IoT Updates with IPv6 Multicast

Brett Sheffield, Librecast Project

@brett_sheffield

#FOSDEM2020

Before we begin...

Multicast

”IP Multicast will play a prominent role on the Internet in the coming years. It is a requirement, not an option, if the Internet is going to scale. Multicast allows application developers to add more functionality without significantly impacting the network.”

– RFC 3170, Sep 2001

Multicast

Efficient

Multicast

Scalable

Multicast

Real-World

Multicast

Privacy

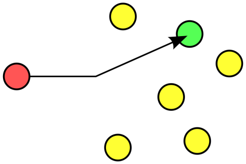
Multicast

Decentralisation

What is Multicast?

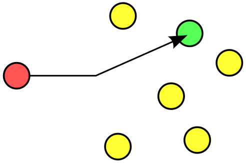
Definition

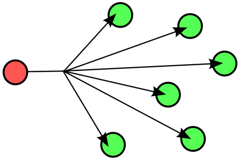
Definition



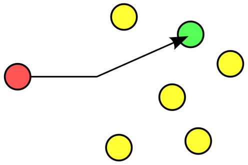
Unicast

Definition

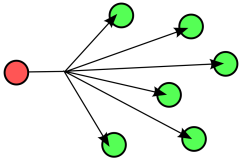

Unicast


Broadcast

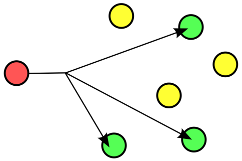
Definition



Unicast

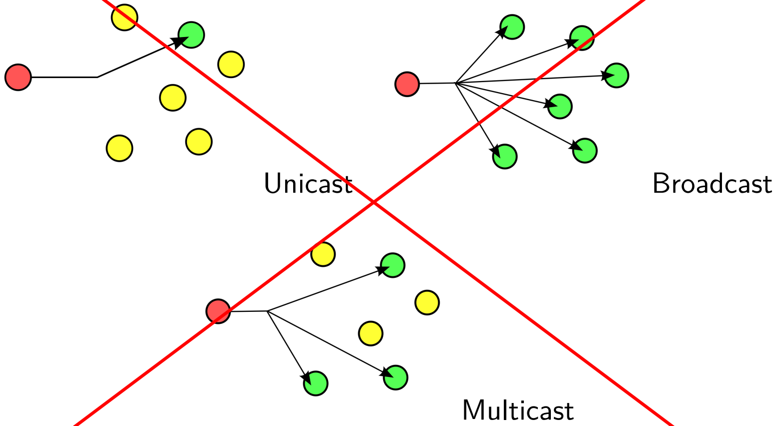


Broadcast



Multicast

Definition



What is Multicast?

Unicast, Broadcast

Multicast

What is Multicast?

Unicast, Broadcast PUSH

Multicast

What is Multicast?

Unicast, Broadcast	PUSH
Multicast	PULL

Multicast Misconceptions

Multicast Misconceptions

- only for streaming

Multicast Misconceptions

- only for streaming
- no use for video on demand

Multicast Misconceptions

- only for streaming
- no use for video on demand
- unreliable

Multicast Misconceptions

- only for streaming
- no use for video on demand
- unreliable
- insecure

Multicast Misconceptions

- only for streaming
- no use for video on demand
- unreliable
- insecure
- can't work on Internet

Multicast Misconceptions

- only for streaming
- no use for video on demand
- unreliable
- insecure
- can't work on Internet

Multicast is ...

Multicast is …
Group Communication

All Communication is Group Communication

IoT Updates

https://github.com/librestack/iotupd

Datagram:

Datagram:

- checksum

Datagram:

- checksum
- size of file

Datagram:

- checksum
- size of file
- size of chunk

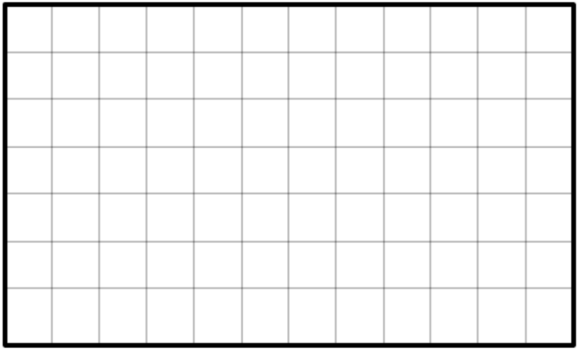
Datagram:

- checksum
- size of file
- size of chunk
- offset

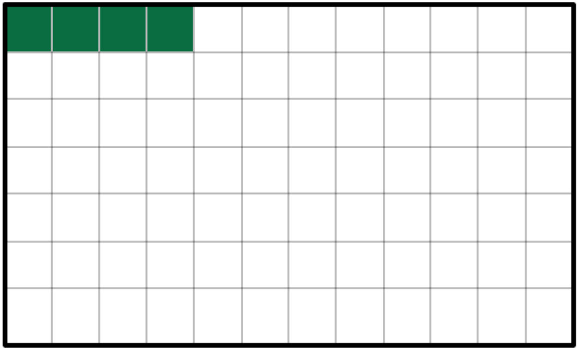
Datagram:

- checksum
- size of file
- size of chunk
- offset
- data

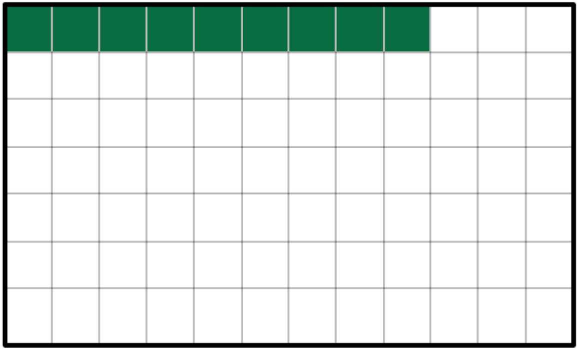
Receiving a File



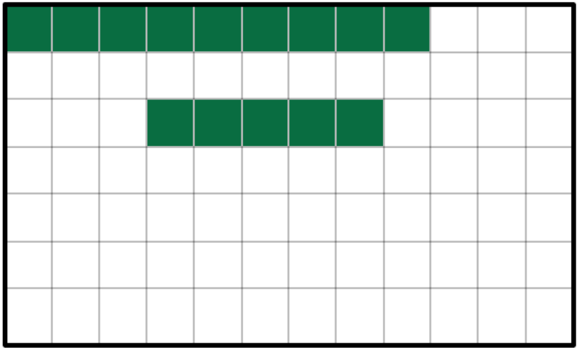
Receiving a File



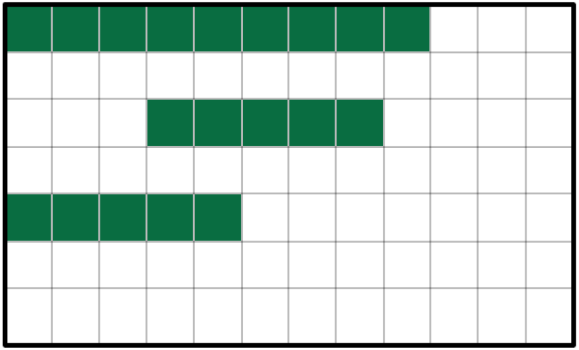
Receiving a File



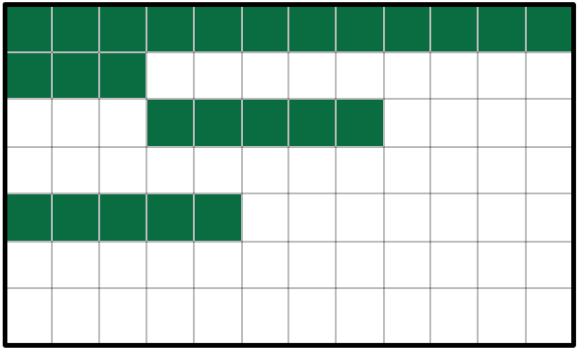
Receiving a File



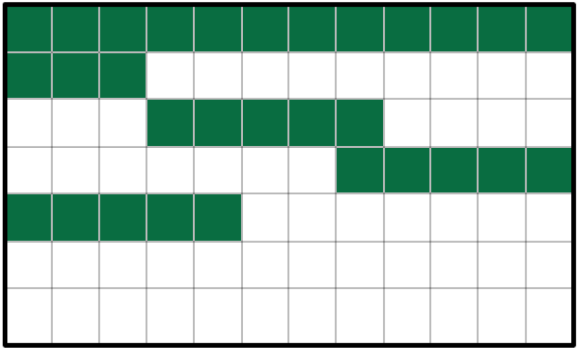
Receiving a File



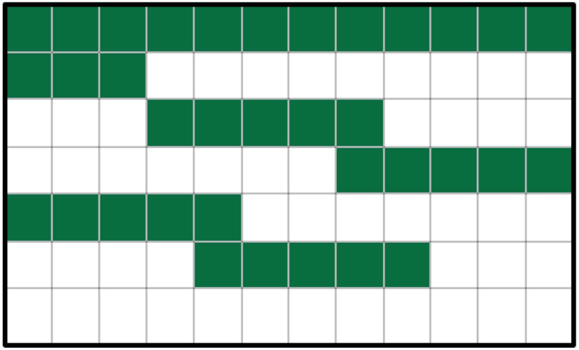
Receiving a File



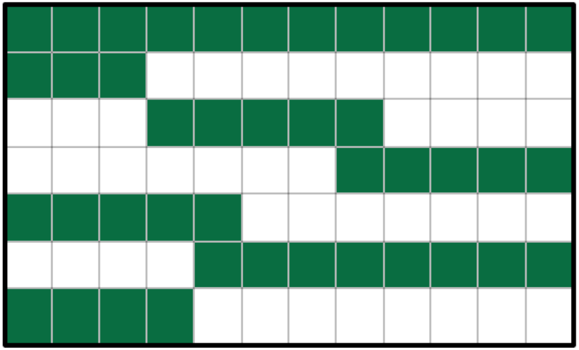
Receiving a File



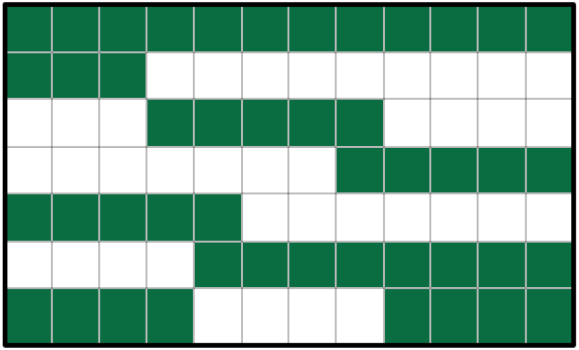
Receiving a File



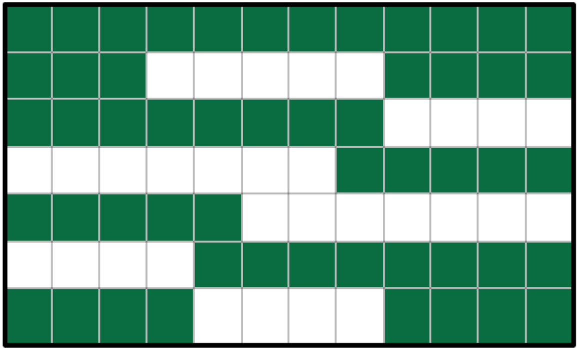
Receiving a File



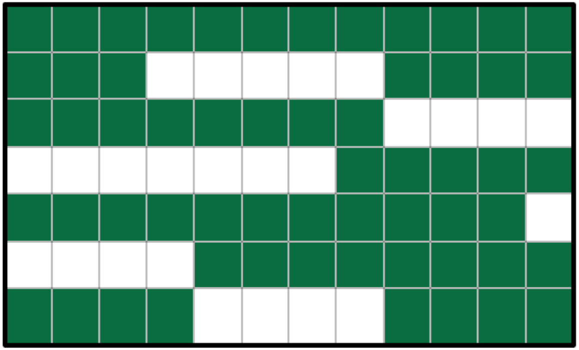
Receiving a File



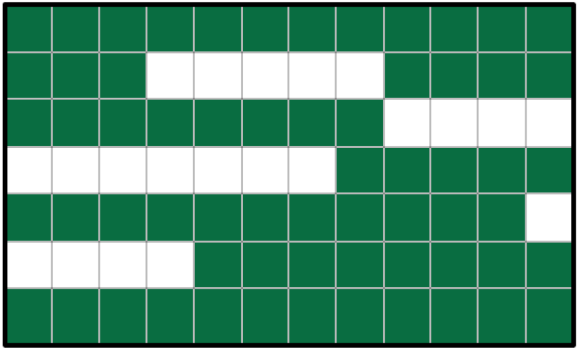
Receiving a File



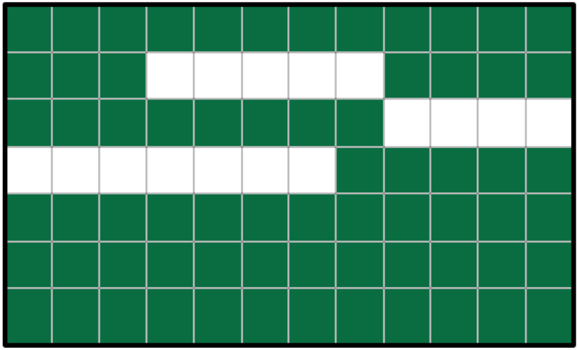
Receiving a File



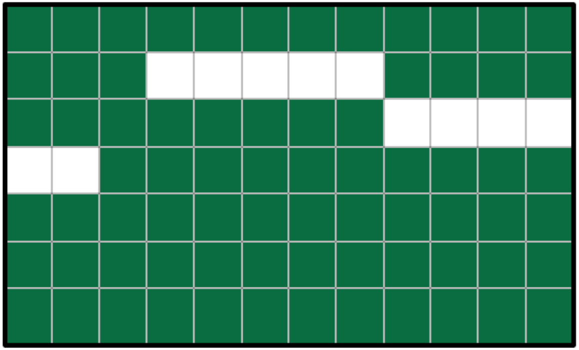
Receiving a File



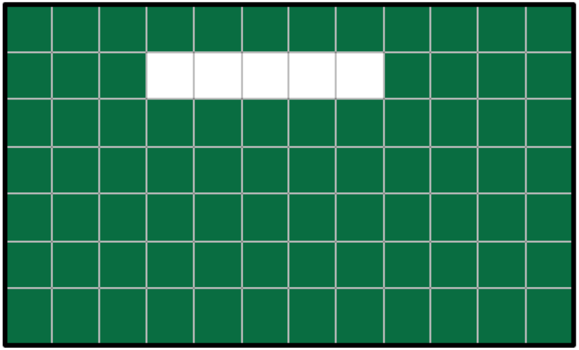
Receiving a File



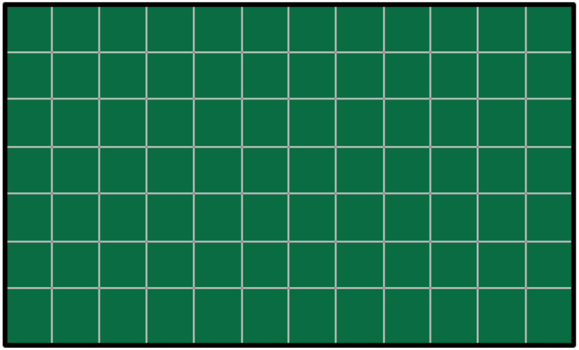
Receiving a File



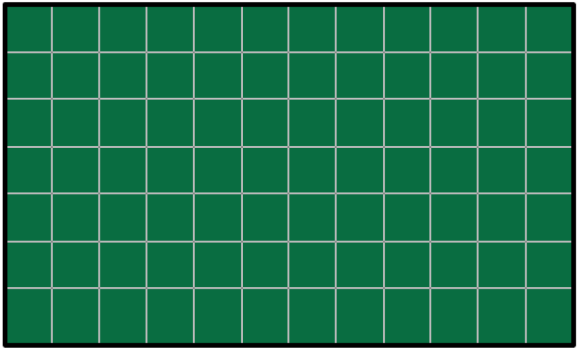
Receiving a File



Receiving a File

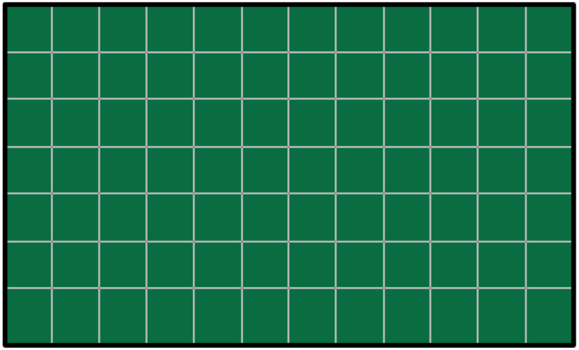


Receiving a File



Data received: 100%

Receiving a File



Data received: 100%

Checksum match

What Just Happened?

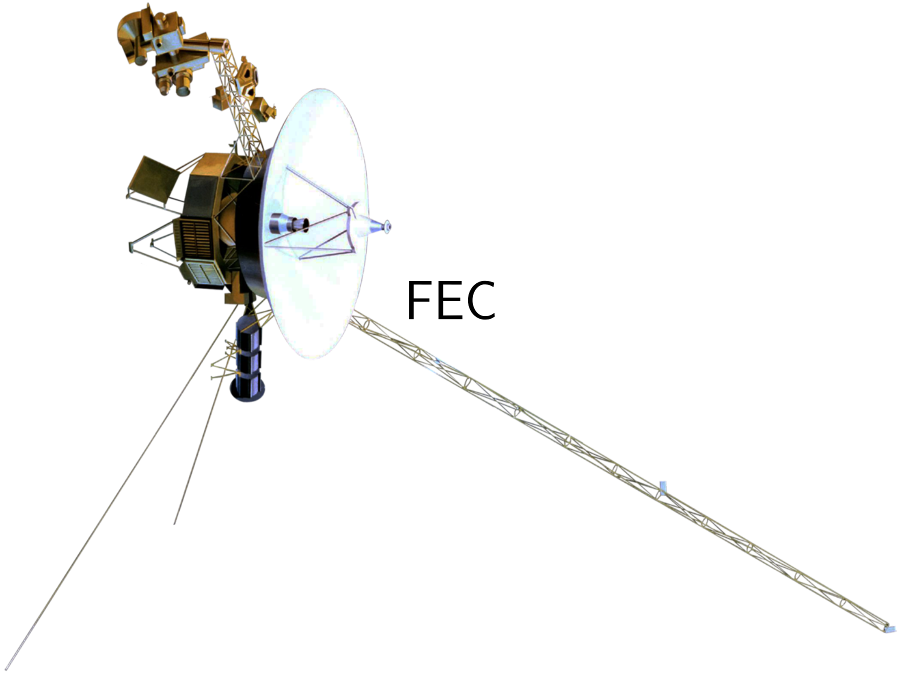
TCP/IP

Are there other ways we can achieve TCP/IP-like reliability?

PGM
(RFC 3208 - Experimental)

NACKs, Replay

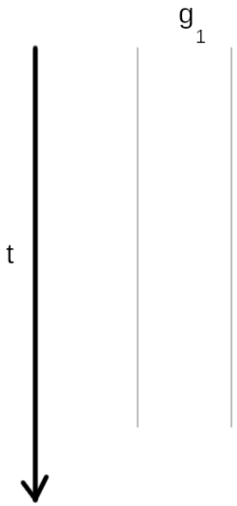
Loop and Repeat



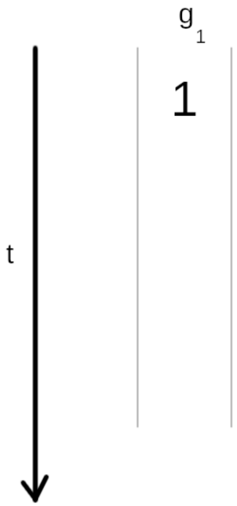
FEC

Flow Control

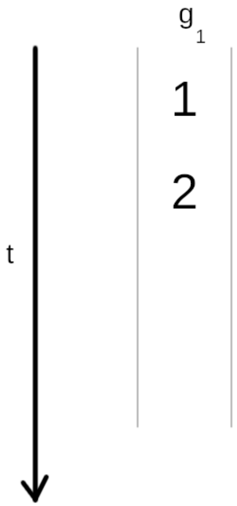
Flow Control



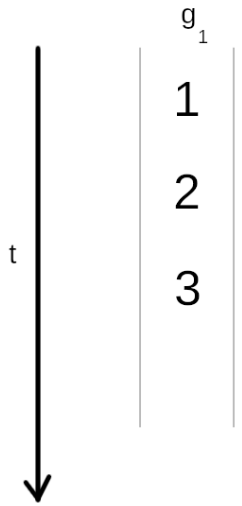
Flow Control



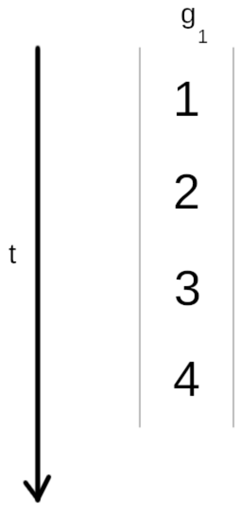
Flow Control



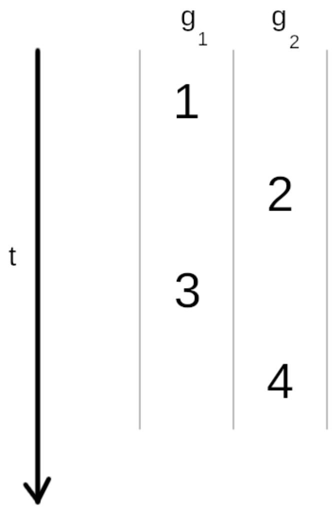
Flow Control



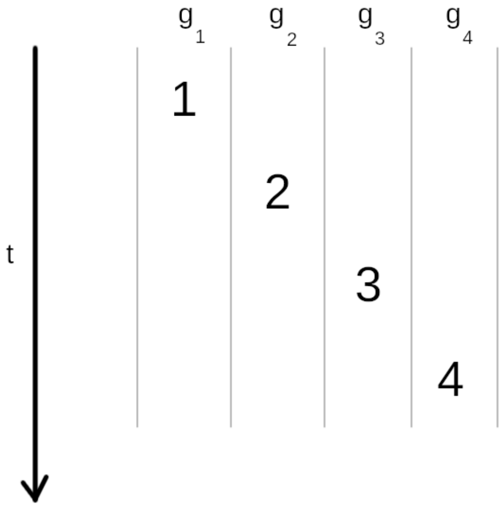
Flow Control



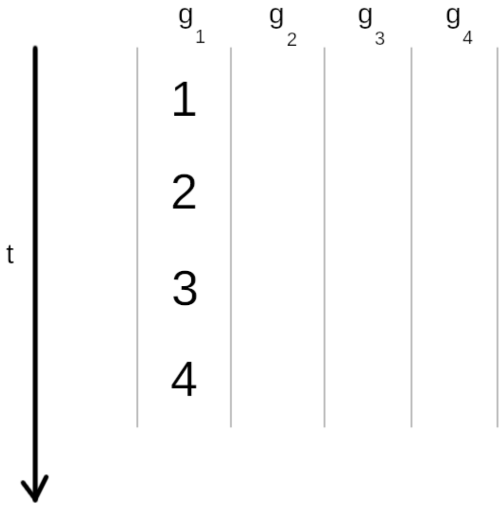
Flow Control



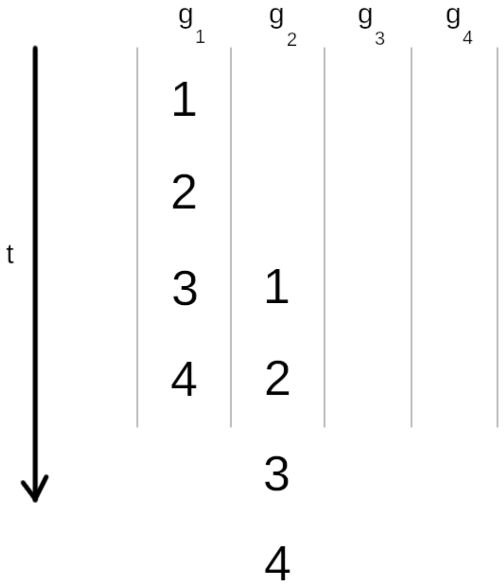
Flow Control



Reliability



Reliability



DNS

Anatomy of an IPv6 Multicast Address

Anatomy of an IPv6 Multicast Address

ff

Anatomy of an IPv6 Multicast Address

ff1

Anatomy of an IPv6 Multicast Address

ff1e

Anatomy of an IPv6 Multicast Address

ff1e: + group address (112 bits)

Multicast "DNS"

ff1e: + HASH("example.com")

Multicast "DNS"

example.com $\Longrightarrow$ ff1e:873e:378f:f6a5:a1f6:fa49:95f1:0faf

Librecast

Librecast

Librecast

- Developers Developers Developers

Librecast

- Developers Developers Developers
- Messaging Library

Librecast

- Developers Developers Developers
- Messaging Library
- Transitional Technology

Librecast

- Developers Developers Developers
- Messaging Library
- Transitional Technology
- Improved Routing Protocol

Librecast

- Developers Developers Developers
- Messaging Library
- Transitional Technology
- Improved Routing Protocol
- Build multicast-enabled applications

Librecast

- Developers Developers Developers
- Messaging Library
- Transitional Technology
- Improved Routing Protocol
- Build multicast-enabled applications
- Work with FOSS projects to enable multicast everywhere

Librecast

- Developers Developers Developers
- Messaging Library
- Transitional Technology
- Improved Routing Protocol
- Build multicast-enabled applications
- Work with FOSS projects to enable multicast everywhere
- Ensure new standards (eg. WebRTC, QUIC) support multicast

```
lc_ctx_t *ctx;
lc_socket_t *sock;
lc_channel_t *chan;
lc_message_t msg;

ctx = lc_ctx_new();
sock = lc_socket_new(ctx);
chan = lc_channel_new(ctx, channelName);
lc_channel_bind(sock, chan);

lc_msg_init_size(&msg, strlen(msgtext) - 1);
lc_msg_send(chan, &msg);

/* clean up */
lc_socket_close(sock);
lc_channel_free(chan);
lc_ctx_free(ctx);
```

Brett Sheffield — Librecast Project

http://brettsheffield.com — Email: brett@librecast.net
Freenode: bacs — Twitter: @brett_sheffield
github.com/brettsheffield — github.com/librestack