failed.
6	Couldn't set IP, netmask and password.
7	Aborting transfer. This may have one of the following causes: • Transfer was aborted by Client • An invalid data packet was received. • A data packet had wrong size. • There were too many transfer errors. • Data packet is out of sync. • Data block received exceeds the expected total size. • Received image doesn't match expected size.
8	Incorrect MD5 sum in transfer.
9	Error writing firmware image.
10	Error writing configuration image.
11	Transfer already in progress.
12	Another transfer is currently in progress.
13	The partition selected is invalid.
14	Invalid transfer type.
15	Insufficient memory to complete transfer.
16	Error reading firmware.
17	Trying to read from nonexisting configuration.
18	Error reading configuration
19	Transfer request type not recognized.
20	Software image isn't recognized format or type.
21	Unable to decrypt configuration data (password might be invalid).

B. File layout of software image files.

A software image file is a binary file. It either contains firmware data or configuration data. A firmware image can replace whatever current firmware is in the flash. Configuration data are contents of the "/var/data" directory to be uploaded. The software image header is as follows:

- 12 bytes Identification.
- 4 bytes Type.

The identification is a string describing the image. Currently only one string is supported "SPSOFTIMG00". The idea is that the last 2 digits may represent some sort of version number.

The type can be either 0 meaning a firmware image and 1 meaning a configuration image.

Configuration images are to be encrypted with tripple-des encryption. The first eigth bytes of the unencrypted image must also contain the string "13141516". This is used by the Safepipe to certify that the image was correctly decrypted.

C. TCP Transfer of software images

Before attemting TCP transfers client must assign temporary IP address to the server.

After sending EUCP Transfer Request with reqtype set to ‘Software image, TCP transfer’, client should check returned confirmation. If there is an indication of error, client must fall back to UDP transfer and resend Transfer Request with reqtype set to ‘Software image’. Otherwise client is free to connect the server on EUCP_TRANSFERPORT (0x7656/tcp) and start sending software image. Note, that server sends confirmation with reqtype set to ‘Image packet received’ on every non-zero read from TCP socket. On transfering all data client should close TCP connection and send 'Transfer complete request’.