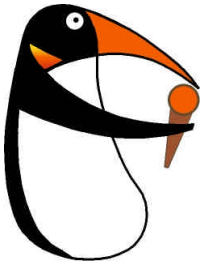
Comparing and Evaluating epoll, select and poll Event Mechanisms

Louay Gammo, Tim Brecht,
Amol Shukla, and David Pariag



University of Waterloo

NSERC CRSNG

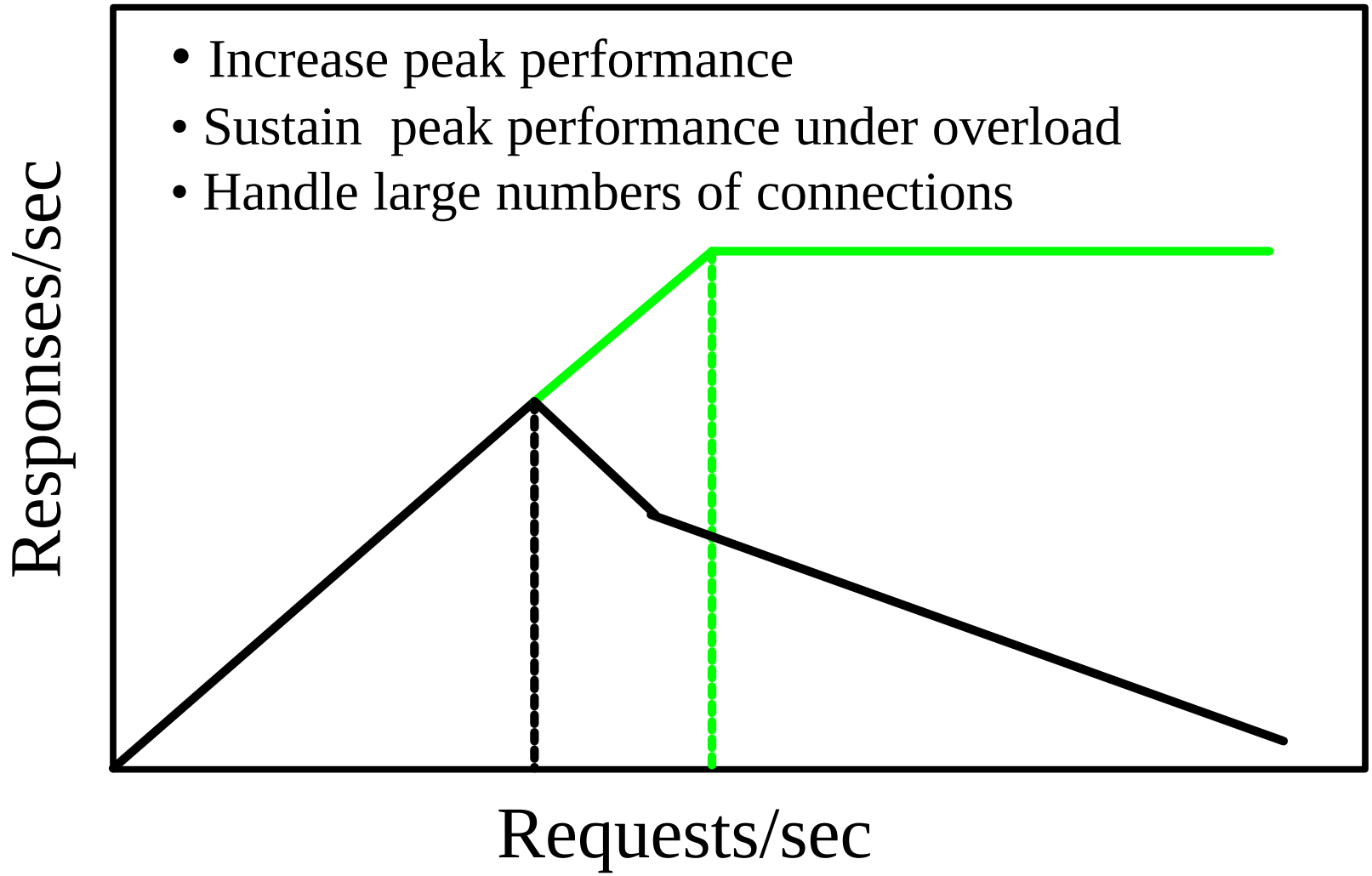
Ontario

hp invent

gelato.ORG

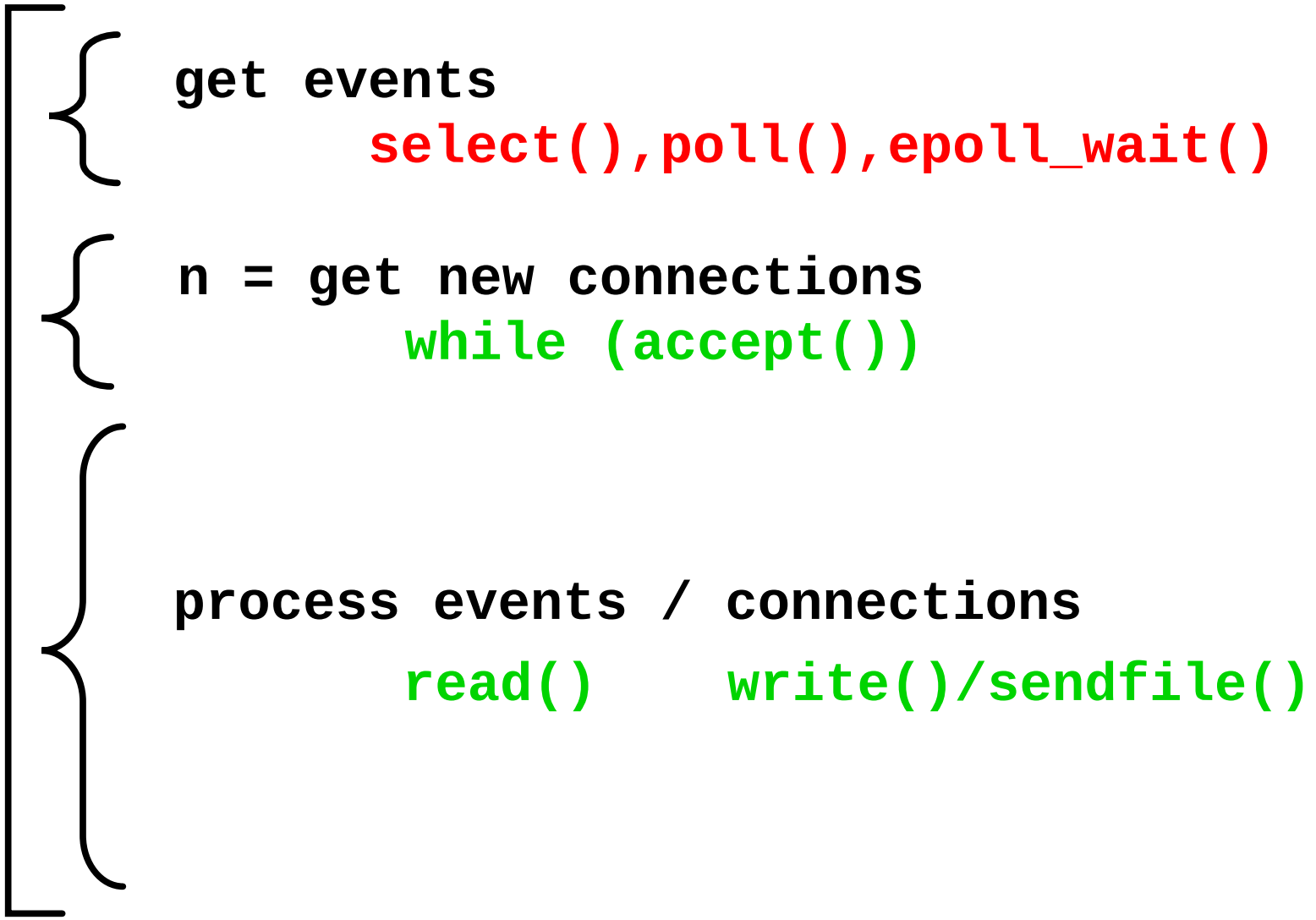
http://gelato.uwaterloo.ca

The Problem and Goals



Better understand apps and interaction with kernel

Server Phases



select and poll

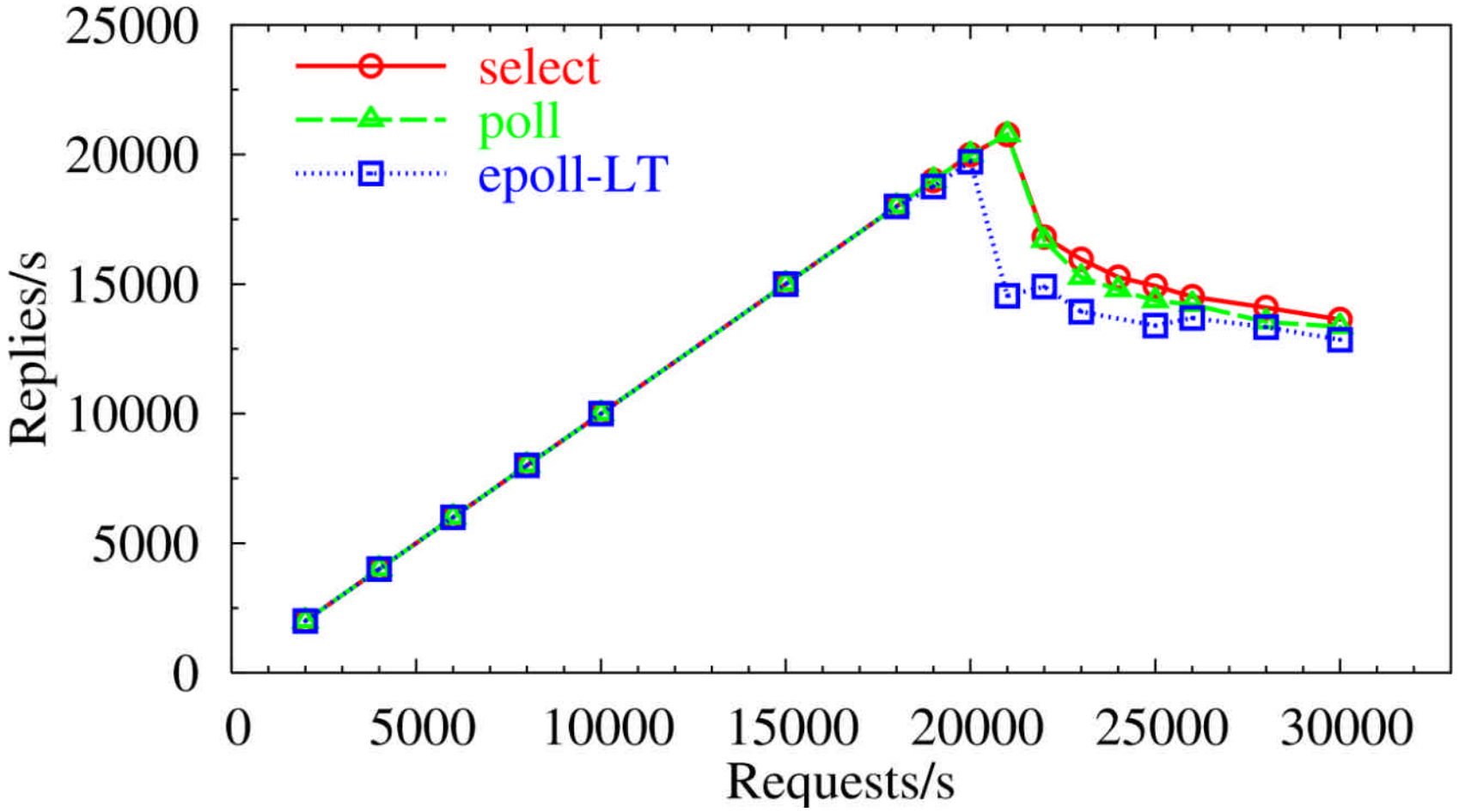
```
FD_SET(fd, &readable);
rdfds = readable; wrfds = writable;
n = select(max, rdfds, wrfds, exfds, &tout);
if (FD_ISSET(fd, rdfds)) { read }
FD_SET(fd, &writable);
FD_CLR(fd, &readable);
```

```
array[i].events = POLLIN;
n = poll(array, max, &tout);
if (array[i].revents & POLLIN) { read }
array[i].events = POLLOUT;
```

epoll

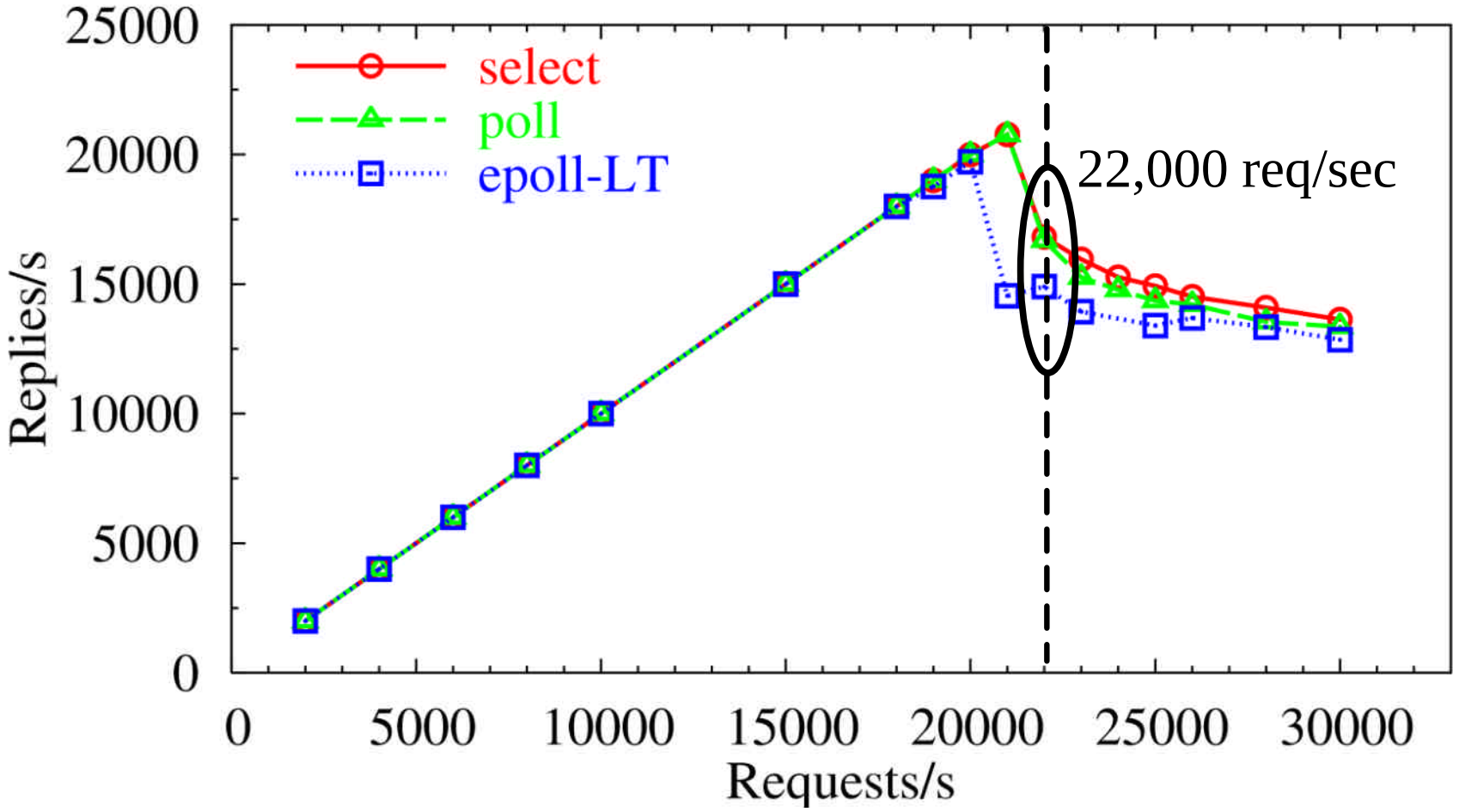
```
epfd = epoll_create(max_fds);

evt.data.fd = fd;
evt.events = EPOLLIN;
epoll_ctl(epfd, EPOLL_CTL_ADD, fd, &evt);

epoll_wait(epfd, results, max_fd, tout);

evt.data.fd = fd;
evt.events = EPOLLOUT;
epoll_ctl(epfd, EPOLL_CTL_MOD, fd, &evt);
```

Motivation



- Simple 1 byte file workload (hammer the event mechanism)

Motivation



- Simple 1 byte file workload (hammer the event mechanism)

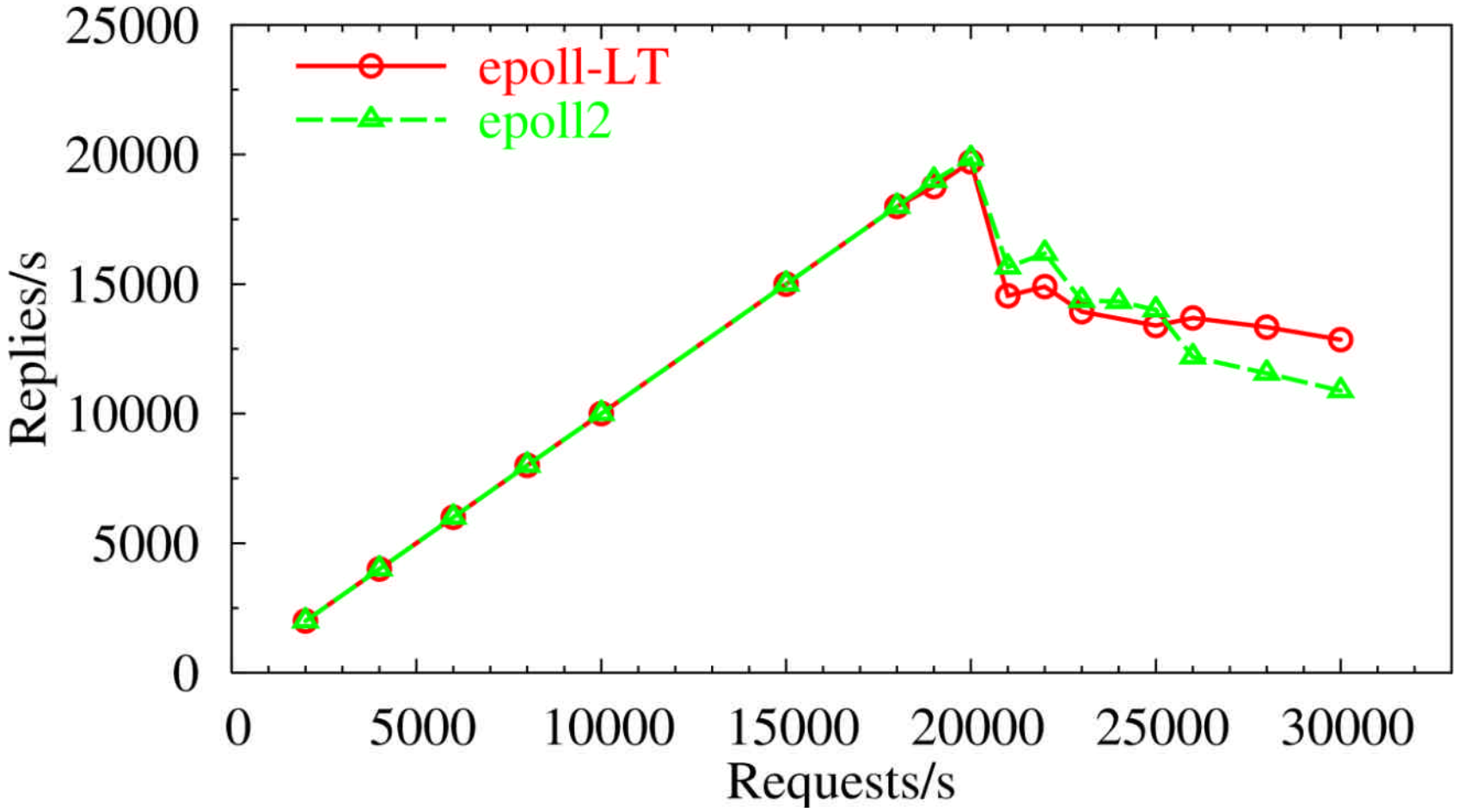
Motivation: gprof comparison

Syscall	select	epoll-LT	poll
read	21.51	20.95	20.97
close	14.90	14.05	14.79
select	13.33		
poll			13.32
epoll_wait		7.15	
epoll_clt		16.34	
setsockopt	11.17	9.13	10.68
accept	10.08	9.51	10.20
write	5.98	5.06	5.70
fcntl	3.66	3.34	3.61
sendfile	3.43	2.70	3.43
replies/sec	14,708	13,026	13,671

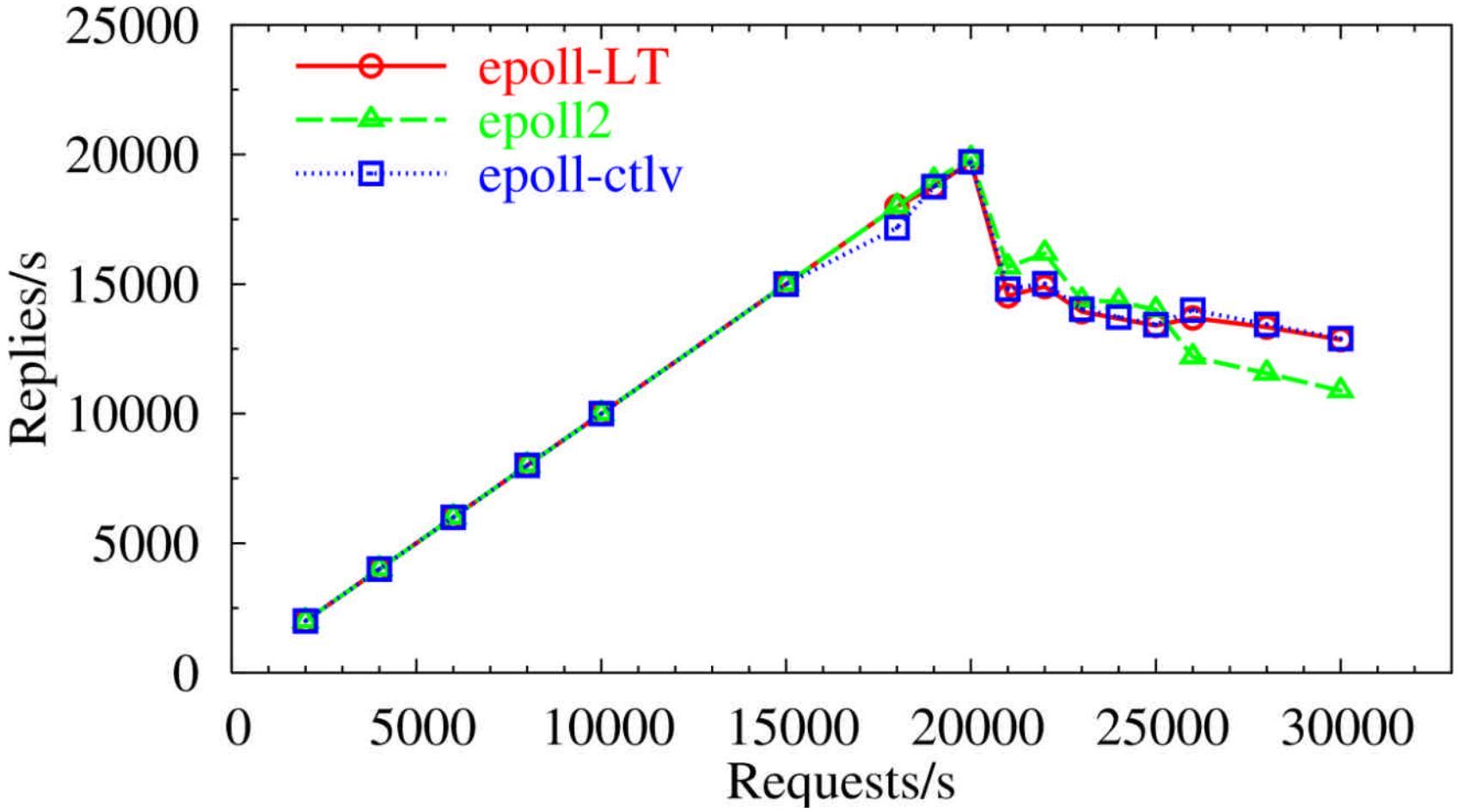

Reducing `epoll_ctl()` overhead

- Don’t modify interests **(epoll2)**
  - add/remove interest at time of accept and close
  - less **`epoll_ctl`** but more **`epoll_wait`**
- Aggregate System Calls **(epoll_ctlv)**
  - system call -- uses an array of fds/interest sets
  - one system call, many changes (ala readv/writev)
- Edge-triggered **(epoll-ET)**
  - only get events when there is a change on the fd
  - requires tracking state of the fd in application

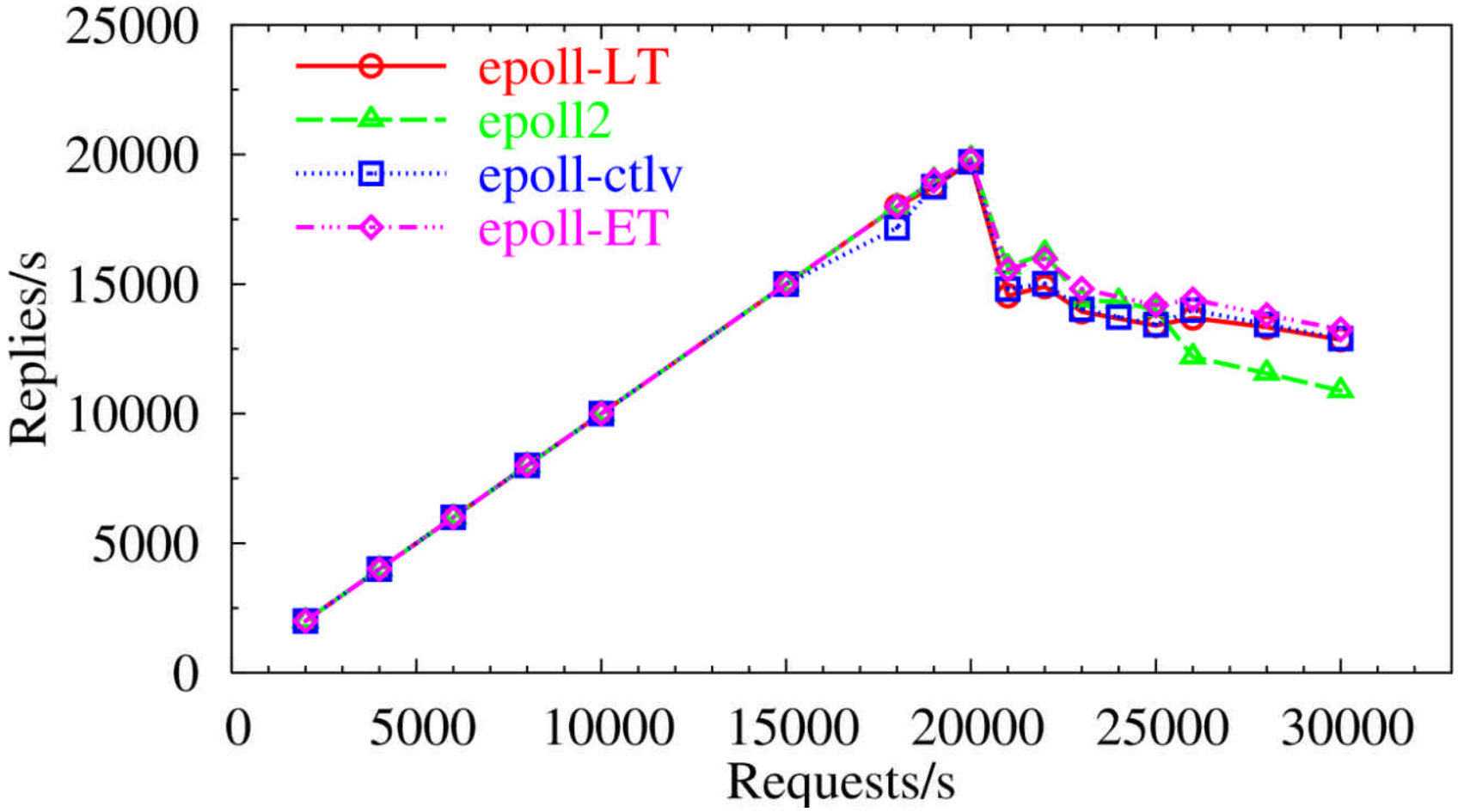
Reducing `epoll_ctl` calls



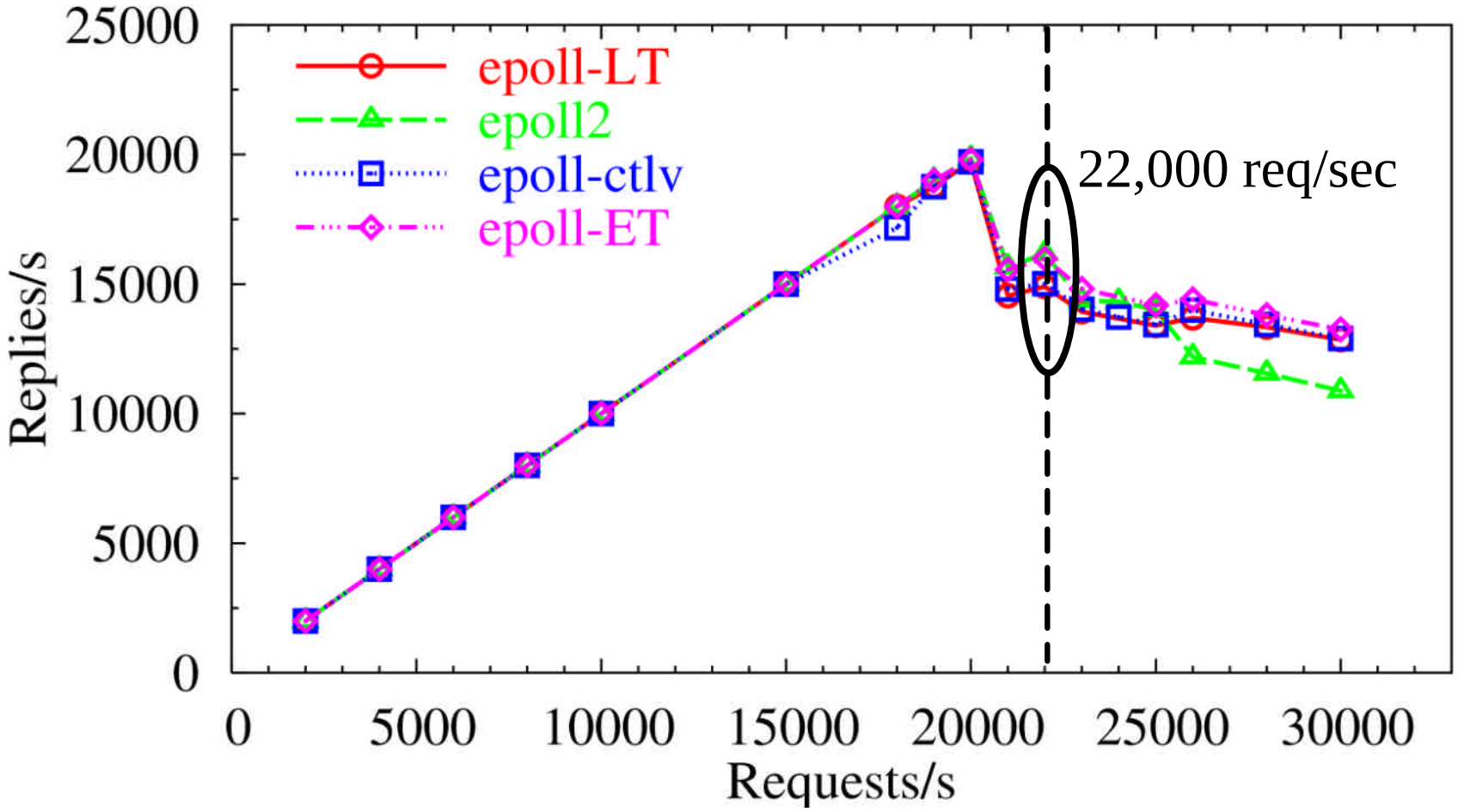
Reducing `epoll_ctlv` calls



Reducing `epoll_ctlv` calls



Reducing `epoll_ctlv` calls



gprof results @ 22,000 req/s

Syscall	epoll	epoll2	ctlv	edge
read	20.95	20.08	21.41	22.19
close	14.05	13.02	14.90	14.14
epoll_wait	7.15	12.56	6.01	6.52
epoll_ctl	16.34	10.27	5.98	11.06
epoll_ctlv			9.28	
sockopt	9.13	7.57	9.13	9.08
accept	9.51	9.05	9.76	9.30
write	5.06	4.13	5.10	5.31
fcntl	3.34	3.14	3.37	3.34
sendfile	2.70	3.00	2.71	3.91
replies/sec	13,026	13,598	13,343	13,665

gprof results @ 22,000 req/s

Syscall	epoll	epoll2	ctlv	edge
read	20.95	20.08	21.41	22.19
close	14.05	13.02	14.90	14.14
epoll_wait	7.15	12.56	6.01	6.52
epoll_ctl	16.34	10.27	5.98	11.06
epoll_ctlv			9.28	
sockopt	9.13	7.57	9.13	9.08
accept	9.51	9.05	9.76	9.30
write	5.06	4.13	5.10	5.31
fcntl	3.34	3.14	3.37	3.34
sendfile	2.70	3.00	2.71	3.91
replies/sec	13,026	13,598	13,343	13,665

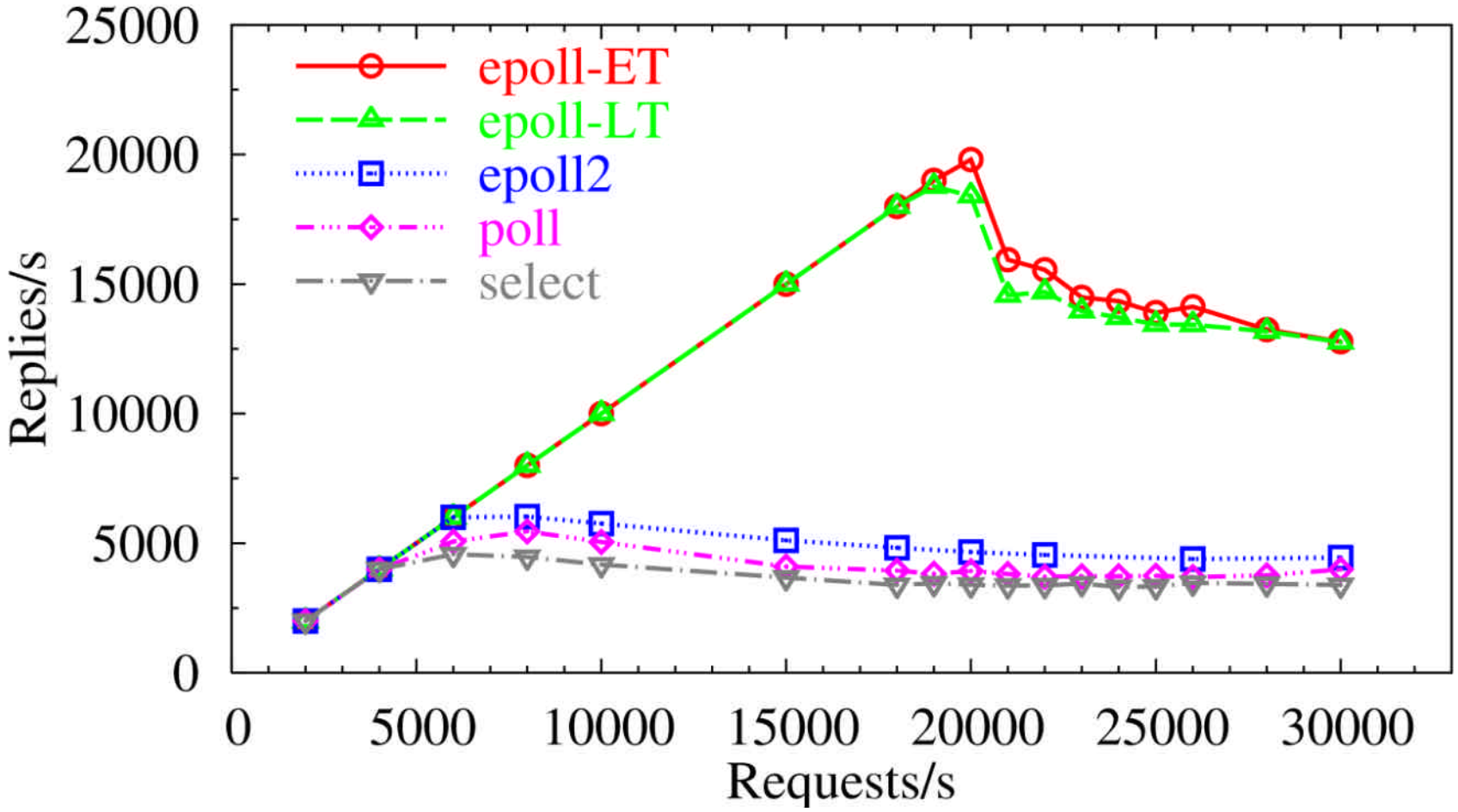
gprof results @ 22,000 req/s

Syscall	epoll	epoll2	ctlv	edge
read	20.95	20.08	21.41	22.19
close	14.05	13.02	14.90	14.14
epoll_wait	7.15	12.56	6.01	6.52
epoll_ctl	16.34	10.27	5.98	11.06
epoll_ctlv			9.28	
sockopt	9.13	7.57	9.13	9.08
accept	9.51	9.05	9.76	9.30
write	5.06	4.13	5.10	5.31
fcntl	3.34	3.14	3.37	3.34
sendfile	2.70	3.00	2.71	3.91
replies/sec	13,026	13,598	13,343	13,665

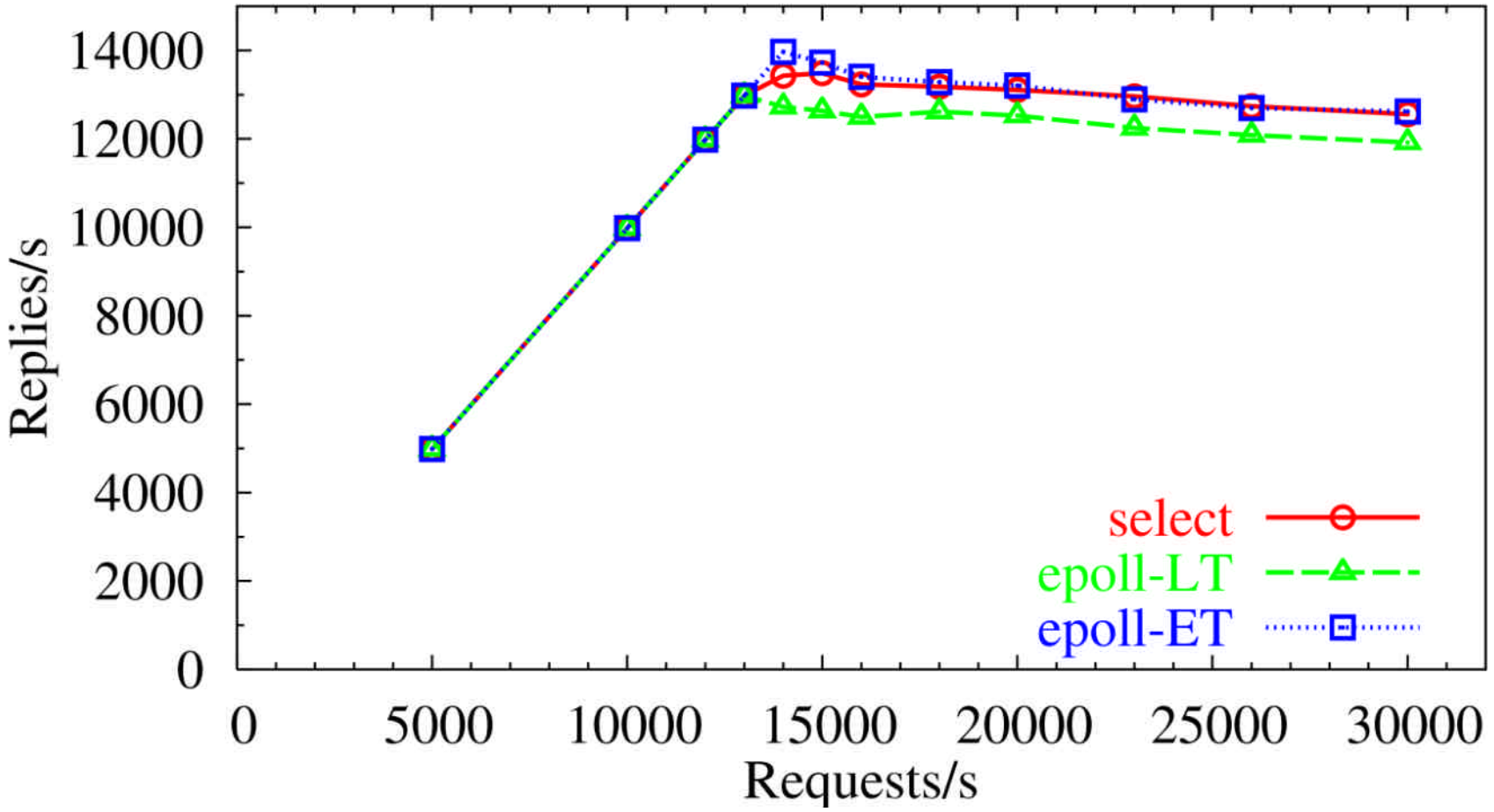
gprof results @ 22,000 req/s

Syscall	epoll	epoll2	ctlv	edge
read	20.95	20.08	21.41	22.19
close	14.05	13.02	14.90	14.14
epoll_wait	7.15	12.56	6.01	6.52
epoll_ctl	16.34	10.27	5.98	11.06
epoll_ctlv			9.28	
sockopt	9.13	7.57	9.13	9.08
accept	9.51	9.05	9.76	9.30
write	5.06	4.13	5.10	5.31
fcntl	3.34	3.14	3.37	3.34
sendfile	2.70	3.00	2.71	3.91
replies/sec	13,026	13,598	13,343	13,665

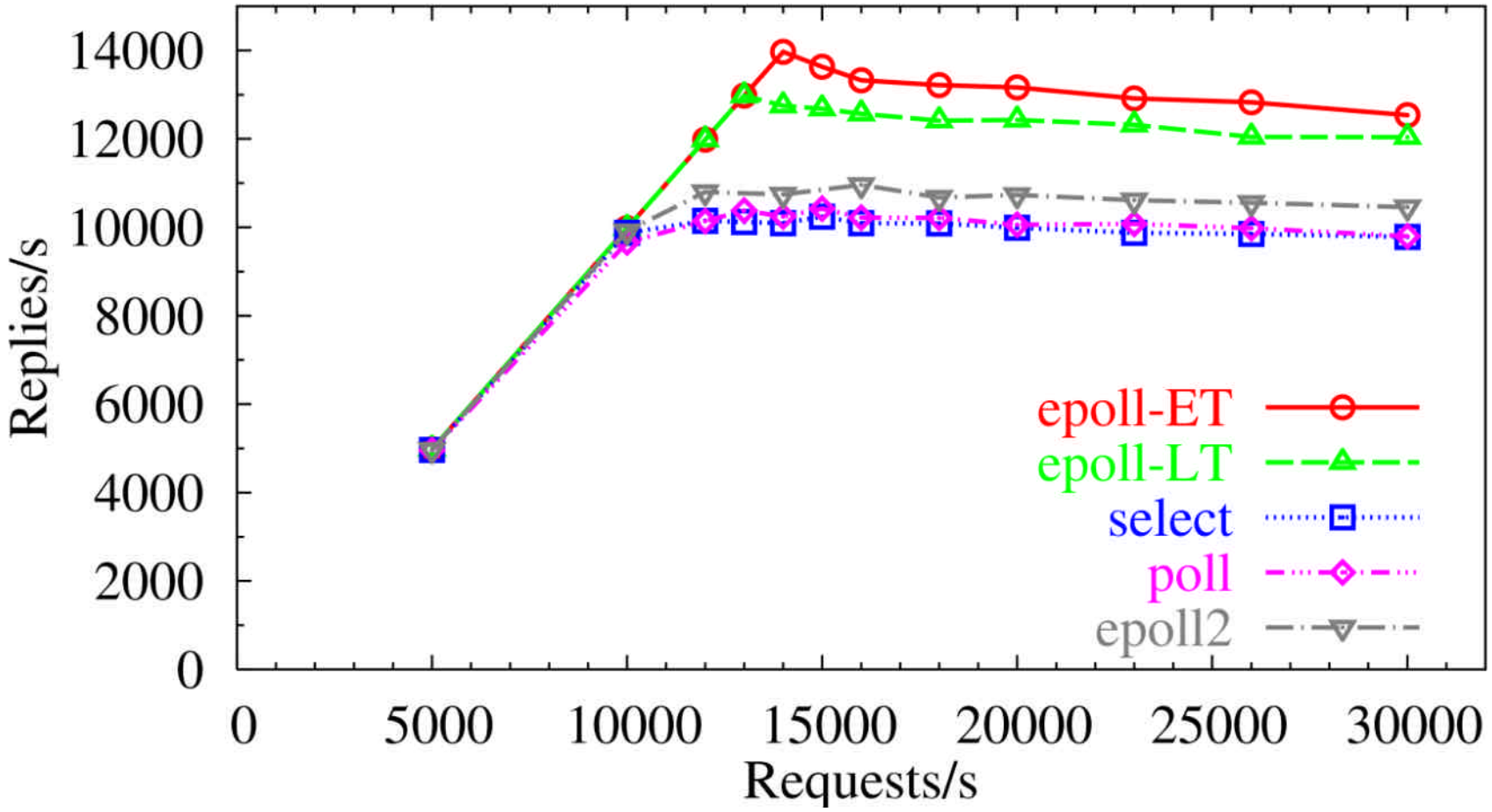
1 Byte File: 10,000 idle conns



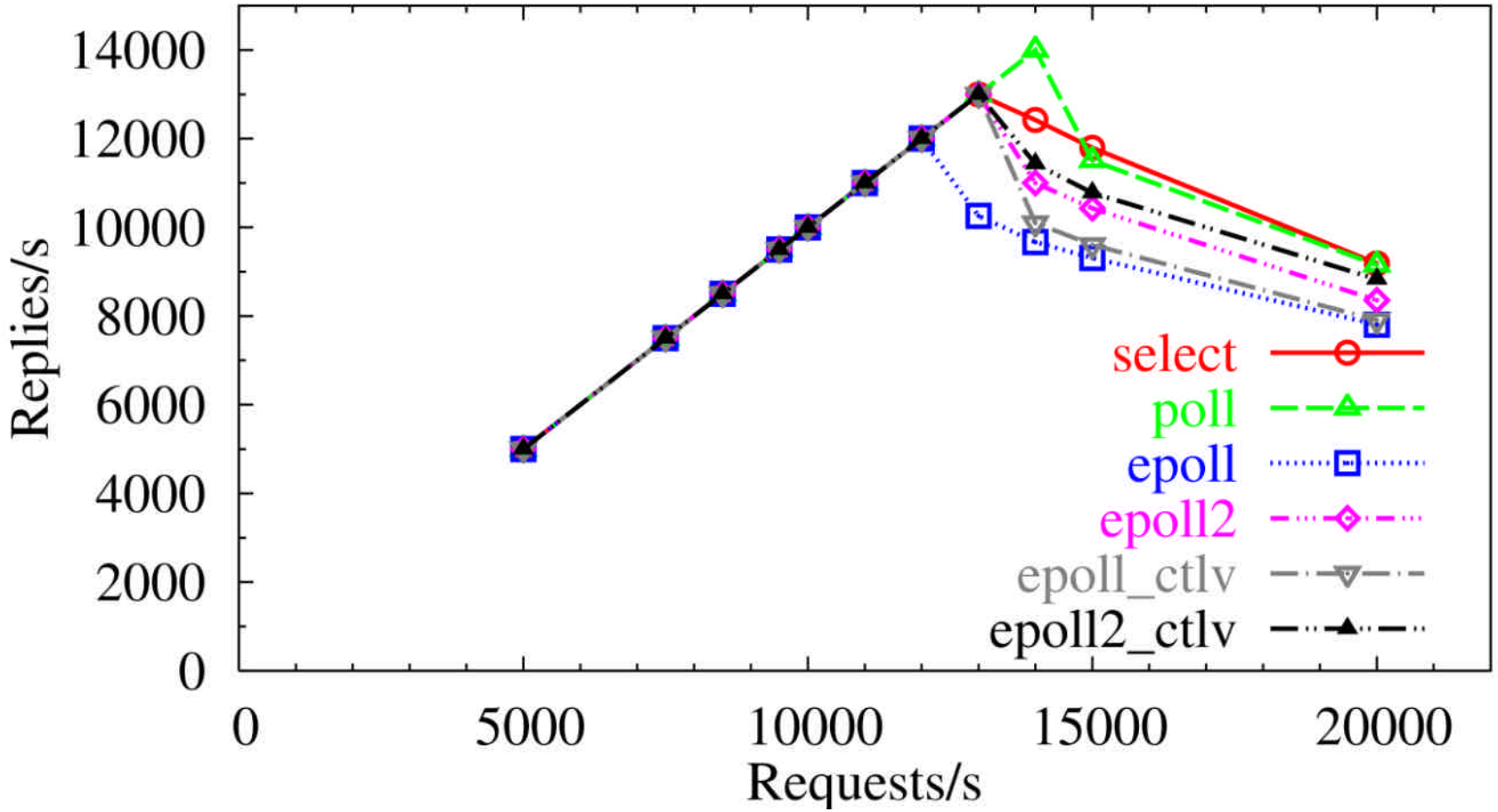
SPECweb99-like Workload



SPECweb99-like: 10,000 idle conns



Itanium Results: 1 byte file



Warning: appear to be problems with poll/epoll on IA64
(we are working with people to resolve them)

Discussion

- Reducing **`epoll_ctl`** costs did not improve tput
  - **`epoll2`** does quite well despite extra costs
  - **`epoll-LT/ET`** quite similar performance
- **`select`** and **`poll`** do well on some workloads
  - multiple accepts reduce event dispatch overheads

- Need better understanding
  - Once kernel bug is fixed use perf tools on IA64
- How to better represent Internet/WAN wloads
  - idle connections not very realistic ???