RCV: Network Monitoring and Diagnostic System with Interactive User Interface

Soya Park
KAIST
291 Daehak-ro, Yuseong-gu
Daejeon, South Korea
soya@kaist.ac.kr

Ketan Talaulikar
Cisco Systems
Embassy Tech Square Main Rd
Bengaluru, India
ketant@cisco.com

Chris Metz
Cisco Systems
300 E Tasman Dr, San Jose
California, United States
chmetz@cisco.com

POSTER PAPER

Abstract— **Minimizing the impact of network convergence events has been an active area of research and innovation since it usually occurs unexpectedly and triggers (often jolting) alarms in network operations centers. Examination of router logs, network management system information, and router configuration reviews are the standard tools for assessment and often do not lead to satisfactory conclusions.**

We introduce Route Convergence Visualizer providing the network operator with a fast and simple tool to understand what exactly happened and who was impacted during a network convergence event. The main novel ingredients include a) effectiveness of information delivery regarding convergence events, b) programmability of the tool, and c) application of the tool.

At last, we will enlighten an indispensable collaboration between network operators and software engineers for SDN solutions which require contributions from two disparate engineering disciplines, networking and application development.

Keywords-computer network; network monitoring; routing; visualization; collaboration

I. Introduction

Routers in a large enterprise or service provider backbone network typically operate a link-state routing protocol[1][2][3]. With a link-state routing protocol, each router keeps track of its directly attached links to other routers. When a router detects a change in the status of a link (e.g. going down due to cable break) or its session with neighboring router at the other end of a link (e.g. due to some failure at neighboring router), messages containing new link-state information are flooded to all other routers in the network and stored in each router's link-state database (LSDB). Routers receive these new link-state messages and trigger a shortest path first (SPF) computation to (re)build a new routing table and then install these changed routes in the forwarding information base (FIB). The process of detecting a link-state change, flooding this information throughout the network, executing SPF computations and programming the FIB is referred to as network convergence.

During the time the routers in the network are converging, packets might be dropped. Given the mission-criticality and volume of traffic carried by networks today, any packet loss is likely to impact customer applications and user experiences. Indeed, the network convergence time is viewed by operators and their customers as key performance metric that is factored into service level agreements (SLA).

Minimizing the impact of network convergence events has been an active area of research and innovation. Routers can now locally and immediately reroute packets around failed links and nodes in both IP and MPLS networks while the network is converging[4]. Enhancements to the SPF algorithm and faster router CPUs can reduce its computation time thus allowing the router to more quickly compute and program a new routing table[5].

Despite these enhancements, network convergence events are a bane to network operators. They usually occur unexpectedly and trigger (often jolting) alarms in network operations centers (NOCs). The aftermath of a convergence event involves figuring out what happened, where did it happen, how long was service disrupted and who was disrupted. Examination of router logs, network management system information and router configuration reviews are the standard tools for assessment and often do not lead to satisfactory conclusions.

What is needed by operators to better understand and analyze network convergence events? Two things: a new form of network instrumentation to measure and record network convergence information on each router and a central location to gather and analyze this information.

The former is provided by a new feature called Route Convergence Monitoring and Diagnostics (RCMD)[6]. This is a software implementation present on each router that records and stores time-stamped router and network convergence data in real-time and subsequently provides per-router convergence information via a structured data model. The latter is Route Convergence Visualizer (RCV) that resides on top of a centralized software defined network (SDN) controller. RCV collects the RCMD information from each router and places them in a database for visualization and analysis thus providing the network operator with a fast and simple tool to understand what exactly happened and who was impacted during a network convergence event.

This paper describes the architecture and implementation of the RCV. It was conceived and developed as a result of a "hackathon-style" collaboration between network and software engineers. At the end of the paper, we will illuminate further on the role of collaboration in SDN software engineering and application development.

Contribution. We enable RCV to perform extensive monitoring and diagnostic system for the following four requirements:

1. How to deliver information effectively in order to answer following questions?

- How do network design changes affect convergence?
- What is the route change flooding propagation delays seen in the network?
- How can network operators take the network status in at a glance?

2. How much freedom and programmability to manipulate the as operator's demand?

3. What are the application with RCV?

Previously, there are tools only offer a local per-router view of convergence event measurements. RCV is capable of retrieving a global view of the network and presenting a per-convergence event basis.

With RCV, a network operator can estimate duration and arrival time at each node along the flooding path from node A to B, SPF and FIB programming times, and what causes the network convergence event. Furthermore, it presents the grade per each link based on duration if it takes longer than usual so that the user can inspect per-link shortage as well.

RCV is applicable to wide range of network and it is easily plugged into network due to its simple and light-weight of its structure. It also conveys the network convergence data with lucid user interface and API in the format of raw data so that the users can exploit data into further automation software as their tastes.

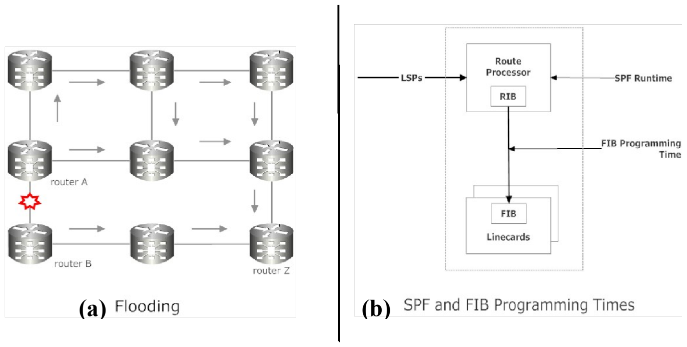


Figure 1. Network-Wide and Local Routing Convergence Functions captured by Route Convergence Monitoring & Diagnostics (a) Routing convergence functions captured by RCMD. On the left when a link goes down (router A – Router Z link), the RCMD process on each router timestamps each LSP as it arrives (b) The RCMD process measures the SPF run to completion time and how long it takes to download routes and labels from the router's route processor (RP) to the forwarding information base (FIB) on the router linecards (LC).

II. MEASURING CONVERENCE

There are several factors that impact the time it takes for a network to converge[7][8][9]:

- Event Detection by IGP
- SPF Processing by IGP
- Link-State Protocol Message (LSP/LSA) Flooding
- Route and label programming into FIB on routers

Event detection is handled by logic on the router's physical or logical interface or if configured timers expire. SPF processing and FIB programming are functions performed on each router and their respective durations are dependent on configuration and router implementation. LSP/LSA flooding is the time it takes for these messages to be propagated across the network. With the exception of event detection (which is fairly deterministic and shorter time in the convergence timeline), RCMD measures and stores information regarding the flooding, SPF and FIB programming times.

RCMD is capable of capturing various routing convergence functions (see Figure 1). It can also keep track of convergence times for selected critical prefixes (e.g. IGP address of BGP next_hop which would be the leaf node at the edge of the service provider network).

RCMD configuration and convergence measurement data are all defined in XML Schema Definition (XSD) files. The actual per-router generated RCMD data is persisted in XML files stored in a defined directory on the router processor's (RP) local storage. RCMD data persistence allows it to be gathered up and analyzed hours or days after a convergence event.

RCMD can be accessed through the router command line interface (CLI) and via XML request. However, that only offers a local per-router view of convergence event measurements. It is also possible to export a router's RCMD XML files to an external TFTP server or for this data to be polled via XML interface at regular intervals by an external application. This potentially places all RCMD information for all routers in the network at one location but the question remains what to do with that data. The answer is an application with a global view of the network and access to each router's RCMD information on a per-convergence event basis.

III. ROUTE CONVERGENCE VISUALIZER

One notion of SDN often described is that of a centralized application platform with a global view of and corresponding reach into the network. This turns out to be an ideal location to place our RCV application.

Figure 2 depicts the RCV application, it components as well as the application/controller platform and the network of

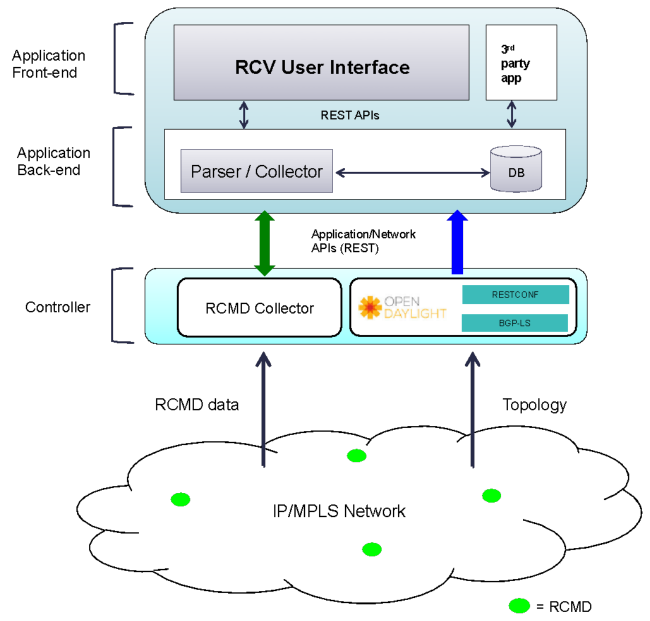


Figure 2. The architecture of Route Convergence Visualizer Applications running on top of centralized SDN application platform (controller)

RCMD-capable routers. At the bottom of the figure, there is a network of IP/MPLS routers configured to generate and store RCMD data when a convergence event occurs.

At the central location we have first the application/ controller platform composed of an instance of the OpenDaylight controller and a customized RCMD collector function. OpenDaylight (ODL) is an open-source application development platform with a suite of southbound protocol plug-ins for communicating with network devices and northbound REST APIs for application interactions[10]. It is employed because first it is straightforward to develop and run applications on top of ODL and second, there is an existing BGP Link-State (BGP-LS) plug-in for retrieving the network topology and the exposing it to applications via REST APIs[11]. The rendering of the network topology offers additional context for the operator while digesting and analyzing a convergence event. The RCMD collector is a customized file retrieval function that pulls RCMD data from each router and passed it up to the application.

The RCV application is broken up into two parts: a front-end and back-end. The back-end interfaces directly to the APIs and data presented by the application/controller platform. Additional functions including data processing, database persistence and a REST server are located in the back-end.

The RCV back-end is composed of the following:

- Parser/Collector – gathers and parses RCMD data from the RCMD collector before passing it on to the database
- Database – uses to persist RCMD data for access and analysis. MongoDB is used offering a non-relational and scalable database for storing normalized data that in our case reflects convergence events and the time and impact on prefix and MPLS label changes

TABLE I. EXAMPLES OF DETECTABLE REASONS OF NETWORK CONVERGENCE EVENT

Acronym	Reason
cf	modification at config
ls	LSP new/expiry
ln	LSP node
ll	LSP link
lp	LSP prefix
ne	neighbor event

For the network operator, the RCV application presents a simple, WEB-based user interface (UI) implemented in Cisco NeXt[12]. NeXt is a web front-end toolkit to build applications to visualize network with simple and interactive experience and high performance.

A. *User Interface*

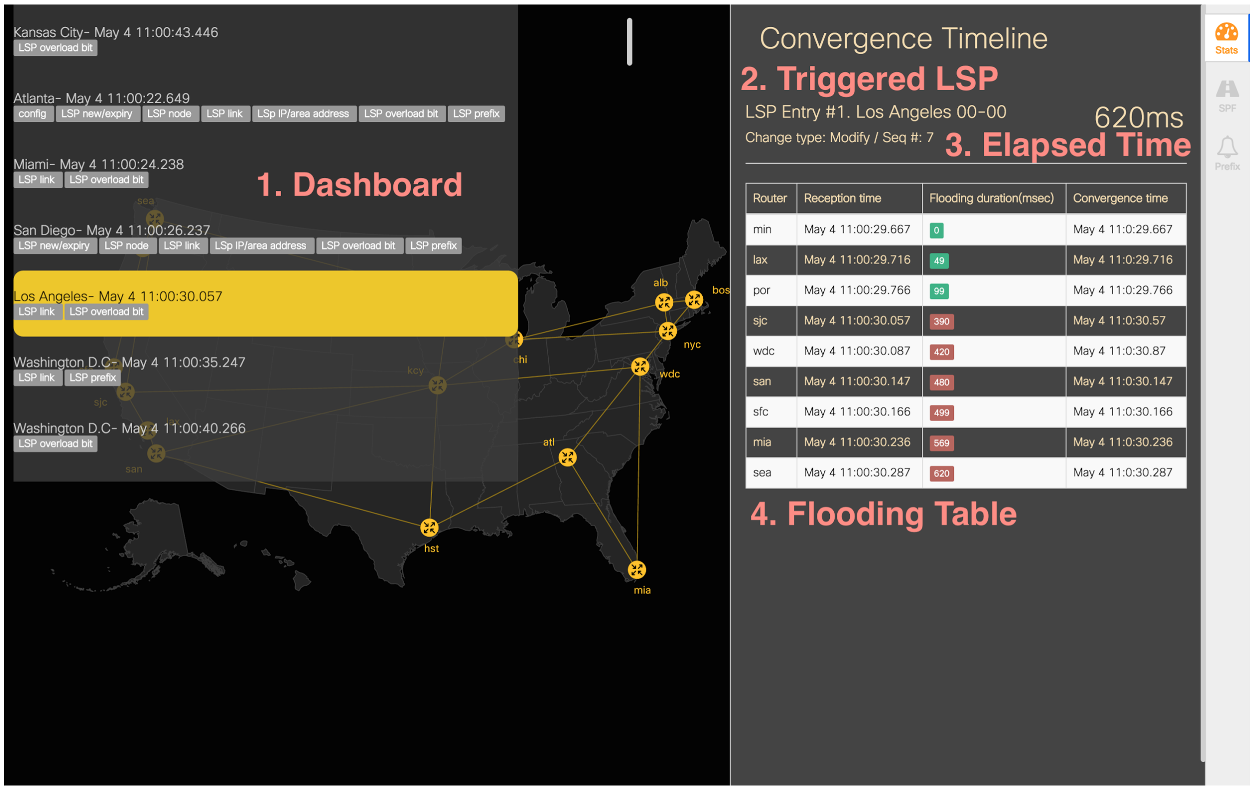
The RCV user-interface provides WEB browser access to a simple presentation of network and router convergence information.

After logging in the user is presented with the RCV dashboard (1. Dashboard) composed of a network topology (described below) and a pull-down panel listing discrete convergence events gathered and stored in the database.

Upon selecting one, data related to that convergence event (a more apt description might be epoch) slides out from the right (see figure 3). The name of the event-triggering router (2. Triggered LSP) is displayed based on the earliest recorded timestamp. The total overall network convergence time (3. Elapsed time) is shown at the top of this slide-out. This is computed by the taking the time when the LSP message was originated by a router connected to the failure point and ending at the when the last router complete its SPF computation and FIB programming. Below that is the Flooding Table (4. Flooding Table) showing various per-router convergence measurements:

- List of routers sorted by LSP reception time
- Per-router LSP reception timestamps
- Flooding duration is accumulated time of how long it takes for an LSP to travel from one router to the next router in the list
- Convergence time is the absolute time between the LSP trigger time and the maximum time under convergence process
- Reason for the convergence event(See Table 1)

There is the topology reported to the ODL platform via BGP-LS and exposed to the RCV application across REST APIs in the northbound direction. This offers the operator a point of reference only and does not necessarily depict the post


Kansas City- May 4 11:00:43.446
LSP overload bit

Atlanta- May 4 11:00:22.649
config | LSP new/expiry | LSP node | LSP link | LSp IP/area address | LSP overload bit | LSP prefix

Miami- May 4 11:00:24.238
LSP link | LSP overload bit

San Diego- May 4 11:00:26.237
LSP new/expiry | LSP node | LSP link | LSp IP/area address | LSP overload bit | LSP prefix

Los Angeles- May 4 11:00:30.057
LSP link | LSP overload bit

Washington D.C- May 4 11:00:35.247
LSP link | LSP prefix

Washington D.C- May 4 11:00:40.266
LSP overload bit

Convergence Timeline

LSP Entry #1. Los Angeles 00-00

Change type: Modify / Seq #: 7

620ms

Router	Reception time	Flooding duration(msec)	Convergence time
min	May 4 11:00:29.667	0	May 4 11:0:29.667
lax	May 4 11:00:29.716	49	May 4 11:0:29.716
por	May 4 11:00:29.766	99	May 4 11:0:29.766
sjc	May 4 11:00:30.057	390	May 4 11:0:30.57
wdc	May 4 11:00:30.087	420	May 4 11:0:30.87
san	May 4 11:00:30.147	480	May 4 11:0:30.147
sfc	May 4 11:00:30.166	499	May 4 11:0:30.166
mia	May 4 11:00:30.236	569	May 4 11:0:30.236
sea	May 4 11:00:30.287	620	May 4 11:0:30.287

Figure 3. Application User Interface. Network convergence event dashboard and a corresponding timeline of flooding event

event topology. The operator can then select a router showing SPF data.

- SPF event number
- Prefix priority sets impacted by the SPF
- Router "Churn" showing the number of routing table entries added, deleted and modified
- Maximum time offset to download IP routes from the RIB to the FIB
- Maximum time offset to download MPLS Labels via LDP to the FIB

The network operator can achieve these diagnostics via APIs - as text or user interfaces with numerous handy features.

A subsequent release will support more advanced UI capabilities.

B. APIs

RCV is also designed to enable 3rd party applications to access and retrieve RCMD from the back-end via the same set of REST APIs.

REST APIs for fetching RCMD data about network flooding and router convergence measurements were implemented. In the example below the UI formulates the REST API request and the database server would respond.

RCV provides a versatile suite of APIs for retrieving event-specific flooding statistics and router convergence processing. The user-interface described below as well as other applications can make use of these APIs.

C. RCV Application

In this section, we describe several potential applications that may utilize the RCV platform.

Network down alarming system: By fetching the API periodically or connecting web socket, network operators could be responsive to any type of network down occurrence. For instance, she could connect the API with the intra mailing system then the system would send out email to related operators as it happens. We expect RCV APIs would encourage network operators to write various application with simple scripts exploiting RCV APIs.

Filtering: Also, as calling the API selectively, the operator can enable history filtering easily. Not only per-router but also per-packet based will be available to the operator and customers.

IV. SDN APPLICATIONS AND THE ROLE OF COLLABORATION

Traditionally networking research is carried out in academic institutions, industrial labs, and companies by those familiar with and knowledgeable of complex networking systems and architectures. The advent of SDN introduces software platforms that abstract away this complexity while offering northbound APIs to software developers and engineers[13], who in fact may not know or even care about networking systems. Implicitly SDN solutions require contributions from two disparate engineering disciplines, networking and application development.

TABLE II. RCV REST API LIST

HTTP Method	URI	Description
GET	/rcv/flood	Return a lists of convergence event timestamp
GET	/rcv/flood/{time}	Return a flood stats of a specific event corresponding given time index
GET	/rcv/lsp/{lsp # + type}	Return a list of flood events in time index that has a given type
GET	/rcv/spf/{lsp #}/{hostname}	Return a SPF information of a specific event

Software engineers excel at programming but likely do not possess knowledge of networking systems and architecture. Likewise, network engineers are comfortable with the latter but limited in the former. How can engineers with different but necessarily complementary skills collaborate to build an SDN application?

The approach used in the development of the RCV application involved the network engineer exiting their comfort zone and dipping their toe into software engineering waters. The software engineer at a minimum must acquire an abstract view of the network problem solutions. The result is an intersection of the knowledge between the network engineers (N) and the software engineers (S) depicted as N ^ S.

How can this collaboration be achieved quickly? First network engineers need to work with software coding techniques. Nothing too complex, a python script parsing logs to extract information needed for the application is one example. Another example might involve reviewing a JSON schema (e.g. representing a network topology) that the software engineer will use in programming the application.

Second we suggest joint participation in a hackathon. This is where network and software engineers form a team to identify a problem, hack up a solution in code and complete before a specific time (typically 24 hours) expires.

In fact, this is how RCV was developed. A couple of network engineers identified a problem and partnered up with a software engineer at a hackathon. The general idea and rudimentary code snippets were contributed by the network engineer. The snippets plus expertise in python, databases and javascript formed the building blocks used by the software engineer to complete the application.

Hackathons facilitating network and software engineering collaboration offer many benefits. Research ideas and prototypes can be converted to running code and made more accessible to a wider audience. It may even provide lasting advances[14][15]. This new paradigm of network and software engineering collaboration was demonstrated in the creation of the RCV application.

TABLE III. RCV REST API EXAMPLES. (TOP) THE RESPONSE THAT SHOWS A LIST OF CONVERGENCE EVENT. THE REASON TAG GIVES INFORMATION ABOUT THE CAUSE OF THE CONVERGENCE EVENT (BOTTOM) SHOWS A FLOOD STATS OF A SPECIFIC EVENT CORRESPONDING GIVEN TIME INDEX

```
GET api/v0/rcv/flood
HTTP/1.1 200 OK
"response":[
 {
   "reason":"cf ls ln ll lt lo lp",
   "trigger_LSP":"0000.0000.0017.00-00",
   "time":"May  4 11:00:18.468"
 },
 {
   "reason":"ls ln ll lt lo lp",
   "trigger_LSP":"0000.0000.0020.00-00",
   "time":"May  4 11:00:18.777"
 }
]
```

```
GET api/v0/rcv/flood/May  4 11:00:18.468
HTTP/1.1 200 OK
"response":[
 {
   "flood": [
       "0000.0000.0017.00-00_New_5_0",
       "0000.0000.0019.00-00_Modify_3_114"
    ]
 }
]
```

V. CONCLUSION

RCMD is a distributed, lightweight router software feature that records the time and duration of locally observed convergence processes and the arrival times of event-triggered LSP packets. A view and analysis of network-wide convergence events are now possible by the implementation of a centralized, SDN application called RCV. The intent is that RCV will expedite analysis and troubleshooting of events resulting in network convergence.

Looking ahead, RCV can be improved and extended across several fronts. A standard method for describing local RCMD information and then pushing it up to the RCV platform could be accomplished via Netconf/YANG[16][17]. Measurements on other functions related to network “churn” including BGP, RSVP-TE and FRR are possible. Analysis of RCMD measurement coupled with known topology and link speeds could yield precise numbers on volume of data impacted. And finally, RCV and its capabilities could be rolled into other routing analytics packages.

We anticipate application developers, in collaboration with network engineers will take advantage the REST API and SDN platforms to create applications such as RCV. We hope this new paradigm of joint cooperation will motivate and inspire network and software engineers to build new SDN applications that solve problems.

REFERENCES

1. J. Moy, "RFC 2328: Ospf version 2.", 1998.
2. C. Metz, "At the core of IP networks: link-state routing protocols." IEEE Internet Computing 3.5 (1999): 72-77.
3. R. Callon, "RFC 1195, entitled Use of OSI ISIS for routing in TCP.", 1990.
4. C. Metz, B. Colby, and F. Clarence, "Beyond mpls... less is more." IEEE Internet Computing 11.5 (2007): 72.
5. C. Alaettinoglu, J. Van, and Y. Haobo, "Towards milli-second IGP convergence." Proceedings of NANOG. Vol. 20. 2000.
6. F. Clarence, K. J. Talaulikar, and B. Muthuvarathan, "Route convergence monitoring and diagnostics." U.S. Patent No. 9,237,075. 12 Jan. 2016.
7. P. Francois, C. Filsfils, J. Evans, and O. Bonaventure, "Achieving sub-second IGP convergence in large IP networks." ACM SIGCOMM Computer Communication Review 35.3 (2005): 35-44.
8. C. Villamizar, "Convergence and restoration techniques for ISP interior routing." NANOG, 2002.
9. S. Poretsky, B. Imhoff, and K. Michielsen. "RFC 6413: Benchmarking Methodology for Link-State IGP Data-Plane Route Convergence.", 2011.
10. "Opendaylight - Open-Soruce SDN controller homepage." [Online]. Available: https://wiki.opendaylight.org/view/Main_Page
11. "Opendaylight BGP-LS homepage" [Online]. Available: https://wiki.opendaylight.org/view/BGP_LS
12. "Cisco NEXT homepage". [Online]. Available: https://developer.cisco.com/site/neXt/
13. S. Wee, "1.4 The next generation of networked experiences." 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC). IEEE, 2014.
14. "IETF Hackathon homepage." [Online]. Available: https://www.ietf.org/blog/2015/05/ietf- hackathon/
15. A. Sivaraman, K. Winstein, P. Varley, J. Batalha, A. Goyal, S. Das, ... and H. Balakrishnan, "Protocol design contests." ACM SIGCOMM Computer Communication Review 44.3 (2014): 38-44.
16. M. Bjorklund, "RFC 6020: YANG-a data modeling language for the network configuration protocol.", 2010.
17. R. Enns, M. Bjorklund, and J. Schoenwaelder, "RFC 6241, Network Configuration Protocol (NETCONF).", 2011.