net2o: vapor $\rightarrow$ reality

Bernd Paysan

EuroForth 2010, Hamburg

Outline

Looking for a challange

- As presented last year on EuroForth, the challange I'm looking at is a clean slate reimplementation of "the Internet"
- My previous company managed to sell me with my team instead of firing us—so the planned sabbatical doesn't happen
- This means it will take more time, but on the other hand it has to be simpler and more compact
- This talk is partly status report and much more a list of things to do
- IETF discussions about strategic internet development indicate that I'm on the right track

Looking for a challange

- As presented last year on EuroForth, the challange I'm looking at is a clean slate reimplementation of "the Internet"
- My previous company managed to sell me with my team instead of firing us—so the planned sabbatical doesn't happen
- This means it will take more time, but on the other hand it has to be simpler and more compact
- This talk is partly status report and much more a list of things to do
- IETF discussions about strategic internet development indicate that I'm on the right track

Looking for a challange

- As presented last year on EuroForth, the challange I'm looking at is a clean slate reimplementation of "the Internet"
- My previous company managed to sell me with my team instead of firing us—so the planned sabbatical doesn't happen
- This means it will take more time, but on the other hand it has to be simpler and more compact
- This talk is partly status report and much more a list of things to do
- IETF discussions about strategic internet development indicate that I'm on the right track

Looking for a challange

- As presented last year on EuroForth, the challange I'm looking at is a clean slate reimplementation of "the Internet"
- My previous company managed to sell me with my team instead of firing us—so the planned sabbatical doesn't happen
- This means it will take more time, but on the other hand it has to be simpler and more compact
- This talk is partly status report and much more a list of things to do
- IETF discussions about strategic internet development indicate that I'm on the right track

Looking for a challange

- As presented last year on EuroForth, the challange I'm looking at is a clean slate reimplementation of "the Internet"
- My previous company managed to sell me with my team instead of firing us—so the planned sabbatical doesn't happen
- This means it will take more time, but on the other hand it has to be simpler and more compact
- This talk is partly status report and much more a list of things to do
- IETF discussions about strategic internet development indicate that I'm on the right track

Recap: Requirements

Scalability Must work well with low and high bandwidths, loose and tightly coupled systems, few and many hosts connected together over short to far distances.

Easy to implement Must work with a minimum of effort, must allow small and cheap devices to connect. One idea is to replace "busses" like USB and firewire with cheap LAN links.

Security Users want authentication and authorization, but also anonymity and privacy. Firewalls and similar gatekeepers (load balancers, etc.) are common.

Media capable This requires real-time capabilities, pre-allocated bandwidth and other QoS features, end-to-end.

Transparency Must be able to work together with other networks (especially Internet 1.0).

Recap: Requirements

Scalability Must work well with low and high bandwidths, loose and tightly coupled systems, few and many hosts connected together over short to far distances.

Easy to implement Must work with a minimum of effort, must allow small and cheap devices to connect. One idea is to replace "busses" like USB and firewire with cheap LAN links.

Security Users want authentication and authorization, but also anonymity and privacy. Firewalls and similar gatekeepers (load balancers, etc.) are common.

Media capable This requires real-time capabilities, pre-allocated bandwidth and other QoS features, end-to-end.

Transparency Must be able to work together with other networks (especially Internet 1.0).

Recap: Requirements

Scalability Must work well with low and high bandwidths, loose and tightly coupled systems, few and many hosts connected together over short to far distances.

Easy to implement Must work with a minimum of effort, must allow small and cheap devices to connect. One idea is to replace "busses" like USB and firewire with cheap LAN links.

Security Users want authentication and authorization, but also anonymity and privacy. Firewalls and similar gatekeepers (load balancers, etc.) are common.

Media capable This requires real-time capabilities, pre-allocated bandwidth and other QoS features, end-to-end.

Transparency Must be able to work together with other networks (especially Internet 1.0).

Recap: Requirements

Scalability Must work well with low and high bandwidths, loose and tightly coupled systems, few and many hosts connected together over short to far distances.

Easy to implement Must work with a minimum of effort, must allow small and cheap devices to connect. One idea is to replace "busses" like USB and firewire with cheap LAN links.

Security Users want authentication and authorization, but also anonymity and privacy. Firewalls and similar gatekeepers (load balancers, etc.) are common.

Media capable This requires real-time capabilities, pre-allocated bandwidth and other QoS features, end-to-end.

Transparency Must be able to work together with other networks (especially Internet 1.0).

Recap: Requirements

Scalability Must work well with low and high bandwidths, loose and tightly coupled systems, few and many hosts connected together over short to far distances.

Easy to implement Must work with a minimum of effort, must allow small and cheap devices to connect. One idea is to replace "busses" like USB and firewire with cheap LAN links.

Security Users want authentication and authorization, but also anonymity and privacy. Firewalls and similar gatekeepers (load balancers, etc.) are common.

Media capable This requires real-time capabilities, pre-allocated bandwidth and other QoS features, end-to-end.

Transparency Must be able to work together with other networks (especially Internet 1.0).

Switching Packets, Routing Connections

- Similar to MPLS, packets should run through a switching network, not through routers
- Routing is a combination of DNS (name resolution) and routing calculation (destination lookup)

Physical Route

- Take first n bits of target address and select destination
- Shift target address by n
- Insert bit-reversed source into address field

Switching Packets, Routing Connections

- Similar to MPLS, packets should run through a switching network, not through routers
- Routing is a combination of DNS (name resolution) and routing calculation (destination lookup)

Physical Route

- Take first n bits of target address and select destination
- Shift target address by n
- Insert bit-reversed source into address field

Switching Packets, Routing Connections

- Similar to MPLS, packets should run through a switching network, not through routers
- Routing is a combination of DNS (name resolution) and routing calculation (destination lookup)

Physical Route

- Take first n bits of target address and select destination
- Shift target address by n
- Insert bit-reversed source into address field

Switching Packets, Routing Connections

- Similar to MPLS, packets should run through a switching network, not through routers
- Routing is a combination of DNS (name resolution) and routing calculation (destination lookup)

Physical Route

- Take first n bits of target address and select destination
- Shift target address by n
- Insert bit-reversed source into address field

Switching Packets, Routing Connections

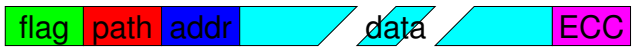
- Similar to MPLS, packets should run through a switching network, not through routers
- Routing is a combination of DNS (name resolution) and routing calculation (destination lookup)

Physical Route

- Take first n bits of target address and select destination
- Shift target address by n
- Insert bit-reversed source into address field

Recap: Packet Header

	Size
Flags	*2*
Path	*2/8*
Address	*2/8*
Junk	*0/8*
Data	*32/128/512/2k*
ECC	*L1 dependent*



Starting Point

- As starting point, I first implement net2o using UDP as transport layer
- UDP offers a reaonable interface for a single server that handles many connections without crazy Unix overhead
- For start, IPv4 only; IPv6 requires more work (no fragmented packets possible)
- Two parts: Packet server and command generator/interpreter

Starting Point

- As starting point, I first implement net2o using UDP as transport layer
- UDP offers a reaonable interface for a single server that handles many connections without crazy Unix overhead
- For start, IPv4 only; IPv6 requires more work (no fragmented packets possible)
- Two parts: Packet server and command generator/interpreter

Starting Point

- As starting point, I first implement net2o using UDP as transport layer
- UDP offers a reaonable interface for a single server that handles many connections without crazy Unix overhead
- For start, IPv4 only; IPv6 requires more work (no fragmented packets possible)
- Two parts: Packet server and command generator/interpreter

Starting Point

- As starting point, I first implement net2o using UDP as transport layer
- UDP offers a reaonable interface for a single server that handles many connections without crazy Unix overhead
- For start, IPv4 only; IPv6 requires more work (no fragmented packets possible)
- Two parts: Packet server and command generator/interpreter

Switching

- Use a hash for "switching" IP-Addresses: Hash value equals prefix
- Hash collissions resolved with longer prefixes
- Prefix granularity: Byte

  MSB=0 Direct routing entry
  MSB=1 larger prefix, look at next byte for more data

Switching

- Use a hash for "switching" IP-Addresses: Hash value equals prefix
- Hash collissions resolved with longer prefixes
- Prefix granularity: Byte

  MSB=0 Direct routing entry
  MSB=1 larger prefix, look at next byte for more data

Switching

- Use a hash for "switching" IP-Addresses: Hash value equals prefix
- Hash collissions resolved with longer prefixes
- Prefix granularity: Byte

  MSB=0 Direct routing entry
  MSB=1 larger prefix, look at next byte for more data

Switching

- Use a hash for "switching" IP-Addresses: Hash value equals prefix
- Hash collissions resolved with longer prefixes
- Prefix granularity: Byte

  MSB=0 Direct routing entry
  MSB=1 larger prefix, look at next byte for more data

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Shared Memory

- Map from address to connection context
- Connection context (will) contain
  - real addresses
  - file handles
  - cryptographic keys
  - authentication information
  - and other status information (a lot of that still unimplemented)
- Event queue for received packets

Commands

- UTF-8 encoded commands: Simple commands are 0-7F, one byte, complexer commands take more bytes
- Commands packet into 8 byte chunks
- 8 byte literals (e.g. addresses) and strings embedded into the command structure
- Command assembler allows seamless commands within Forth code
- Discussion: offsets to literals as UTF-8 code?

Commands

- UTF-8 encoded commands: Simple commands are 0-7F, one byte, complexer commands take more bytes
- Commands packet into 8 byte chunks
- 8 byte literals (e.g. addresses) and strings embedded into the command structure
- Command assembler allows seamless commands within Forth code
- Discussion: offsets to literals as UTF-8 code?

Commands

- UTF-8 encoded commands: Simple commands are 0-7F, one byte, complexer commands take more bytes
- Commands packet into 8 byte chunks
- 8 byte literals (e.g. addresses) and strings embedded into the command structure
- Command assembler allows seamless commands within Forth code
- Discussion: offsets to literals as UTF-8 code?

Commands

- UTF-8 encoded commands: Simple commands are 0-7F, one byte, complexer commands take more bytes
- Commands packet into 8 byte chunks
- 8 byte literals (e.g. addresses) and strings embedded into the command structure
- Command assembler allows seamless commands within Forth code
- Discussion: offsets to literals as UTF-8 code?

Commands

- UTF-8 encoded commands: Simple commands are 0-7F, one byte, complexer commands take more bytes
- Commands packet into 8 byte chunks
- 8 byte literals (e.g. addresses) and strings embedded into the command structure
- Command assembler allows seamless commands within Forth code
- Discussion: offsets to literals as UTF-8 code?

Working Testcase

Server loop

```
init-server
server-loop
```

Debugging output

```
init-client
s" localhost" net2o-udp insert-ipv4
              constant lserver
net2o-code s" This is a test" $, type
              '!' char, emit cr end-code
cmdbuf cell+ 0 lserver sendA
```

Working Testcase

Server loop

```
init-server
server-loop
```

Debugging output

```
init-client
s" localhost" net2o-udp insert-ipv4
              constant lserver
net2o-code s" This is a test" $, type
              '!' char, emit cr end-code
cmdbuf cell+ 0 lserver sendA
```

Flow Control

- UDP offers no quality of service
- TCP/IP flow control is horribly broken, assumes no buffers—reality are buffers everywhere, filled up completely by TCP/IP (causing horribly lags)
- Idea: PLL-based flow control, tries to prevent buffers filling up
- "Fast start:" Send first few packets out as fast as possible, to measure actual data rate

Flow Control

- UDP offers no quality of service
- TCP/IP flow control is horribly broken, assumes no buffers—reality are buffers everywhere, filled up completely by TCP/IP (causing horribly lags)
- Idea: PLL-based flow control, tries to prevent buffers filling up
- "Fast start:" Send first few packets out as fast as possible, to measure actual data rate

Flow Control

- UDP offers no quality of service
- TCP/IP flow control is horribly broken, assumes no buffers—reality are buffers everywhere, filled up completely by TCP/IP (causing horribly lags)
- Idea: PLL-based flow control, tries to prevent buffers filling up
- "Fast start:" Send first few packets out as fast as possible, to measure actual data rate

Flow Control

- UDP offers no quality of service
- TCP/IP flow control is horribly broken, assumes no buffers—reality are buffers everywhere, filled up completely by TCP/IP (causing horribly lags)
- Idea: PLL-based flow control, tries to prevent buffers filling up
- "Fast start:" Send first few packets out as fast as possible, to measure actual data rate

Cryptography

- Ellyptic Curve Cryptography code for the assymmetric part (much faster than RSA, a lot stronger per bit)
- Wurstkessel as symmetric cryptography and hashes
- Ubiquituous encryption is very important for network neutrality!

Cryptography

- Ellyptic Curve Cryptography code for the assymmetric part (much faster than RSA, a lot stronger per bit)
- Wurstkessel as symmetric cryptography and hashes
- Ubiquituous encryption is very important for network neutrality!

Cryptography

- Ellyptic Curve Cryptography code for the assymmetric part (much faster than RSA, a lot stronger per bit)
- Wurstkessel as symmetric cryptography and hashes
- Ubiquituous encryption is very important for network neutrality!

Presentation/Browser

- Typesetting engine
- Embedding of images, audio, and video—but please no plugins!
- Properly secured scripting (needs to be simple enough for that!)

Presentation/Browser

- Typesetting engine
- Embedding of images, audio, and video—but please no plugins!
- Properly secured scripting (needs to be simple enough for that!)

Presentation/Browser

- Typesetting engine
- Embedding of images, audio, and video—but please no plugins!
- Properly secured scripting (needs to be simple enough for that!)

Summary

- There is already a little bit of code
- A lot more work for long dark winter evenings
- After completion of reference implementation: RFC, IETF discussions, presentations at larger network-related conferences

Summary

- There is already a little bit of code
- A lot more work for long dark winter evenings
- After completion of reference implementation: RFC, IETF discussions, presentations at larger network-related conferences

Summary

- There is already a little bit of code
- A lot more work for long dark winter evenings
- After completion of reference implementation: RFC, IETF discussions, presentations at larger network-related conferences

For Further Reading I

Bernd Paysan

Internet 2.0

http://www.jwdt.com/~paysan/internet-2.0.html