Real-Time-Core- und -Edge-Switches

Klassengesellschaft im Netz – die Theorie

LAN-Switches müssen die unterschiedlichsten Kommunikationsströme über eine Infrastruktur transportieren. Dafür, dass die Daten ihren individuellen Anforderungen entsprechend fließen, sorgen Class-of-Service-Mechanismen, die eine Klassengesellschaft im Netz etablieren.

Wir wollten wissen, welche CoS-Queuing-Mechanismen heute in aktuellen LAN-Switches wirklich implementiert sind und wie gut diese derzeit arbeiten. Aus diesem Grund haben wir ein Feld von Class-of-Service-fähigen LAN-Switches in unseren Real-World Labs an der FH Stralsund einer umfangreichen Testprozedur unterzogen. Teil 1 unseres Testberichtes steht in diesem Haft ab Seite 14. An dieser Stelle sollen aber zunächst die theoretischen Grundlagen, die zum Verständnis unserer LAN-Tests erforderlich sind, näher beleuchtet werden.

Die Switch-Hersteller versuchen, das beste Verhältnis zwischen guter Verteilung und einfacher Umsetzung der Queuing-Strategien zu finden.

Lastspitzen sorgen in vielen auf Ethernet basierenden LANs immer wieder dafür, dass es im Netz eng wird, und dann kommt es je nach Aus- und Überlastung zu teils erheblichen Datenverlusten. Im Zeitalter konvergenter Netze, die Echtzeitapplikationen wie Voice- und Video-over-IP aber auch Produktionsteuerungsdaten in das klassische Datennetz integrieren, führen solche Datenverluste in der Praxis zu empfindlichen Kommunikationsstörungen und Produktionsausfällen. Um solchen Problemen vorzubeugen rüsten die Ethernet-Hersteller ihre Switches mit einer zusätzlichen Funktion aus, die es ermöglichen soll, bestimmten Applikationen die Vorfahrt im Netzwerk einzuräumen, wenn es einmal eng wird. Diese Priorisierungs-Mechanismen werden allgemein als Class-of-Service oder – verfälschend in Anlehnung an ATM – als Quality-of-Service bezeichnet. Üblich ist eine achtstufige Priorisierung auf Layer-2 nach IEEE 802.1p/Q oder auf Layer-3 nach RFC 1349/2474/2475. Die jeweils zugeordnete Priorisierung lesen die Systeme aus den Headern der Datenpakete aus.

Wie Switches die Datenpakete dann gemäß ihrer Priorität behandeln, hängt von den jeweils implementierten Queuing-Mechanismen ab. So besteht beispielsweise die Möglichkeit, bestimmten Daten absolute Vorfahrt einzuräumen oder auch für niedrigere Prioritäten Mindestdurchsatzraten zu garantieren. Eine gute Queuing- oder Scheduling-Strategie sollte folgende Voraussetzungen erfüllen:

- Sie muss die faire Verteilung der Bandbreite auf die verschiedenen Serviceklassen unterstützen. Dabei sollte auch die Bandbreite für besondere Dienste berücksichtigt werden, so dass es zu bestimmten Gewichtungen bei der Fairness kommen kann.
- Sie bietet Schutz zwischen den verschiedenen Serviceklassen am Ausgangsport, so dass eine Serviceklasse mit geringer Priorität nicht die anderen Serviceklassen anderer Queues beeinflussen kann.
- Wenn ein Dienst nicht die gesamte Bandbreite verwendet, die für ihn reserviert ist, dann sollte diese Überkapazität auch anderen Diensten zur Verfügung stehen, bis der eigentliche Dienst diese Kapazitäten wieder benötigt.
- Ein schneller Algorithmus, der hardwaremäßig implementiert werden kann, muss für die Strategie existieren. Nur dann kann diese Strategie auch auf Switches eingesetzt werden, die mit hoher Geschwindigkeit arbeiten. Algorithmen, die nur softwareseitig implementiert werden können, eignen sich lediglich für langsame Switches.

Störungen im LAN – Packet-Loss, Jitter & Co.

Die Übertragung von einem Endpunkt im Netzwerk zum anderen erfordert eine gewisse Zeit. Dabei gibt es zunächst einen festen Teil, der durch die Auswahl der zu verwendenden Codecs, also der Sprach-Digitalisierungs-Algorithmen, und der Netzwerkkomponenten beeinflussbar und ziemlich gut berechenbar ist. Dieser wird durch die Zeit, die die Kodierungsalgorithmen an beiden Endpunkten benötigen, durch die Hardware-Durchlaufzeit auf den beteiligten End- und Knotenpunkten und durch die rein physikalischen Übertragungsgeschwindigkeiten der verschiedenen Medien über bestimmte Entfernungen festgelegt. Zusätzlich entstehen Verzögerungen beispielsweise durch volle Warteschlangen bei Überlast oder durch die eventuelle Wahl alternativer Routen zum Zielpunkt. Die beiden letzteren Punkte können auch die Ursache für zwei andere Übertragungsfehler sein. Beim sogenannten Jitter treffen Pakete, die in regelmäßigen Intervallen in das Netz geschickt werden, in unregelmäßigen Abständen beim Empfänger ein. Ist bei isochromem Datenverkehr wie der IP-Sprachübertragung ein Paket zu schnell am Ziel, dann kann es für die Ausgabe noch nicht verwendet werden. Kommt es dagegen später als erwartet, können Lücken in der Sprachwiedergabe entstehen. Diesem Jitter kann man durch den Einsatz eines Jitter-Buffers entgegenwirken, der Pakete aus dem Netz entgegennimmt und verzögert, aber gleichmäßig an die Dekodiereinheit weitergibt. Natürlich erhöht sich dadurch auch der Delay. Treffen die Pakete beim Empfänger in einer anderen Reihenfolge ein, als vom Sender beabsichtigt, spricht man von einem Sequence-Error. Häufigste Ursache hierfür ist, dass einige zu einer Übertragung gehörende Pakete aufgrund einer Überlast regeroutet werden und so ihr Ziel auf einem anderen, möglicherweise langsameren Weg erreichen. Wie gut solche Fehler in der Paket-Reihenfolge ausgeglichen beziehungsweise überspielt werden können, hängt in erster Linie von der Länge des Jitter-Buffers ab.

Gehen bei der Übertragung Pakete ganz verloren (Packet-Loss), dann sind die Auswirkungen desto größer, je höher die Anzahl der Sprachdaten-Bytes in dem verlorenen Paket war und je stärker der Codec komprimiert. Gehen mehrere aufeinander folgende Pakete verloren (Consecutive-Packet-Loss), sind die Auswirkungen auf die Sprachqualität deutlich stärker, als wenn die

Info

Das Testfeld

Core-Switches
Extreme Networks
Alpine 3804
Hewlett-Packard
HP ProCurve Switch 5308 XL
MRV OptiSwitch Master

Edge-Switches
Cisco 2950G-24
D-Link DES-3326S
Extreme Networks
Summit48si
Hewlett-Packard
HP ProCurve Switch 5308 XL
MRV OptiSwitch Master

Verluste gleichmäßig streuen. Diese Verlustart tritt überwiegend in Burst-Situationen auf. Die Ursache für Packet-Loss liegt häufig darin, dass auf dem Übertragungsweg Bandbreitenengpässe auftreten und durch länger dauernde Bursts Warteschlangen überlaufen, weshalb dann Pakete verworfen werden oder Pakete in den Warteschlangen so weit verzögert werden, dass sie nicht mehr über den Jitter-Buffer sinnvoll verwendet werden können. Werden die Jitter-Buffer sehr groß ausgelegt, um entsprechende Netzwerkfehler wie Sequence-Errors oder Jitter auszugleichen, führt diese Technik selbst dann zu einer zu großen Verzögerung, die dann gleichfalls die Echtzeitkommunikation stört. Jitter-Buffer verringern also Probleme, die durch Jitter und Squenz-Error entstehen können, erzeugen aber ihrerseits zusätzliche Delay-Zeiten. Sehr gute Endgeräte verwalten den Jitter-Buffer daher dynamisch.

Bei entsprechender Überlast im Netz sind Datenverluste ganz normal, jedoch sollen sie durch die Priorisierungsmechanismen in der Regel auf nicht echtzeitfähige Applikationen verlagert werden. Arbeitet diese Priorisierung nicht wie vorgesehen, kommt es auch im Bereich der hoch priorisierten Sprachdaten zu Verlusten. Für eine realitätsnahe und aussagefähige Auswertung der Messergebnisse ist es darüber hinaus entscheidend zu wissen, welche Framegrößen in welchen Verteilungen in realen Netzwerken vorkommen. Analysen der Verteilung der Framegrößen, beispielsweise für das NCI-Backbone oder von den Applikationen her typischer Business-DSL-Links haben ergeben, dass rund 50 Prozent aller Datenrahmen in realen Netzwerken 64 Byte groß sind. Die übrigen rund 50 Prozent der zu transportierenden Datenrahmen streuen über alle Rahmengrößen von 128 bis 1518 Byte.

Für Echtzeit-Anwendungen wie die Voice- oder Video-Übertragung ist zunächst das Datenverlustverhalten von entscheidender Bedeutung. Ab 5 Prozent Verlust ist je nach Codec mit deutlicher Verschlechterung der Übertragungsqualität zu rechnen, 10 Prozent führen zu einer massiven Beeinträchtigung, ab 20 Prozent Datenverlust ist beispielsweise die Telefonie definitiv nicht mehr möglich. So verringert sich der R-Wert für die Sprachqualität gemäß E-Modell nach ITU G.107 schon bei 10 Prozent Datenverlust um je nach Codec 25 bis weit über 40 Punkte, also Werte, die massive Probleme im Telefoniebereich sehr wahrscheinlich machen. Aufgrund ihrer Bedeutung für die Übertragungsqualität haben wir daher das Datenrahmenverlustverhalten als primäres K.o.-Kriterium für unsere Tests definiert. Die Parameter Latency und Jitter – die wir standardmäßig ebenfalls messen – sind dann für die genauere Diagnose und weitere Analyse im Einzelfall wichtig. Sind jedoch die Datenverlustraten von Hause aus schon zu hoch, können gute Werte für Latency und Jitter die Sprachqualität auch nicht mehr retten. Dafür, dass es zu solchen massiven Datenverlusten im Ethernet-LAN erst gar nicht kommt, sollen entsprechend gut funktionierende Priorisierungsmechanismen sorgen. Sie tun dies aber durchaus nicht immer, wie die Erfahrung aus vorhergehenden Tests zeigt.

Qualitätssicherungsverfahren im LAN

Auf Ethernet basierende LANs ermöglichen eine kostengünstige durchgehende Netzwerk-Lösung vom Backbone-Bereich bis an die Endgeräte am Arbeitsplatz und sind nicht zuletzt aus diesem Grund weltweiter Standard in nahezu allen Unternehmen und Institutionen. Allerdings bietet das Ethernet auf Layer-2 und -3 nicht die gleiche Übertragungsqualität wie von Hause aus echtzeitfähige Technologien, beispielsweise ATM. Das Ethernet-Protokoll ist zwar einfacher und arbeitet mit geringerem Overhead, die Übertragungen erfolgen aber ohne vorherigen Aufbau einer Verbindung oder die Aushandlung der Qualität der Übertragungsstrecke von Endpunkt zu Endpunkt. Für alle Applikationen wird nur eine Best-Effort-Behandlung – so gut, wie eben möglich – bereitgestellt, unabhängig von deren tatsächlichen Anforderungen oder den Anforderungen der Nutzer. Die Absicherung einer Verbindung erfolgt – wenn überhaupt – erst in Protokollen höherer Ebenen wie dem TCP.

In Ethernet-Netzwerken und unter Verwendung des TCP/IP-Protokolls – und damit auch im Internet, Intranet oder Extranet – gibt es also keine garantierten Verbindungseigenschaften. Deshalb ist auch die Implementierung von Quality-of-Service oder kurz QoS, wie man es von ATM kennt, nicht möglich. Trotzdem versuchen die Ethernet-Produktentwickler durch Priorisierung und Reservierung von Ressourcen auch in IP-Netzen verschiedene Serviceklassen, die Class-of-Service oder CoS, zu etablieren. Allgemein sind zwei Wege zu unterscheiden, Service-Qualitäten zu realisieren, zum einen über die Reservierung von Netzwerkressourcen, die Resource-Reservation, und zum anderen über eine bevorzugte Behandlung bestimmter Pakete bei deren Weiterleitung, die Daten-Priorisierung.

Grundlage für letztere ist die Entscheidung, welches Paket welche Priorität besitzt. Diese Entscheidung kann auf Grundlage der generell zur Verfügung stehenden Informationen aus den Headern der Ebenen 2, 3 oder 4 erfolgen. So ist es möglich, den Verkehr beispielsweise hinsichtlich der Quell- und Zieladressen (MAC oder IP) oder der Protokoll- und Portnummern einzuteilen, natürlich in Abhängigkeit davon, bis in welche Ebene das Netzwerkgerät die Protokoll-Header analysieren kann. Geht man einen Schritt weiter, kann man in den Protokoll-Headern der verschiedenen Ebenen bestimmte Bits gezielt setzen und so die Zugehörigkeit eines Paketes zu einer Prioritätsklasse kennzeichnen.

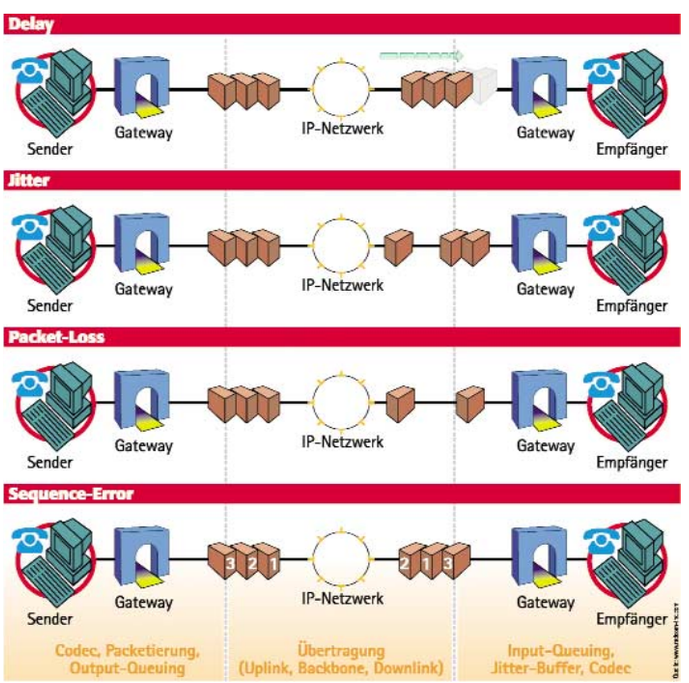
Die Hierarchie der Prioritätsentscheidungen auf den verschiedenen Layern, die ja durchaus widersprüchlich sein kann, ist für jeden Switch intern gelistet und entweder frei konfigurierbar oder fest vorgegeben. Zu beachten ist auch, dass Layer-2-Priorisierungen auf dem Weg durch ein LAN in der Regel verloren gehen, sobald sie auf Layer-3 geswitched beziehungsweise geroutet werden. Die Konfiguration des aktiven Netzwerks, das intelligent die Priorisierungsmechanismen nutzen soll, ist daher gerade in heterogenen Umfeldern nicht gerade trivial. Häufig wird der IT-Verantwortliche gut beraten sein, wenn er sich schon aus Gründen einer vollständigen Kompatibilität für ein Netzwerk aus einer Hand entscheidet. Bei größeren Netzen ist auch eine entsprechende CoS-Management-Software unerlässlich, um die zur Verfügung stehenden Priorisierungsmechanismen auch wirklich effizient nutzen zu können.

IEEE 802.1p /802.1Q

Für die Ebene 2, den Data-Link-Layer, ist in der Spezifikation IEEE 802.1Q die VLAN-Funktionalität beschrieben, die eigentlich dazu gedacht ist, auf Switches virtuelle LANs einzurichten und so unabhängig von

der physikalischen Struktur eine logische Unterteilung des Netzwerks zu erhalten. Die Zuordnung eines Paketes zu einem VLAN erfolgt mit Hilfe einer Marke, dem Tag, im Layer-2-Header. In diesem Tag ist neben der VLAN-ID unter anderem auch ein Feld »User Priority« vorgesehen. Die Nutzung dieses 3-Bit-Feldes zur Einteilung der Pakete in acht mögliche Prioritätsklassen ist in der Spezifikation IEEE 802.1p festgehalten. »Mögliche« Prioritätsklassen sind dies, weil die tatsächlich zur Verfügung stehenden Warteschlangen unterschiedlicher Priorität von der Hardware des Switches abhängen. Die meisten Systeme bieten mindestens vier verschiedene Queues.

Störfaktoren



Störfaktoren wie Pcket-Loss, Delay oder Jitter können die Übertragung von Real-Time-Apllikationen empfindlich stören und beispielsweise die VoIP-Telefonie unmöglich machen.

Type-of-Service

Eine weitere Möglichkeit der Zuordnung eines IP-Paketes ist die Nutzung des Type-of-Service-Byte, kurz ToS, im IP-Header Version 4. Dazu sind zwei Varianten beschrieben. RFC 791 definiert mit den Bits 0 bis 2 acht Klassen, von »Routine« über »Immediate« bis zu »Network-Control«. Pakete mit einem höheren Octal-Wert in diesem 3-Bit-Feld werden vorrangig behandelt (IP-Precedence). Variante 2 verwendet die Bits 3 bis 6, um eine normale, und vier, um besondere Service-Klassen zu kennzeichnen. Festgehalten ist dies in RFC 1349. Ungünstigerweise wird dieses vier Bit große Teilfeld des ToS-Byte ebenfalls als Type-of-Service bezeichnet. Es gibt also im IP-Header ein ToS-Byte und darin enthalten ist ein ToS-Feld. Pakete können anhand des ToS-Feldes entsprechend der eingestellten Klasse Warteschlangen unterschiedlicher Priorität zugeordnet werden. Im IP-Header Version 6 ist ebenfalls ein Byte für eine Klasseneinteilung vorgesehen. Es wird treffend als »Class« bezeichnet und könnte ähnlich verwendet werden.

Diffserv

Eine Arbeitsgruppe der IETF stellte 1997 eine alternative Implementation des ToS-Byte vor. Auch bei den Differentiated-Services, kurz Diffserv, wird dieses Byte dazu verwendet, um Pakete mit Markierungen zu versehen, die dann auf den Netzwerkknotenpunkten eine bestimmte Behandlung bei der Weiterleitung zum nächsten Knoten bewirken (Per-Hop-Behavior). Dazu erhält dieses Byte im IP-Header per Definition eine neue Bedeutung und wird in diesem Anwendungsfall dann als Differentiated-Service-Byte oder kurz DS-Byte bezeichnet.

Die Diffserv-Spezifikation nach RFC 1349 definiert sechs Bits, die dazu dienen, den Differentiated-Services-Code-Point festzulegen. Diese sechs Bits werden genutzt, um verschiedene Service-Klassen zu definieren. Jede Netzwerkkomponente entscheidet anhand dieser Bits, wie die entsprechenden Pakete zu behandeln sind, und steuern das Per-Hop-Behavior. Die sechs Bits sind nochmals in zwei mal drei Bits unterteilt. Diese Struktur ist in RFC 1349 festgeschrieben, aber letztendlich ist es den Herstellern beziehungsweise den Netzwerkadministratoren freigestellt, wie sie diese Bits genau nutzen. Eine sinnvolle Diffserv-Anwendung ist daher nur möglich, wenn ein Managementsystem durchgängig die notwendigen Service-Klassen-Zuordnungen steuert. Die insgesamt 64 Codierungs-Möglichkeiten müssen auf die vorhandenen Hardware-Queues beziehungsweise auf die zur Verfügung stehenden Links abgebildet werden und so dafür sorgen, dass die unterschiedlichen Dienste mit der gewünschten Qualität übertagen werden können. Diese Mechanismen müssen in einer Domäne konsistent arbeiten und zwischen verschiedenen Domänen durch Mapping gesichert werden.

Die Funktionsweise bei der Priorisierung ist im Grunde immer die gleiche. Pakete werden auf den Gateways und Knotenpunkten anhand dieser Unterscheidungsmerkmale in den Headern den Warteschlangen oder Queues unterschiedlicher Priorität zugeordnet. Die Queues höherer Priorität werden dann entsprechend der Policy der entsprechenden Queuing-Mechanismen bevorzugt weitergeleitet. Welches Prinzip dieser Bevorzugung zu Grunde liegt, ist unterschiedlich. In vielen Fällen sollte eine Priorisierung aber nicht ohne eine Festlegung einer gewissen Bandbreite erfolgen. Diese könnte beispielsweise so aussehen, dass die Queue mit der höchsten Priorität nur eine bestimmte maximale Bandbreite erhält. Sonst kann es passieren, dass bei einer Überlast ausschließlich hoch eingestufte Pakete transportiert werden, während sich die Pakete in den unteren Queues stauen, bis sie verworfen werden. Das Festlegen einer minimalen Bandbreite für Pakete niedrigerer Priorität erfüllt den gleichen Zweck. Moderne Switches verschieben diese Grenzen dynamisch, abhängig vom momentanen Verkehr. Zu beachten ist jedoch, dass bei voller Ausreizung der entsprechenden Bandbreiten und der den Queues zugeordneten Buffern auch Pakete höherer Priorität keine Chance mehr haben, transportiert zu werden, und ebenso verfallen können. Hierin liegt ein grundsätzlicher Nachteil der Ethernet-Technologie. Obwohl eigentlich alle aktuellen Netzwerkgeräte das ToS- beziehungsweise DS-Byte auswerten können, ist diese Funktion in den seltensten Fällen aktiviert und wird höchstens im In-House-Bereich oder anderen abgegrenzten und kontrollierbaren Umgebungen genutzt.

Queuing-Mechanismen

Die Hersteller von Switches verwenden oft eigene Namen für die CoS-Queuing-Strategien oder ändern die eigentliche Strategie nach ihren Vorstellungen ab. Oft werden auch verschiedene Strategien miteinander kombiniert, um die Ergebnisse zu verbessern. Die ursprünglichen Queuing-Strategien sind:

- First-In First-Out (FIFO),
- Stirct- und Rate-Controlled-Priority-Queuing (PQ),
- Fair-Queuing (FQ),
- Weighted-Fair-Queuing (WFQ),
- Weighted-Round-Robin-Queuing (WRR), auch als Class-Based-Queuing (CBQ) bezeichnet und
- Deficit-Weighted-Round-Robin-Queuing (DWRR).

Die Ersten werden die Ersten sein

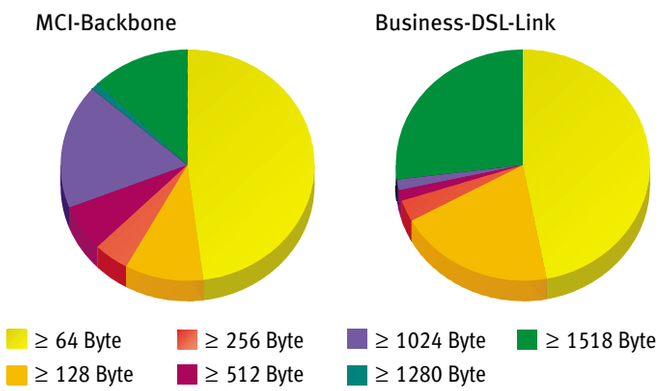
Die einfachste Queuing-Strategie ist das First-In-First-Out-Queuing (FIFO). Dabei werden alle Frames in einer einzelnen Queue gespeichert – und zwar in genau der Reihenfolge, in der sie beim Switch ankommen. Für den Ausgangsport werden die Daten dann sequenziell aus der Warteschleife gelesen. Diese Strategie wird oft auch als First-Come-First-Serve-Queuing (FCFS) bezeichnet. FIFO erzeugt auf softwaregesteuerten Switches extrem wenig Last für die CPU, so dass sie immer noch sehr schnell arbeiten können und dennoch recht preiswert herzustellen sind. Auch das Verhalten von FIFO-Queuing ist sehr gut vorhersagbar. Frames werden nicht dauerhaft gespeichert, und der maximale Delay eines Frames ist abhängig von der Größe der verwendeten Queue. Dadurch kann man mit kurzen Queues einen schnellen Switch realisieren, der billig ist und einen geringen Delay bietet. Nachteilig an dieser Strategie ist, dass die gepufferten Frames nicht unterschiedlich behandelt werden können und somit keine Unterstützung für verschiedene Dienste möglich ist. Auch beeinflusst eine Queue alle Datenströme, die

durch die Queue gehen. Steigt also die »Verstopfung« der Queue, können zeitkritische Dienste wie Sprache nicht mehr vernünftig abgearbeitet werden. Ein Burst kann die gesamte Größe des Speichers ausnutzen, wodurch Datenströme von anderen Verbindungen nicht mehr gespeichert werden können und verworfen werden. Wenn keine andere Scheduling-Strategie beim Switch eingestellt ist, so ist standardmäßig meist FIFO-Queuing implementiert.

Priority-Queuing (PQ) ist die Grundlage aller Scheduling-Strategien, die eine Unterteilung von verschiedenen CoS-Klassen ermöglichen. Dazu wird bei PQ für jede CoS-Klasse eine Queue angelegt und verwaltet. Die einzelnen Queues werden nach dem FIFO-Prinzip abgearbeitet. CoS wird dadurch erreicht, dass die Frames einer Queue erst dann abgearbeitet werden, wenn die Queues höherer Priorität keine Frames mehr speichern. Die Vorteile liegen bei PQ darin, dass es wie FIFO auch für softwaregesteuerte Switches geeignet ist, da es relativ wenig CPU-Last erzeugt. Weiterhin besteht die Möglichkeit, gepufferte Frames in einer einfachen Art und Weise zu organisieren. So können verschiedene Verkehrsklassen unterschiedlich behandelt werden. Zum Beispiel können so Echtzeitanwendungen wie Sprache oder Video eine Priorität bekommen, die höher ist als die anderer Anwendungen. Die Echtzeitanwendungen werden dann schneller weitergeleitet als andere Anwendungen. Nachteilig am PQ ist, dass, wenn viel hoch priorisierter Verkehr auf einen Ausgangsport geleitet wird, die Queues mit geringerer Priorität nur wenig oder auch gar nicht bedient werden. Man kann dieses Problem schon mit einem einfachen Begrenzen der Datenströme mit hoher Priorität auf der Eingangsseite verbessern. Die Hersteller unterscheiden typischerweise zwei verschiedene Modus von PQ: Strict-PQ sowie Rate-Controlled-PQ.

Strict-PQ sichert ab, dass die Queues höherer Priorität immer vor den Queues geringerer Priorität abgehandelt werden. Dabei besteht natürlich immer die Gefahr, dass ein relativ großer Datenstrom dieser Priorität die Datenströme geringerer Priorität ausbremst und sogar ganz »verhungern« lässt. Das kann aber auch durchaus gewünscht sein, wenn zum Beispiel unternehmenswichtige Echtzeitdienste oder Dienste zur Netzwerkwartung auf der hohen Priorität liegen. Diese Dienste müssen dann auf jeden Fall weitergeleitet werden, egal, wie belastet das Netzwerk ist. Strict-PQ verwendet dabei auch Bandbreite für die hohe Priorität, die eigentlich für andere CoS-Klassen reserviert ist. Bei Rate-Controlled-PQ werden Frames aus der Queue mit höherer Priorität nur dann vor denen aus Queues niedrigerer Queues abgearbeitet, wenn die Datenrate der höheren Priorität nicht einen eingestellten Wert überschreitet. Wenn dieser Wert überschritten wird, können Queues mit niedrigerer Priorität vor denen mit der hohen bedient werden. Hier ist der Netzwerkadministrator gefordert, dass er die Mechanismen so konfiguriert, dass das Netzwerk der erforderlichen Bandbreiten-Policy entspricht.

Verteilung der Framegrößen



Analysen der Verteilung der Framegrößen, beispielsweise für das NCI-Backbone oder von den Applikationen her typischer Business-DSL-Links haben ergeben, dass rund 50 Prozent aller Datenrahmen in realen Netzwerken 64 Byte groß sind.

Ausgleichende Gerechtigkeit

Fair-Queuing (FQ) wurde entwickelt, um jedem Datenstrom einen ausgeglichenen, also fairen Zugang zum Netzwerk zu ermöglichen. Dabei werden die eingehenden Frames zuerst vom System anhand des Datenstroms klassifiziert und anschließend in eine Queue eingefügt, die für diesen Datenstrom reserviert ist. Aus den Queues wird dann jeweils ein Frame pro Round-Robin-Durchlauf abgearbeitet. Leere Queues werden dabei übersprungen. Der besondere Vorteil von FQ liegt darin, dass Datenströme, die besonders Burst-reich sind, die Qualität der anderen Datenströme nicht beeinflussen. Wenn ein Datenstrom mehr Bandbreite benötigt, hat das nur Auswirkungen auf die eigene Queue, aber keine auf die anderen Queues. Leider kann FQ nur softwaremäßig umgesetzt werden. Da FQ aber relativ komplex ist, kann es nur bei langsameren Netzwerkelementen eingesetzt werden. Außerdem ist FQ dafür entwickelt worden, allen Datenströmen die gleichen Bandbreiten zu reservieren. Es gibt somit keine Möglichkeit, Datenströme mit verschiedenen Anforderungen an die Bandbreite zu verwalten. Ein weiterer Nachteil liegt darin, dass die Fairness abhängig von der Framegröße ist. Datenströme mit großen Frames können dank FQ mehr Daten pro gesendeten Frame transportieren und erhalten somit eine höhere Bandbreite als Datenströme mit kleinen Framegrößen. Ein weiterer Nachteil bildet sich durch das verwendete Round-Robin. Trifft ein Frame in eine leere Queue, kurz nachdem der Round-Robin-Scheduler diese Queue »besucht« hat, so muss der Frame so lange in der Queue verweilen, bis der Scheduler alle anderen Queues einmal bedient hat. Trifft ein Frame jedoch kurz vor dem Besuch des Schedulers auf diese Queue, wird er sofort bearbeitet. Dadurch lassen sich nur schwer Aussagen über den Delay treffen, und der Jitter ist sehr hoch. Echtzeitanwendungen lassen sich so mit FQ nicht gut übertragen. Ein ganz großer Nachteil liegt darin, dass für jeden Datenstrom eine extra Queue angelegt wird. Würde man FQ nun in einem stark belasteten Switch verwenden, müssten Tausende Queues verwaltet werden. Im so genannten Class-based-FQ wird der Ausgangsport in eine Anzahl von verschiedenen Serviceklassen unterteilt. Jede dieser Serviceklassen allokiert einen definierten Prozentsatz der Bandbreite des Ausgangsports. Auf diese Serviceklassen wird dann jeweils einzeln FQ angewendet. Dadurch kann für jeden Datenstrom innerhalb einer Serviceklasse Fairness erreicht werden. Die einzelnen Serviceklassen selbst sind aber mit unterschiedlichen Bandbreiten von einander verschieden.

Alle sind gleich, aber manche sind gleicher

Weighted-Fair-Queuing (WFQ) ist die Grundlage für Class-of-Service-Scheduling-Strategien. WFQ unterstützt dabei Datenströme mit verschiedenen Bandbreiten, indem es jeder Queue eine Gewichtung zuteilt, die ein Prozentsatz der gesamten Bandbreite des Ausgangsports ist. Außerdem unterstützt WFQ auch Datenströme mit verschieden langen Frames. Dadurch bekommen Datenströme mit langen Frames nicht mehr Bandbreite als Datenströme mit kurzen Frames. Durch die faire Abhandlung von Frames verschiedener Länge ist WFQ relativ komplex. Die Unterstützung für die faire Verteilung von Bandbreite wird beim WFQ durch die Annahme erreicht, dass jeder Prozess (Queue) gleich viel Zeit erhält. Das ist aber nur ein theoretischer Scheduler, der nicht implementiert werden kann. Sein Verhalten ist gleich einem gewichteten Bit-für-Bit-Round-Robin-Scheduling. Bei dieser Scheduling-Strategie wird jedes einzelne Bit vom Beginn einer Queue nach WRR-Manier übertragen. Diese Annäherung erreicht eine faire Verteilung, da sie die Frames in einzelne Bits aufteilt, die dann gleichmäßig übertragen werden. Wenn die Frames nun am anderen Ende des Links wieder zusammengesetzt werden, so ist entscheidend, welches letzte Bit eines Frames zuerst ankommt. Diese »Frame-Ende-Zeit« beeinflusst die Reihenfolge, in der die Frames dann wieder am Ausgangsport abgesendet werden. WFQ nähert sich den theoretischen Aussagen an, indem es für jeden Frame eine »Frame-Ende-Zeit« errechnet. Aus der gegebenen Bandbreite des Ausgangsports, der Anzahl der aktiven Queues, der relativen Gewichtung der einzelnen Queues und der Länge der einzelnen Frames in den Queues wird diese Ende-Zeit für jeden einzelnen ankommenden Frame ermittelt. Der Scheduler bearbeitet dann den Frame mit der kürzesten errechne-

ten Ende-Zeit und leitet diesen weiter. Dabei ist zu beachten, dass die Ende-Zeit keine reale Zeit für die Übertragung ist, sondern vielmehr eine Nummer, die die Reihenfolge der Frames ausdrückt, in der sie abgearbeitet werden sollen. Wenn alle Frames klassifiziert und in der entsprechenden Queue gepuffert sind, errechnet der Scheduler für jeden Frame eine Ende-Zeit. Bei der Abarbeitung der gepufferten Frames bearbeitet der Scheduler zuerst die Frames mit der kürzesten Ende-Zeit. Anschließend wird der Frame mit der nächstgrößeren Ende-Zeit bearbeitet. WFQ hat zwei Vorteile: Einerseits bietet es Schutz für jede Serviceklasse, indem es einen minimalen Anteil der Bandbreite des Ausgangsports für jede Klasse zusichert, unabhängig davon, welche Eigenschaften diese Klasse hat. Anderseits bietet WFQ die Möglichkeit, für gewichtete, faire Anteile der Bandbreite des Ausgangsports einen begrenzten Delay zuzusichern. WFQ hat allerdings auch einige Nachteile. WFQ wird von den Herstellern in Software realisiert, nicht in Hardware. Dadurch stellt WFQ hohe Anforderungen an die zu Grunde liegende Hardware, zumal der Algorithmus des WFQ ohnehin schon sehr rechenintensiv ist. Weiterhin kann bei Verwendung von vielen Serviceklassen ein unregelmäßiger Datenstrom sehr leicht die Leistung anderer Datenströme der gleichen Serviceklasse beschränken. Außerdem ist die Begrenzung des Delays immer noch auf recht große Werte reduziert, auch wenn es schon besser funktioniert als bei anderen Scheduling-Strategien. Da WFQ eine relativ alte Scheduling-Strategie ist, hat es im Laufe der Jahre viele Entwicklungen gegeben, die aus dem eigentlichen WFQ hervorgingen. Diese Entwicklungen sind zum Beispiel Class-Based-WFQ, Self-Clocking-Fair-Queuing (SCFQ), Worst-Case-Fair-Weighted-Fair-Queuing (WF2Q) oder auch Worst-Case-Fair-Weighted-Fair-Queuing+ (WF2Q+).

WFQ ist an den Endpunkten von Netzwerken angesiedelt, um eine faire Verteilung von Bandbreite auf verschiedene Serviceklassen zu sichern. WFQ kann auch auf verschiedene Verhalten eingestellt werden. Die erste Möglichkeit ist, WFQ so einzustellen, dass es eine große Anzahl von Queues verwalten kann. Mit der zweiten Einstellmöglichkeit kann man eine begrenzte Anzahl von Queues, die mehrere Datenströme enthalten, verwalten. Bei dieser Einstellung werden CoS-Regeln oder die drei Prioritätsbits von IP-Type-of-Service verwendet, um Frames den Queues zuzuordnen. Für jede der Queues wird ein bestimmter Prozentsatz der Bandbreite des Ausgangsports allokiert. Der Prozentsatz ist dabei abhängig vom errechneten Gewicht für die Serviceklasse. Diese Näherung erlaubt dem System, mehrere Bandbreiten für jede Queue auf Grund der CoS-Regeln oder auf Grund der ToS-Bits zu reservieren. Eine gehobene Version von WFQ, manchmal als Class-Based-WFQ bezeichnet, kann alternativ eingesetzt werden, um eine begrenzte Anzahl von Queues zu verwalten, die mehrere Datenströme enthalten. In dieser Konfiguration bestimmen benutzerdefinierte Frame-Klassifikationsregeln, welcher Queue Pakete zugewiesen werden. Diese Queues bestimmen einen benutzerkonfigurierten Prozentsatz der Bandbreite des Ausgangsports.

Mittelwege

Weighted-Round-Robin (WRR) ist eine Gruppe von Scheduling-Strategien, die die Nachteile von FQ und PQ beheben sollen. Dabei behebt WRR die Nachteile von FQ, indem es jedem Datenstrom einen eigenen Teil der Bandbreite des Ausgangsports zuweisen kann. Weiterhin wird bei WRR der Nachteil von Strict-PQ behoben, bei dem Frames in Queues mit geringerer Priorität die gesamte Bandbreite sowie der Pufferspeicher vorenthalten werden kann. Bei WRR wird mindestens ein Frame pro Schleifendurchlauf aus jeder Queue entfernt. WRR wird manchmal auch als Class-Based-Queuing (CBQ) bezeichnet. Beim WRR werden die Frames zunächst klassifiziert und in verschiedene Serviceklassen aufgeteilt, zum Beispiel in die Serviceklassen »real-time«, »interactive« oder »filetransfer«. Anschließend werden diese Frames in eine Queue eingefügt, die für die entsprechende Serviceklasse reserviert ist. Jede dieser Queues wird in Round-Robin-Reihenfolge abgearbeitet. Dabei werden, genau wie bei FQ und PQ, leere Queues übersprungen. WRR unterstützt die Reservierung von verschiedenen Bandbreiten für verschiedene Serviceklassen, indem es entweder mehr als ein Frame aus Queues mit höherer Bandbreitereservierung abarbeitet oder, indem es immer nur einen Frame pro Besuch des Schedulers abarbeitet, aber dann die Queue pro Round-Robin-Durchlauf mehrfach besucht wird. Um die Menge der reservierten Netzwerkressourcen für jede Serviceklasse zu regulieren, kann eine Anzahl von Parametern verändert werden. Der Delay eines Frames in einer bestimmten Queue ist abhängig von der Rate, mit der die Frames in die Queue platziert werden, von der Größe der Queue sowie von der Anzahl der Frames, die bei jedem Round Robin Durchlauf abgearbeitet werden, als auch von der Anzahl der anderen Queues, die auf den gleichen Ausgangsport konfiguriert sind. Die Größe des Jitter ist abhängig von der Varianz des Delays in dieser Queue, der Varianz in allen andern Queues und der Varianz der Intervalle zwischen den Round-Robin-Runden. Die Größe des Frame-Loss ist abhängig von einer Kombination aus der Framerate, mit der die Frames in die Queue gelegt werden, der Tiefe der Queue, dem »Random Early Detect (RED)Profile«, das für die Queue verwendet wird, und der Anzahl der Frames, die pro Durchlauf des Schedulers aus der Queue abgearbeitet werden. Die Füllrate kann dabei durch eine Kontrolle der Bandbreite auf der Eingangsseite kontrolliert werden.

WRR bietet eine ganze Anzahl von Vorteilen:

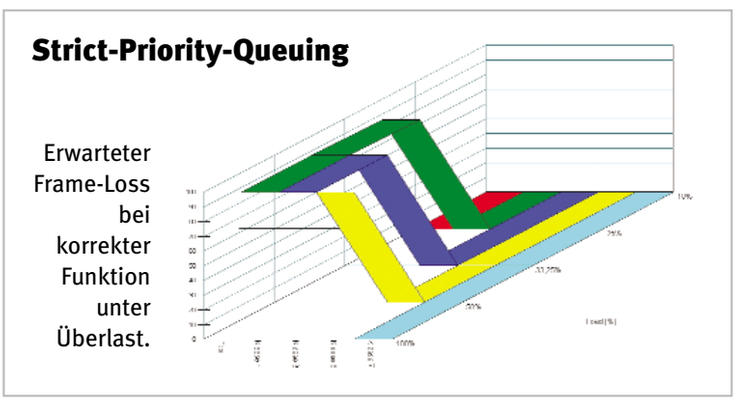
- WRR kann in der Hardware implementiert werden und eignet sich somit besonders für Switches mit sehr hohen Geschwindigkeiten.
- WRR bietet eine grobe Kontrolle von der auf die Serviceklassen verteilten Bandbreite des Ausgangspots.
- WRR stellt sicher, dass zumindest immer ein Teil der eingestellten Bandbreite zur Verfügung steht. Somit verhindert es, dass der Netzwerkverkehr »verhungert«.
- WRR bietet einen effizienten Mechanismus, um den Transport von verschiedenen Serviceklassen auf eine geeignete Anzahl von gebündelten Datenströmen zu unterstützen.
- WRR bietet durch die Klassifizierung des Datenverkehrs in Serviceklassen eine gerechte Verteilung und mehr Stabilität für Netzwerkanwendungen. Will man zum Beispiel Echtzeitanwendungen gemeinsam mit FTP-Daten mittels Strict-PQ verwalten, kann es sein, dass die

FTP-Daten völlig unterdrückt werden, falls der Anteil der Echtzeitdaten entsprechend groß ist. WRR basiert auf dem Gedanken, dass es besser ist, Ressourcen zu reduzieren als Ressourcen zu verbieten, um Verstopfungen zu verhindern. Denn die Unterdrückung von Ressourcen der Klassen mit geringer Priorität behindert auch alle Signale für die Reduktion des Dienstes. Dadurch sind Anwendungen nicht in der Lage, auf der Senderseite den Netzwerkverkehr angemessen zu begrenzen.

Der hauptsächliche Nachteil von WRR liegt darin, dass es nur dann den gewünschten Anteil der Bandbreite pro Serviceklasse verwendet, wenn die Länge der einzelnen Frames gleich oder die durchschnittliche Framegröße bekannt ist. Wenn zum Beispiel drei Queues Frames an den gleichen Ausgangsport senden wollen und die erste Queue 40 Prozent, die zweite 30 Prozent und die dritte 20 Prozent der Bandbreite reserviert haben, ist die Ausgangsverteilung bei gleicher Framelänge gesichert. Wenn man davon ausgeht, dass alle Frames 100 Byte lang sind, ergibt sich, dass nach der ersten Runde des Schedulers von der Queue 1 vier Frames übertragen wurden (400 Byte), von der zweiten drei Frames (300 Byte), von der dritten zwei Frames (200 Byte) und von der vierten ein Frame (100 Byte). Das entspricht also genau den Anforderungen, die an WRR gestellt sind.

Wenn nun aber die Frames der verschiedenen Klassen unterschiedliche Längen haben, kommt es dazu, dass die Klassen mit den längeren Frames mehr Daten übertragen, als ihnen eigentlich zusteht. Geht man beispielsweise davon aus, dass in der Queue 1 die Frames 100 Byte, in der Queue 2 200 Byte, in der Queue 3 300 Byte und in der Queue 4 400 Byte lang sind, erhält man eine ganz andere Verteilung der Bandbreite als eingestellt. Dadurch ergibt sich auch die gleiche Verteilung der Anzahl von Frames nach einem Durchlauf des Schedulers. Jedoch ist die entsprechende Anzahl der übertragenen Bytes völlig anders. Queue 1 überträgt vier Frames (400 Byte), Queue 2 drei Frames (600 Byte), die Queue 3 im gleichen Zeitraum zwei Frames (600 Byte) und die Queue 4 überträgt einen Frame (400 Byte). Bei der resultierenden Gesamtmenge von 2000 Byte ergibt sich nun, dass Queue 1 20 Prozent, Queue 2 30 Prozent, Queue 3 ebenfalls 30 Prozent und Queue 4 20 Prozent der verfügbaren Bandbreite beanspruchen. Das entspricht eindeutig nicht der gewünschten und konfigurierten Verteilung der Bandbreite.

Deficit-Weighted-Round-Robin (DWRR) ist die Grundlage für Scheduling-Strategien, die die Nachteile von WRR und WFQ beheben sollen. Der Nachteil von WRR, bei unterschiedlichen Framelängen keine genaue Verteilung der Bandbreite zu unterstützen, wird behoben. Auch der Nachteil von WFQ in Bezug auf eine zu hohe Komplexität ist beim DWRR nicht mehr vorhanden. Somit ist DWRR auch für sehr schnelle Switches geeignet und bietet eine gute Skalierung der Bandbreite des Ausgangsports auf die einzelnen Queues.



Bei DWRR werden verschiedene Parameter für jede Queue konfiguriert:

- Eine Gewichtung bestimmt den Anteil der Bandbreite des Ausgangsports, der einer Queue zugestanden wird.
- Ein Deficit-Counter, der die Anzahl der Bytes bestimmt, die bei jedem Mal übertragen werden, wenn der Scheduler die entsprechende Queue bedient. Der Deficit-Counter erlaubt einer Queue, die beim letzten Durchlauf nicht bearbeitet werden konnte, da ein Frame am Anfang der Queue größer war als der Wert des Deficit-Counter, eine Art »Gutschrift« zu bekommen, die dann beim nächsten Durchlauf verwendet werden kann.

Da DWRR in Hardware implementiert werden kann, ist es auch für schnelle Switches geeignet. DWRR bietet alle Vorteile von WRR und behebt dessen Problem bei der gerechten Verteilung der Bandbreite bei unterschiedlichen Frame-Längen.

Die Hersteller von Netzwerkkomponenten sind heute bestrebt, das beste Verhältnis zwischen guter Verteilung und einfacher Umsetzung bei der Implementierung der Queuing-Strategien zu finden. Eine mögliche Lösung besteht zum Beispiel darin, dass PQ mit DWRR gemeinsam eingesetzt wird. Diese beiden Techniken können hardwaremäßig umgesetzt werden. Dabei werden höher priorisierte Datenströme immer vor den niedriger priorisierten weitergeleitet. Wenn die hoch priorisierten Ströme ihre Bandbreite überschreiten, so wird dieser Datenstrom auf die reservierte Bandbreite begrenzt. Somit können Datenströme mit geringer Priorität nicht »verhungern«.

Höher priorisierte Datenströme werden nur so lange bevorzugt behandelt, wie sie im Bereich ihrer reservierten Bandbreite bleiben. Innerhalb einer Priorität können Queues dann nach DWRR-Prinzip verwaltet werden, um eine Einhaltung der reservierten Bandbreite zu erreichen. Diese Kombination von Strict-PQ und DWRR ist relativ billig und kann hardwareseitig implementiert werden. Sie erlaubt hoch priorisiertem Datenverkehr, schnell bearbeitet zu werden, was einen geringen Delay ermöglicht, ohne dass Datenströme geringer Priorität »verhungern«.

Fazit

Für die für unseren Switch-Vergleichstest unterstellten Anwendungsgebiete ist eine gleichmäßige, konfigurierbare Verteilung der Prioritäten erwünscht. DWRR bietet sich also als Scheduling-Strategie an, da es sehr fair und auch schnell genug ist, um in Switches zu arbeiten, die Anschlüsse mit Gigabit-Geschwindigkeit haben. Dieses Verfahren erscheint wegen seiner Geschwindigkeit durch hardwaremäßige Implementierung und durch die Art des Schedulings von der Theorie her das am besten geeignete Verfahren zu sein.

Da DWRR recht komplex in der Realisierung ist, kann man es natürlich auch mit einem einfachen WRR ersetzen. Allerdings sollte man in diesem Fall bedenken, dass Qualitätsklassen mit vorwiegend kürzeren Datenframes weniger benachteiligt weitergeleitet werden. Das heißt, wenn zwei Klassen die gleiche Bandbreite zugewiesen bekommen, wird die Klasse mit den längeren Frames mehr resultierende Bandbreite erhalten als die Klasse mit den kürzeren Frames. Der Grund dafür liegt im Verfahren selbst.

Als Ersatz für das DWRR kann man auch eine WFQ-Strategie verwenden. Das Resultat ist dabei das gleiche, nur eignet sich DWRR besser für Switches im Gigabit-Bereich als WFQ. Der Grund dafür liegt in der Implementierung. DWRR kann hardwaremäßig implementiert werden, WFQ aber nur softwaremäßig. Softwaremäßige Implementierungen sind in der Regel aber deutlich langsamer als hardwaremäßige.

Um bei extrem überlasteten Netzen auch eine Qualitätsklasse zu haben, die von den anderen unbeeinflusst geswitcht wird, bietet sich die Zusammenarbeit des DWRR – oder auch WRR beziehungsweise WFQ – mit einer Strict-PQ-Strategie an. So können die Daten zur Netzwerkkontrolle, wie sie in der CoS-Klasse 7 in dem Standard 802.1D gefordert, jederzeit ihr Ziel erreichen. Da vier verschiedene Qualitätsklassen untersucht werden sollten, bietet es sich an, drei Queues mit DWRR zu versehen und eine Queue mit Strict-PQ. Die mit DWRR verwalteten Queues werden für die CoS-Klassen 1, 3 und 5 verwendet. Sie entsprechen Daten für Hintergrundverkehr mit geringer Priorität, Excellent-Effort für hervorragende Übertragung und Videodaten, die in Echtzeit mit höchster Priorität übertragen werden sollten.

In diesem Aufbau für die Scheduling-Strategien können dann verschiedene Gewichtungen eingestellt werden. Dabei sollte die Gewichtung des Strict-PQ nie verletzt werden. Bei Überlast in allen Prioritäten kann es in den anderen Queues theoretisch dazu kommen, dass diese ihre gewünschten Gewichtungen nicht einhalten können.

Soweit zur Theorie – wie sich diese Mechanismen, die in aktuellen Core- wie Edge-Switches implementiert sind, in der Praxis unserer Real-World-Labs-Tests verhalten haben, steht dann in unserem großen LAN-Switch-Vergleichstests ab Seite 14 in diesem Heft.

Dipl.-Ing. (FH) Toralf Runge, Prof. Dr. Bernhard G. Stütz, [dg]