



Dynamisches Traffic-Engineering

Motivation

Das Routing im Internet reagiert automatisch und schnell auf Hardware-Ausfälle. Zur Vermeidung von Netzwerküberlastung und zur Steigerung des Netzwerkdurchsatzes wenden viele Provider sog. „Traffic Engineering“ (Optimierung des Routings) an – allerdings auf Zeitskalen von mehreren Stunden. Schnelle dynamische Reaktionen auf plötzliche Lastschwankungen und Netzwerküberlastungen sind damit leider nicht möglich.

Am Lehrstuhl für Netzwerkarchitekturen wurde daher REPLEx entwickelt: ein verteilter Algorithmus, welcher dynamisches, schnell reagierendes Traffic-Engineering ermöglicht. REPLEx wurde in Netzwerksimulationen mit realistischem Verkehr und realistischen Topologien ausgiebig untersucht und erzielte dabei hervorragende Ergebnisse. Bislang wurden von uns nur Szenarien untersucht, dass REPLEx im Netzwerk eines einzelnen Providers mit dem Routingprotokoll OSPF verwendet wird. Es sind jedoch noch etliche weitere interessante Szenarien denkbar, z.B. bei Verwendung über Provi-dergrenzen hinweg, mit anderen Routingprotokollen usw.

Aufgabenstellungen

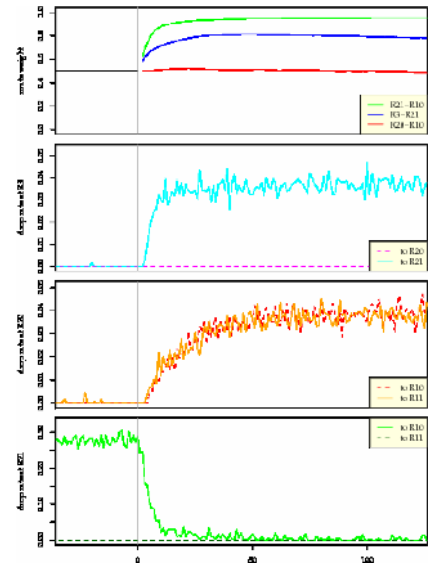
Es sind verschiedene Aufgabenstellungen sowie deren Kombination möglich:

- REPLEx im BGP-Interdomain-Kontext über Provider-Grenzen hinweg (hauptsächlich Auswertung)
- REPLEx in MPLS-Netzen
 - ① Implementierung in Java
 - ① Auswertung
- Verschiedene Erweiterungen
 - ① Parity-Routing (hauptsächlich Auswertung)
 - ① Update-Frequenz; weiche Änderungen (Algorithmenentwicklung, Implementierung in Java, Auswertung)
- Verhalten bei Link-Ausfällen (Auswertung)

Die einzelnen Aufgabenstellungen unterscheiden sich hinsichtlich der Art und des Umfangs der Arbeit.

Die Implementierungsarbeiten erfolgen jeweils als Erweiterung des in Java geschriebenen Netzwerksimulators SSFNet v2.0.

Für die Auswertung müssen zunächst geeignete Netzwerk-Szenarien entworfen werden, welche dann in Simulations-Beschreibungen für SSFNet umzusetzen sind (hierfür empfiehlt sich Perl, Python o. ä.). Anschließend werden die sehr umfangreichen detaillierten Ausgaben von SSFNet analysiert und statistisch aufbereitet (mit GNU R oder Gnuplot, Perl, Python, ...).



Voraussetzungen

Grundkenntnisse von Netzwerkprotokollen (insbesondere TCP/IP und Routing).

Für die Implementierungsarbeiten: Java-Kenntnisse, gute Auffassungsgabe.

Für die Auswertung: Kenntnisse in Perl/Python/... sowie GNU R/gnuplot/... sparen Arbeit.

Stichworte

Routing, Traffic-Engineering, Simulation, Analyse



Mehr Informationen am Lehrstuhl für Netzarchitekturen und Netzdienste bei:

Nils Kammenhuber <kammenhuber@net.in.tum.de>