Compilation
	build: make
	clean: make clean

Execution Instructions
	sender: sender [file_to_send] [server_ip] [server_port]
	receiver: receiver [file_to_recv] [listening_port]
	agent: agent [drop_rate] [listening_port] [server_ip] [server_port]

What I Do
	A simplex rdt interface is provided to user programs. Both server and client call
	sp_init() to use this static library, sp_socket() to create sockets, and sp_close()
	to close sockets. The server calls sp_listen() and the client calls sp_connect to
	establish a connection. The client may call the non blocking sp_send() to add data
	into the send buffer if the buffer is not full, the blocking sp_send_all() to add
	exactly size bytes into the buffer, and the blocking sp_send_file() to send a file.
	The server may call the non blocking sp_recv() to fetch data from the receive buffer
	if the buffer is not empty, the blocking sp_recv_all() to fetch exactly size bytes
	from the buffer, and the blocking sp_recv_file() to receive a file.

	The sender and receiver can transport data without running the agent. If the agent is
	involved, it just acts like the sender of the receiver and the receiver of the sender.
	Both sender and receiver can still work without knowing any information about the
	agent. The probability that the agent drops a data packet is set to drop_rate/1000.

Issues And Solutions
	There is a send thread that sends packets in the send buffer. Mutex in pthread library
	is used to deal with the synchronization between the send thread and sp_send(). When
	I use a 10MB file to test the three program, the received file often differs a litte
	from the sent file. I just couldn't find the cause of this problem, but when I change
	the critical section in the send thread to cover more code, the problem is not appear
	anymore. (see the comments in bgsend() in sprdt.c)