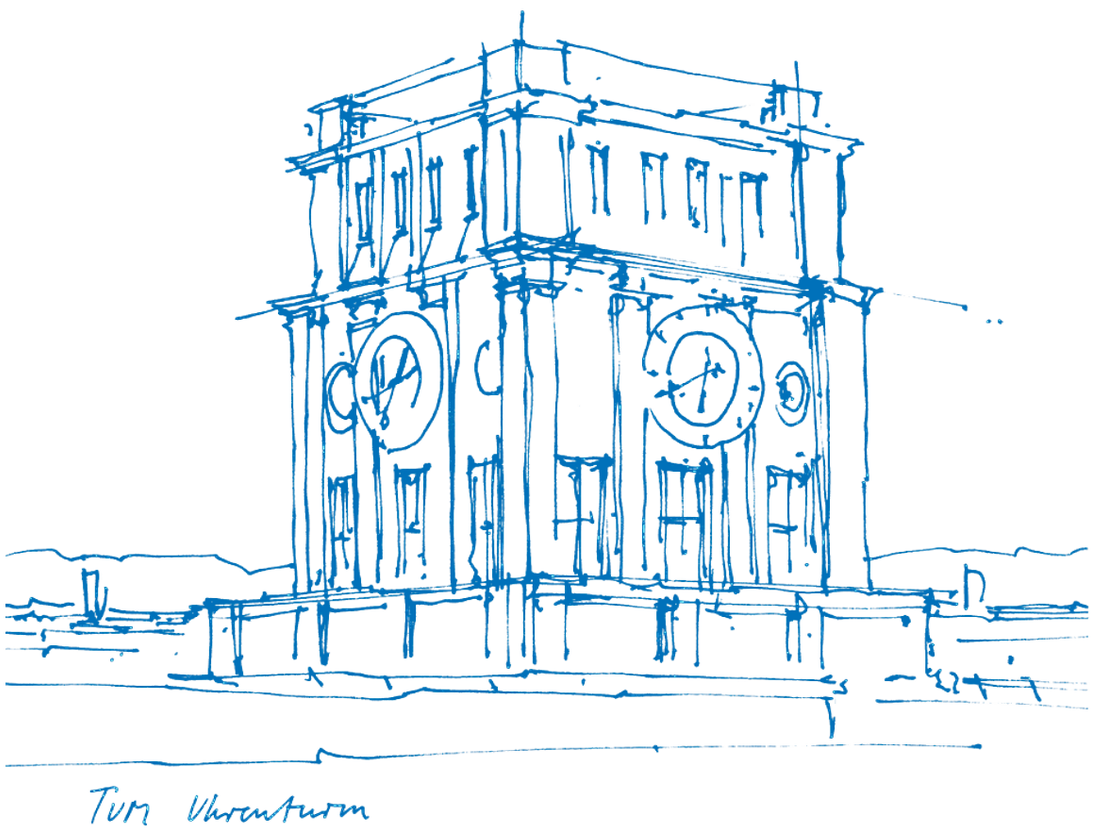
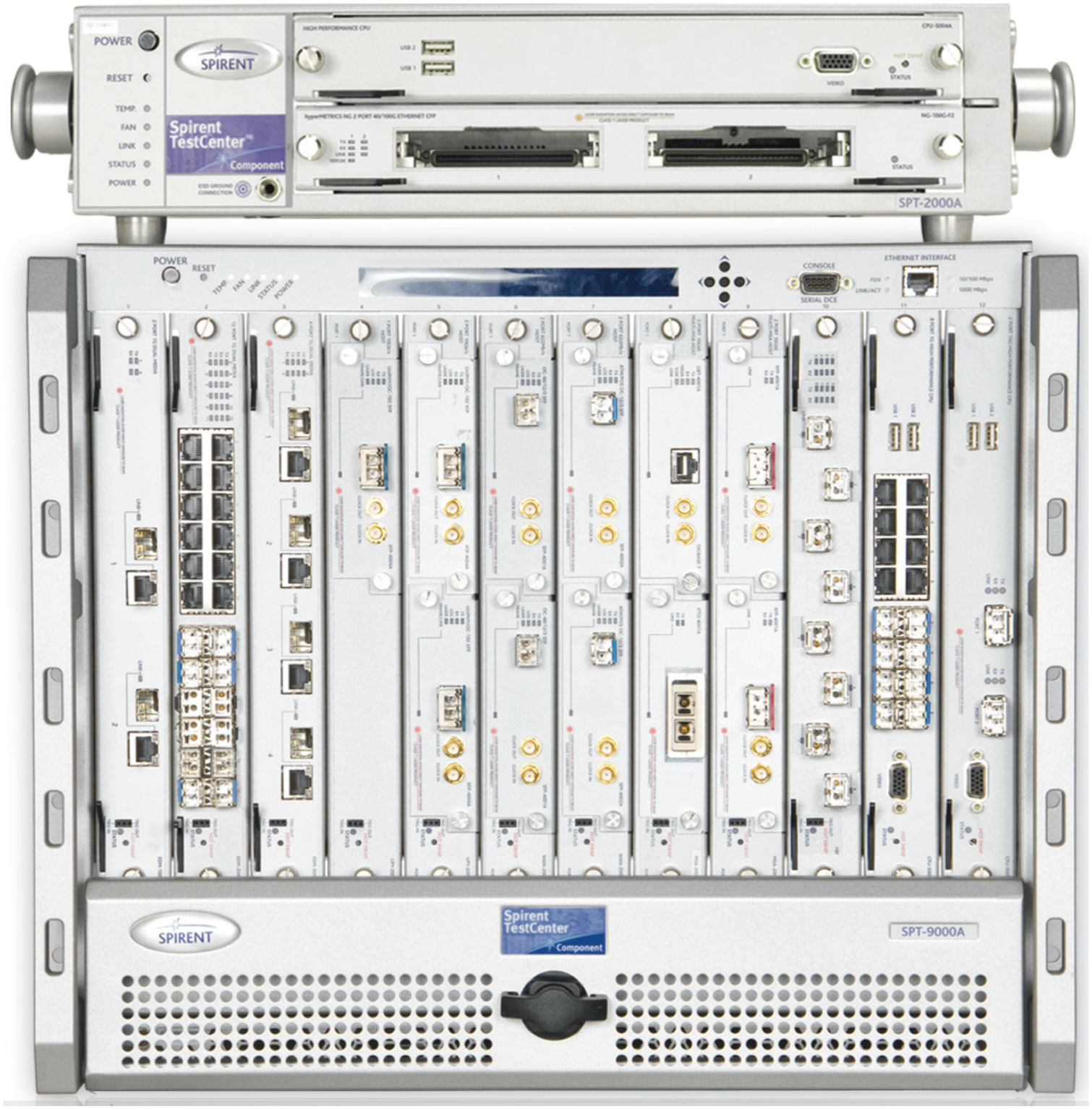
The MoonGen Packet Generator

Paul Emmerich
emmericp@net.in.tum.de
Technical University of Munich
Chair of Network Architectures and Services
DPDK Summit, 21.10.2016



Packet generators



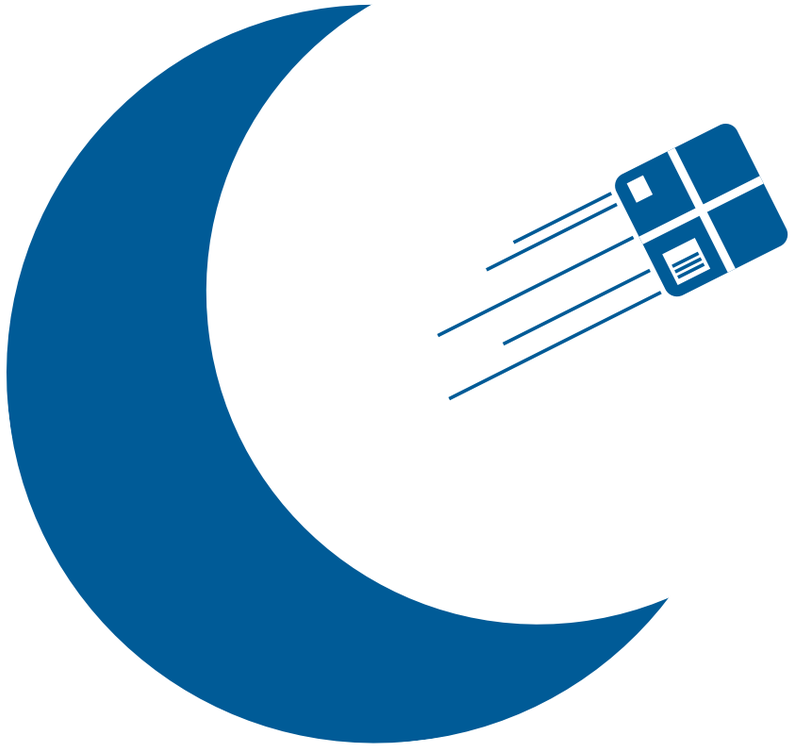
Source: www.spirent.com

Commodity hardware

Source: www.intel.com

MoonGen - A fast software packet generator

Combines the advantages of software (cheap, flexible) and hardware (precise, accurate) packet generators.

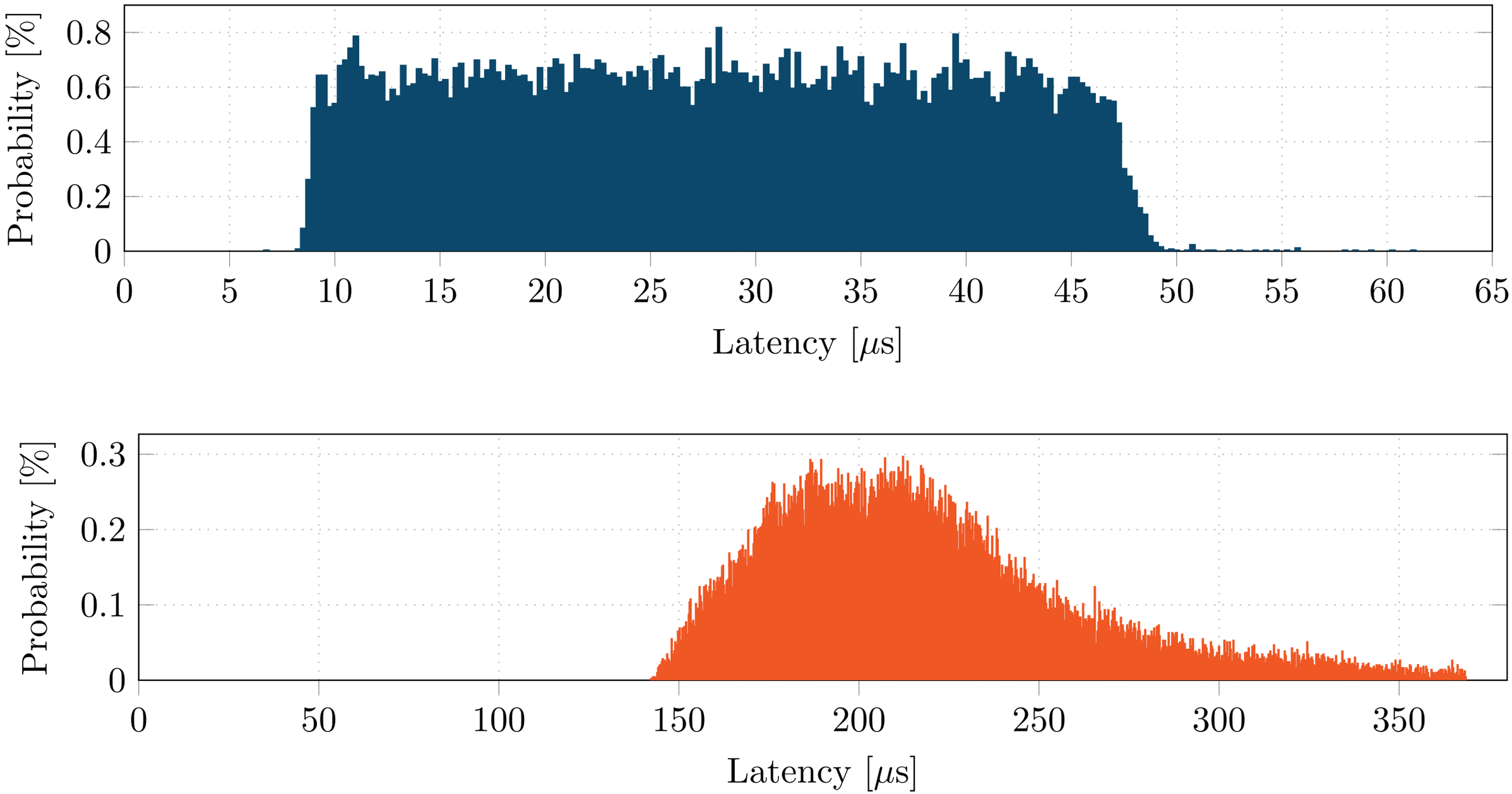


- *Fast*: DPDK for packet I/O, explicit multi-core support
- *Flexible*: Craft all packets in user-controlled Lua scripts
- *Timestamping*: Hardware features found on NICs
- *Rate control*: Hardware features and novel software approach
- *Free and open source*: Code available on GitHub

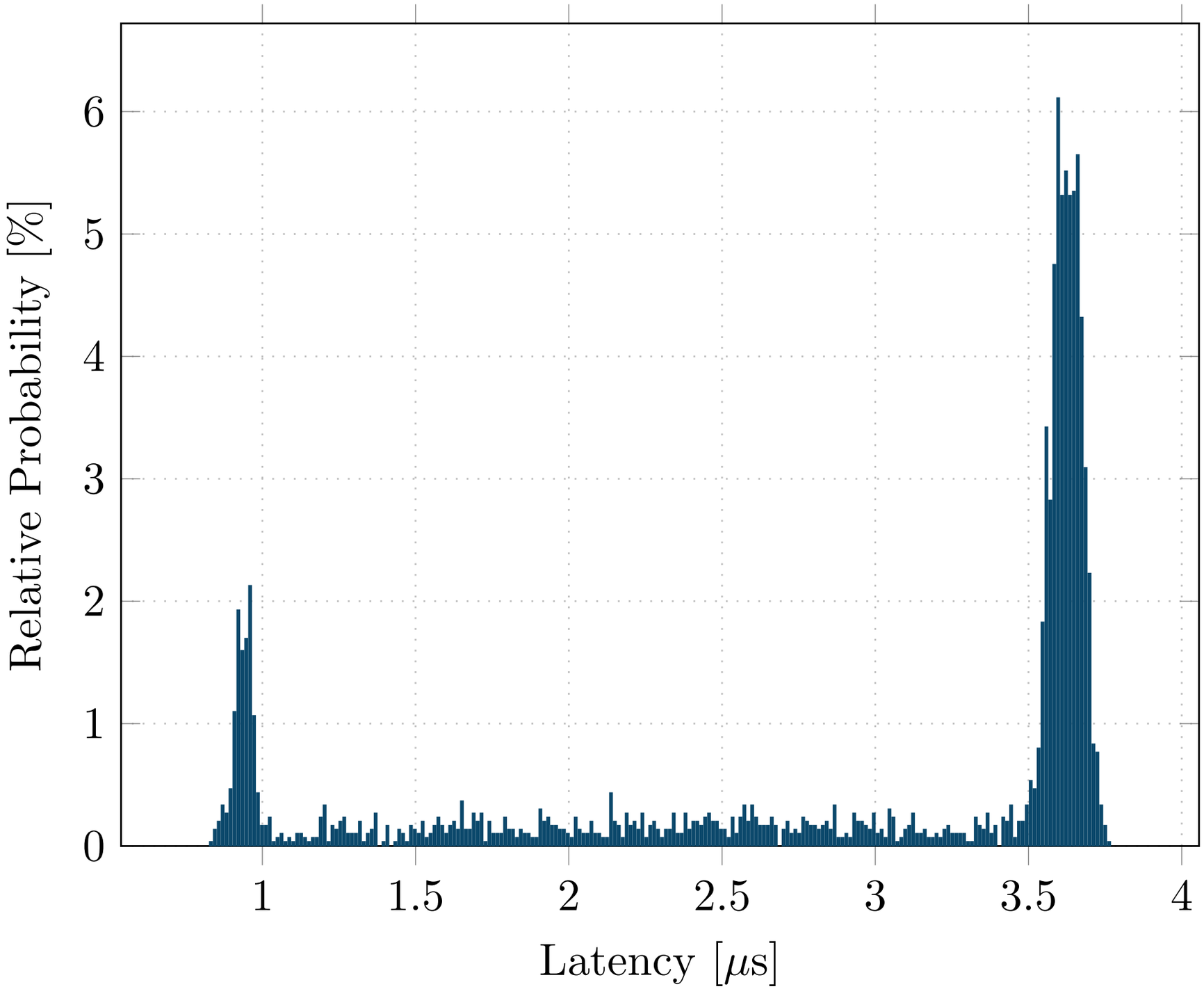
https://github.com/emmericp/MoonGen

Paul Emmerich, Sebastian Gallenmüller, Daniel Raumer, Florian Wohlfart, and Georg Carle.
MoonGen: A Scriptable High-Speed Packet Generator. *Internet Measurement Conference (IMC) 2015*, October 2015.

Latency measurements



Latency measurements



Generating complex packets

- Arbitrarily complex header stacks
- Generates and JIT compiles C structs
- Defaults for all header fields
  - E.g., calculates lengths, ports based on upper protocol
- Getters and setters, automatic endianness handling
- Following example from
  https://github.com/emmericp/moongen-scripts/blob/master/vxlan.lua

```
local vxlanStack = packetCreate(
  "eth", "ip4", "udp", "vxlan",
  {"eth_8021q", "innerEth"},
  {"ip4", "innerIp4"},
  {"udp", "innerUdp"}
)
```

Generating complex packets

- Create a mempool with a packet archetype

```
local mempool = memory.createMemPool(function(buf)
   local pkt = vxlanStack(buf)
   pkt:fill{
       -- fields not explicitly set here are initialized to defaults
       ethSrc = queue, -- MAC of the tx device
       ethDst = "AB:CD:EF:AB:CD:EF",
       ip4Src = "10.0.0.2",
       ip4Dst = "10.0.0.3",
       vxlanVNI = 10100,
       -- outer UDP ports are set automatically by the VXLAN handler
       innerEthSrc  = "12:34:56:78:90:ab",
       innerEthDst  = eth.BROADCAST,
       innerEthVlan = 100,
       innerIp4Src  = "192.168.0.1",
       innerIp4Dst  = "255.255.255.255",
       innerUdpSrc  = 1024,
       innerUdpDst  = 1024,
       pktLength    = 128
   }
   pkt.innerIp4:calculateChecksum()
end)
```

Generating complex packets

- Write a transmit loop

```
local bufs = mempool:bufArray()
while mg.running() do
    bufs:alloc(128)
    for i, buf in ipairs(bufs) do
        local pkt = vxlanStack(buf)
        pkt.innerUdp:setDstPort(
            1000 + math.random(0, 1000)
        )
        -- randomize other fields here
    end
    bufs:offloadUdpChecksums()
    queue:send(bufs)
end
```

Check out MoonGen on GitHub

MoonGen comes with a lot of examples
See if one fits your use case

https://github.com/emmericp/MoonGen

Questions?