Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 26, 2014

S. Previdi, Ed.
C. Filsfils, Ed.
Cisco Systems, Inc.
B. Decraene
S. Litkowski
Orange
M. Horneffer
R. Geib
Deutsche Telekom
R. Shakir
British Telecom
R. Raszuk
Individual

April 24, 2014

SPRING Problem Statement and Requirements
draft-previdi-spring-problem-statement-04

Abstract

The ability for a node to specify a forwarding path, other than the normal shortest path, that a particular packet will traverse, benefits a number of network functions. Source-based routing mechanisms have previously been specified for network protocols, but have not seen widespread adoption. In this context, the term 'source' means 'the point at which the explicit route is imposed'.

This document outlines various use cases, with their requirements, that need to be taken into account by the Source Packet Routing in Networking (SPRING) architecture for unicast traffic. Multicast use-cases and requirements are out of scope of this document.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

The ability for a node to specify a unicast forwarding path, other than the normal shortest path, that a particular packet will traverse, benefits a number of network functions, for example:

- Some types of network virtualization, including multi-topology networks and the partitioning of network resources for VPNs
- Network, link, path and node protection such as fast re-route
- Network programmability
- OAM techniques
- Simplification and reduction of network signaling components
- Load balancing and traffic engineering

Source-based routing mechanisms have previously been specified for network protocols, but have not seen widespread adoption other than in MPLS traffic engineering.

These network functions may require greater flexibility and per packet source imposed routing than can be achieved through the use of the previously defined methods. In the context of this charter, 'source' means 'the point at which the explicit route is imposed'.

In this context, Source Packet Routing in Networking (SPRING) architecture is being defined in order to address the use cases and requirements described in this document.

SPRING architecture should allow incremental and selective deployment without any requirement of flag day or massive upgrade of all network elements.

SPRING architecture should allow optimal virtualization: put policy state in the packet header and not in the intermediate nodes along the path. Hence, the policy is completely virtualized away from midpoints and tail-ends.

SPRING architecture objective is not to replace existing source routing and traffic engineering mechanisms but rather complement them and address use cases where removal of signaling and path state in the core is a requirement.

2. Dataplanes

The SPRING architecture should be general in order to ease its applicability to different dataplanes.

MPLS dataplane doesn't require any modification in order to apply a source-based routed model (e.g.: [I-D.filsfils-spring-segment-routing-mpls]).

IPv6 specification [RFC2460], amended by [RFC6564] and [RFC7045], defines the Routing Extension Header which provides IPv6 source-based routing capabilities.

The SPRING architecture should leverage existing MPLS dataplane without any modification and leverage IPv6 dataplane with a new IPv6 Routing Header Type (IPv6 Routing Header is defined in [RFC2460]).

3. IGP-based MPLS Tunneling

The source-based routing model, applied to the MPLS dataplane, offers the ability to tunnel services (VPN, VPLS, VPWS) from an ingress PE to an egress PE, with or without the expression of an explicit path and without requiring forwarding plane or control plane state in intermediate nodes.

The source-based routing model, applied to the MPLS dataplane, offers the ability to tunnel unicast services (VPN, VPLS, VPWS) from an ingress PE to an egress PE, with or without the expression of an explicit path and without requiring forwarding plane or control plane state in intermediate nodes. p2mp and mp2mp tunnels are out of the scope of this document.

3.1. Example of IGP-based MPLS Tunnels

This section illustrates an example use-case taken from [I-D.filsfils-spring-segment-routing-use-cases].

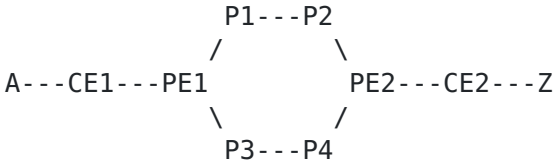

Figure 1: IGP-based MPLS Tunneling

In Figure 1 above, the four nodes A, CE1, CE2 and Z are part of the same VPN. CE2 advertises to PE2 a route to Z. PE2 binds a local label LZ to that route and propagates the route and its label via

MPBGP to PE1 with nhop 192.168.0.2. PE1 installs the VPN prefix Z in the appropriate VRF and resolves the next-hop onto the node segment associated with PE2.

In order to cope with the reality of current deployments, the SPRING architecture should allow PE to PE forwarding according to the IGP shortest path without the addition of any other signaling protocol. The packet each PE forwards across the network will contain (within their label stack) the necessary information derived from the topology database in order to deliver the packet to the remote PE.

4. Fast Reroute

FRR technologies have been deployed by network operators in order to cope with link or node failures through pre-computation of backup paths.

The SPRING architecture should address following requirements:

- support of FRR on any topology
- pre-computation and setup of backup path without any additional signaling (other than the regular IGP/BGP protocols)
- support of shared risk constraints
- support of node and link protection
- support of microloop avoidance

Further illustrations of the problem statement for FRR are to be found in [I-D.francois-spring-resiliency-use-case].

5. Traffic Engineering

Traffic Engineering has been addressed using IGP protocol extensions (for resources information propagation) and RSVP-TE for signaling explicit paths. Different contexts and modes have been defined (single vs. multiple domains, with or without bandwidth admission control, centralized vs. distributed path computation, etc).

In all cases, one of the major components of the TE architecture is the soft state based signaling protocol (RSVP-TE) which is used in order to signal and establish the explicit path. Each path, once computed, need to be signaled and state for each path must be present in each node traversed by the path. This incurs a scalability problem especially in the context of SDN where traffic differentiation may be done at a finer granularity (e.g.: application

specific). Also the amount of state needed to be maintained and periodically refreshed in all involved nodes contributes significantly to complexity and the number of failures cases, and thus increases operational effort while decreasing overall network reliability.

The source-based routing model allows traffic engineering to be implemented without the need of a signaling component.

The SPRING architecture should support traffic engineering, including:

- loose or strict options
- bandwidth admission control
- distributed vs. centralized model (PCE, SDN Controller)
- disjointness in dual-plane networks
- egress peering traffic engineering
- load-balancing among non-parallel links
- Limiting (scalable, preferably zero) per-service state and signaling on midpoint and tail-end routers.
- ECMP-awareness
- node resiliency property (i.e.: the traffic-engineering policy is not anchored to a specific core node whose failure could impact the service.

5.1. Examples of Traffic Engineering Use Cases

As documented in [I-D.filsfils-spring-segment-routing-use-cases] here follows the description of two sets of use cases:

- Traffic Engineering without Admission Control
- Traffic Engineering with Admission Control

5.1.1. Traffic Engineering without Bandwidth Admission Control

In this section, we describe Traffic Engineering use-cases without bandwidth admission control.

5.1.1.1. Disjointness in dual-plane networks

Many networks are built according to the dual-plane design, as illustrated in Figure 2:

Each access region k is connected to the core by two C routers (C(1,k) and C(2,k)).

C(1,k) is part of plane 1 and aggregation region K

C(2,k) is part of plane 2 and aggregation region K

C(1,k) has a link to C(2, j) iff k = j.

The core nodes of a given region are directly connected. Inter-region links only connect core nodes of the same plane.

{C(1,k) has a link to C(1, j)} iff {C(2,k) has a link to C(2, j)}.

The distribution of these links depends on the topological properties of the core of the AS. The design rule presented above specifies that these links appear in both core planes.

We assume a common design rule found in such deployments: the inter-plane link costs (Cik-Cjk where i<>j) are set such that the route to an edge destination from a given plane stays within the plane unless the plane is partitioned.

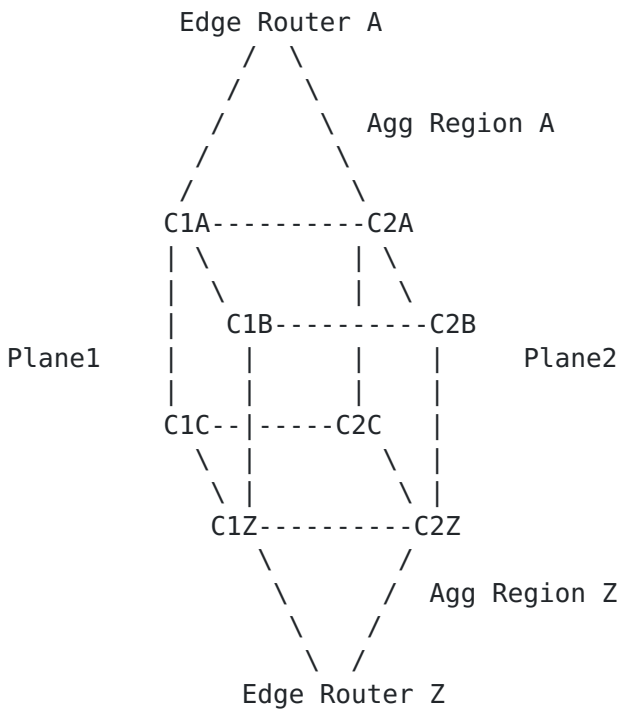

Figure 2: Dual-Plane Network and Disjointness

In this scenario, the operator requires the ability to deploy different strategies. For example, A should be able to use the three following options:

- the traffic is load-balanced across any ECMP path through the network
- the traffic is load-balanced across any ECMP path within the Plane1 of the network
- the traffic is load-balanced across any ECMP path within the Plane2 of the network

Most of the data traffic from A to Z would use the first option, such as to exploit the capacity efficiently. The operator would use the two other choices for specific premium traffic that has requested disjoint transport.

The SPRING architecture should support this use case with the following requirements:

- Zero per-service state and signaling on midpoint and tail-end routers.

- ECMP-awareness.

- Node resiliency property: the traffic-engineering policy is not anchored to a specific core node whose failure could impact the service.

5.1.1.2. Egress Peering Traffic Engineering

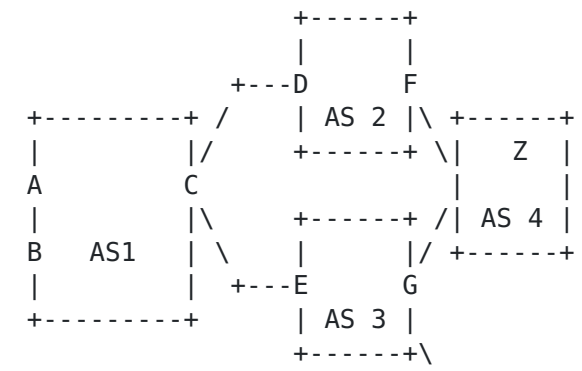

Figure 3: Egress peering traffic engineering

Let us assume, in the network depicted in Figure 3, that:

C in AS1 learns about destination Z of AS 4 via two BGP paths (AS2, AS4) and (AS3, AS4).

C may or may not be configured so to enforce next-hop-self behavior before propagating the paths within AS1.

C may propagate all the paths to Z within AS1 (add-path).

C may install in its FIB only the route via AS2, or only the route via AS3, or both.

In that context, SPRING should allow the operator of AS1 to apply the following traffic-engineering policy, regardless the configured behavior of next-hop-self:

Steer 60% of the Z-destined traffic received at A via AS2 and 40% via AS3.

Steer 80% of the Z-destined traffic received at B via AS2 and 20% via AS3.

While egress routers are known in the routing domain (generally through their loopback address), the SPRING architecture should enable following:

- identify the egress interfaces of an egress node
- identify the peering neighbors of an egress node
- identify the peering ASes of an egress node

With these identifiers known in the domain, the SPRING architecture should allow an ingress node to select the exit point of a packet as any combination of an egress node, an egress interface, a peering neighbor, and a peering AS.

5.1.1.3. Load-balancing among non-parallel links

The SPRING architecture should allow a given node should be able to load share traffic across multiple non parallel links even if these ones lead to different neighbors. This may be useful to support traffic engineering policies.

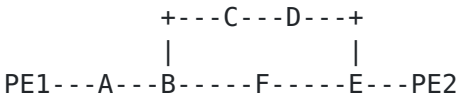

Figure 4: Multiple (non-parallel) Adjacencies

In the above example, the operator requires PE1 to load-balance its PE2-destined traffic between the ABCDE and ABFE paths.

5.1.2. Traffic Engineering with Bandwidth Admission Control

The implementation of bandwidth admission control within a network (and its possible routing consequence which consists in routing along explicit paths where the bandwidth is available) requires a capacity planning process.

The spreading of load among ECMP paths is a key attribute of the capacity planning processes applied to packet-based networks.

5.1.2.1. Capacity Planning Process

Capacity Planning anticipates the routing of the traffic matrix onto the network topology, for a set of expected traffic and topology variations. The heart of the process consists in simulating the placement of the traffic along ECMP-aware shortest-paths and accounting for the resulting bandwidth usage.

The bandwidth accounting of a demand along its shortest-path is a basic capability of any planning tool or PCE server.

For example, in the network topology described below, and assuming a default IGP metric of 1 and IGP metric of 2 for link GF, a 1600Mbps A-to-Z flow is accounted as consuming 1600Mbps on links AB and FZ, 800Mbps on links BC, BG and GF, and 400Mbps on links CD, DF, CE and EF.

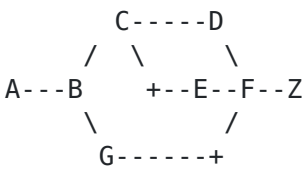

Figure 5: Capacity Planning an ECMP-based demand

ECMP is extremely frequent in SP, Enterprise and DC architectures and it is not rare to see as much as 128 different ECMP paths between a source and a destination within a single network domain. It is a key efficiency objective to spread the traffic among as many ECMP paths as possible.

This is illustrated in the below network diagram which consists of a subset of a network where already 5 ECMP paths are observed from A to M.

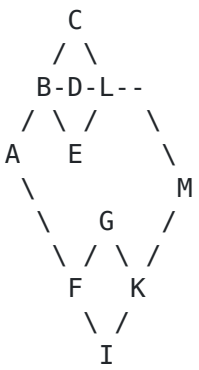

Figure 6: ECMP Topology Example

When the capacity planning process detects that a traffic growth scenario and topology variation would lead to congestion, a capacity increase is triggered and if it cannot be deployed in due time, a traffic engineering solution is activated within the network.

A basic traffic engineering objective consists of finding the smallest set of demands that need to be routed off their shortest path to eliminate the congestion, then to compute an explicit path for each of them and instantiating these traffic-engineered policies in the network.

SPRING architecture should offer a simple support for ECMP-based shortest path placement as well as for explicit path policy without incurring additional signaling in the domain. This includes:

- the ability to steer a packet across a set of ECMP paths
- the ability to diverge from a set of ECMP shortest paths to one or more paths not in the set of shortest paths

5.1.2.2. SDN/SR use-case

The SDN use-case lies in the SDN controller, (e.g.: Stateful PCE as described in [I-D.ietf-pce-stateful-pce].

The SDN controller is responsible to control the evolution of the traffic matrix and topology. It accepts or denies the addition of new traffic into the network. It decides how to route the accepted traffic. It monitors the topology and upon topological change, determines the minimum traffic that should be rerouted on an alternate path to alleviate a bandwidth congestion issue.

The algorithms supporting this behavior are a local matter of the SDN controller and are outside the scope of this document.

The means of collecting traffic and topology information are the same as what would be used with other SDN-based traffic-engineering solutions (e.g. [RFC7011] and [I-D.ietf-idr-ls-distribution].

The means of instantiating policy information at a traffic-engineering head-end are the same as what would be used with other SDN-based traffic-engineering solutions (e.g.: [I-D.ietf-i2rs-architecture], [I-D.crabbe-pce-pce-initiated-lsp] and [I-D.sivabalan-pce-segment-routing]).

In the context of Centralized-Based Optimization and the SDN use-case, here are the benefits that the SPRING architecture should deliver:

Explicit routing capability with or without ECMP-awareness.

No signaling hop-by-hop through the network.

State is only maintained at the policy head-end. No state is maintained at mid-points and tail-ends.

Automated guaranteed FRR for any topology.

Optimum virtualization: the policy state is in the packet header and not in the intermediate nodes along the path. The policy is completely virtualized away from midpoints and tail-ends.

Highly responsive to change: the SDN Controller only needs to apply a policy change at the head-end. No delay is introduced due to programming the midpoints and tail-end along the path.

5.1.2.2.1. SDN Example

The data-set consists in a full-mesh of 12000 explicitly-routed tunnels observed on a real network. These tunnels resulted from distributed headend-based CSPF computation.

We measured that only 65% of the traffic is forwarded over its shortest path.

Three well-known defects are illustrated in this data set:

The lack of ECMP support in explicitly routed tunnels: ATM-alike traffic-steering mechanisms steer the traffic along a non-ECMP path.

The increase of the number of explicitly-routed non-ECMP tunnels to enumerate all the ECMP options.

The inefficiency of distributed optimization: too much traffic is forwarded off its shortest path.

We applied the SDN use-case to this dataset implying a source route model where the path of the packet is encoded within the packet itself. This means that:

The distributed CSPF computation is replaced by centralized optimization and BW admission control, supported by the SDN Controller.

As part of the optimization, we also optimized the IGP-metrics such as to get a maximum of traffic load-spread among ECMP paths by default.

The traffic-engineering policies are supported by a source route model (e.g.: [I-D.filsfils-rtgwg-segment-routing]).

As a result, we measured that 98% of the traffic would be kept on its normal policy (over the shortest-path) and only 2% of the traffic requires a path away from the shortest-path.

Let us highlight a few benefits:

98% of the traffic-engineering head-end policies are eliminated.

Indeed, by default, an ingress edge node capable of injecting source routed packets steers the traffic to the egress edge node. No configuration or policy needs to be maintained at the ingress edge node to realize this.

100% of the states at mid/tail nodes are eliminated.

6. Interoperability with non-SPRING nodes

SPRING must inter-operate with non-SPRING nodes.

An illustration of interoperability between SPRING and other MPLS Signalling Protocols (LDP) is described here in [I-D.filsfils-spring-segment-routing-ldp-interop].

Interoperability with IPv6 non-SPRING nodes will be described in a future document.

7. OAM

The SPRING WG should provide OAM and the management needed to manage SPRING enabled networks. The SPRING procedures may also be used as a tool for OAM in SPRING enabled networks.

OAM use cases and requirements are described in [I-D.geib-spring-oam-usecase] and [I-D.kumar-spring-sr-oam-requirement].

8. Security

There is an assumed trust model such that any node imposing an explicit route on a packet is assumed to be allowed to do so. In such context trust boundaries should strip explicit routes from a packet.

For each data plane technology that SPRING specifies, a security analysis must be provided showing how protection is provided against an attacker disrupting the network by for example, maliciously injecting SPRING packets.

9. IANA Considerations

TBD

10. Manageability Considerations

TBD

11. Security Considerations

TBD

12. Acknowledgements

The authors would like to thank Yakov Rekhter for his contribution to this document.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC6564] Krishnan, S., Woodyatt, J., Kline, E., Hoagland, J., and M. Bhatia, "A Uniform Format for IPv6 Extension Headers", RFC 6564, April 2012.

[RFC7011] Claise, B., Trammell, B., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, September 2013.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, December 2013.

13.2. Informative References

[I-D.crabbe-pce-pce-initiated-lsp] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model", draft-crabbe-pce-pce-initiated-lsp-03 (work in progress), October 2013.

[I-D.filsfils-rtgwg-segment-routing]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing Architecture", draft-filsfils-rtgwg-segment-routing-01 (work in progress), October 2013.

[I-D.filsfils-spring-segment-routing-ldp-interop]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing interoperability with LDP", draft-filsfils-spring-segment-routing-ldp-interop-01 (work in progress), April 2014.

[I-D.filsfils-spring-segment-routing-mpls]
Filsfils, C., Previdi, S., Bashandy, A., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., and E. Crabbe, "Segment Routing with MPLS data plane", draft-filsfils-spring-segment-routing-mpls-01 (work in progress), April 2014.

[I-D.filsfils-spring-segment-routing-use-cases]
Filsfils, C., Francois, P., Previdi, S., Decraene, B., Litkowski, S., Horneffer, M., Milojevic, I., Shakir, R., Ytti, S., Henderickx, W., Tantsura, J., Kini, S., and E. Crabbe, "Segment Routing Use Cases", draft-filsfils-spring-segment-routing-use-cases-00 (work in progress), March 2014.

[I-D.francois-spring-resiliency-use-case]
Francois, P., Filsfils, C., Decraene, B., and R. Shakir, "Use-cases for Resiliency in SPRING", draft-francois-spring-resiliency-use-case-02 (work in progress), April 2014.

[I-D.geib-spring-oam-usecase]
Geib, R. and C. Filsfils, "Use case for a scalable and topology aware MPLS data plane monitoring system", draft-geib-spring-oam-usecase-01 (work in progress), February 2014.

[I-D.ietf-i2rs-architecture]
Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", draft-ietf-i2rs-architecture-02 (work in progress), February 2014.

[I-D.ietf-idr-ls-distribution]
Gredler, H., Medved, J., Previdi, S., Farrel, A., and S. Ray, "North-Bound Distribution of Link-State and TE Information using BGP", draft-ietf-idr-ls-distribution-04 (work in progress), November 2013.

[I-D.ietf-pce-stateful-pce]
Crabbe, E., Medved, J., Minei, I., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-08 (work in progress), February 2014.

[I-D.kumar-spring-sr-oam-requirement]
Kumar, N., Pignataro, C., Akiya, N., Geib, R., and G. Mirsky, "OAM Requirements for Segment Routing Network", draft-kumar-spring-sr-oam-requirement-00 (work in progress), February 2014.

[I-D.sivabalan-pce-segment-routing]
Sivabalan, S., Medved, J., Filsfils, C., Crabbe, E., and R. Raszuk, "PCEP Extensions for Segment Routing", draft-sivabalan-pce-segment-routing-02 (work in progress), October 2013.

Authors' Addresses

Stefano Previdi (editor)
Cisco Systems, Inc.
Via Del Serafico, 200
Rome 00142
Italy

Email: sprevidi@cisco.com

Clarence Filsfils (editor)
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Bruno Decraene
Orange
FR

Email: bruno.decraene@orange.com

Stephane Litkowski
Orange
FR

Email: stephane.litkowski@orange.com

Martin Horneffer
Deutsche Telekom
Hammer Str. 216-226
Muenster 48153
DE

Email: Martin.Horneffer@telekom.de

Ruediger Geib
Deutsche Telekom
Heinrich Hertz Str. 3-7
Darmstadt 64295
DE

Email: Ruediger.Geib@telekom.de

Rob Shakir
British Telecom
London
UK

Email: rob.shakir@bt.com

Robert Raszuk
Individual

Email: robert@raszuk.net