First publ. in: IEEE Computer Society, 32nd IEEE Conference on Local Computer Networks, 2007, pp.246-248
http://doi.ieeecomputersociety.org/10.1109/LCN.2007.46

Light-weight End-to-End QoS as DoS Prevention

Marcel Waldvogel Tobias Köck

***Abstract*—This paper proposes a first step into a common solution, where combined and extended interests will hopefully allow us to surpass this threshold. While there are still some open issues, we hope to not only propose a basic working mechanism but also provide fresh ideas to start thinking off the beaten path. Our main contribution is to create a lightweight, end-to-end binding between path and service, which is then used as a basis to associate further attributes and mechanisms to this binding.**

I. Introduction

Our lightweight scheme–router-assisted, receiver-driven QoS (RarQoS)–allows the parties with vested interest to take action and as a result obtain better quality under heavy load. This not only presents a line of defence against rare events such as DoS, but at the same time can be used to improve QoS, a stronger driving force. The change necessary for the network providers is minimal, their role is essentially limited to only act as a third-party verifier of the sources' claims. All the important decisions remain with the end systems and their users or administrators and it further allows incremental deployment,

II. Background

A. Denial of Service

In early 2000 several resource-rich commercial sites were unreachable for several hours, probably due to the actions of a single individual who previously had gained control over many thousand computers world-wide [2]. This shock resulted in a series of proposals how to prevent future disasters. Since then it was tried to reach consensus on how to improve the situation, but to no avail [3]. We believe that the reasons do partly lie in the form of the proposals, as they address the wrong audience. To set the stage, we first identify five components (Zombies, Start Signal, Attack, Fake Source and Abort) [4] of a DDoS attack.

The manifold approaches at DDoS prevention try to hinder the first four components or improving the countermeasures in the abort phase. We can classify these approaches into seven categories [4]. Additionally there are market issues: Those who should invest money in upgrading their equipment and risk more customer support calls or dropping customer satisfaction, among other things, are frequently not those that have an interest in setting up such a system.

III. Router-assisted, receiver-driven QoS

A. Overview

The design of RarQoS diverges from the established DoS prevention path. It was influenced by QoS ideas instead, noting that preventing DoS is just a special case of handling QoS. But as there is no need to provide QoS guarantees, but just a simple form of differentiating between multiple classes of best effort service, it does not suffer from the complexity and state explosion common to many QoS approaches, such as IntServ [5]. It also does not require establishing a mapping between different QoS parameters and contract negotiations, as necessitated by DiffServ [6].

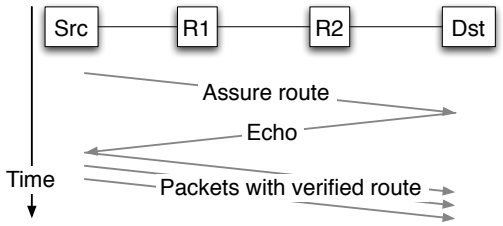

Fig. 1
RarQoS messages: Setup and data flow

B. Basic route recording

Instead of using traditional methods like the IP "record route" option, we are improving it by using a record tuples (*hop count*, *verifier code*) into the designated section of the packet. The verifier code consists of a value derived from flow information (e.g. the address/port/protocol five-tuple) and a secret known only to the issuing router. The derivation function must not be invertible by any other party than the router. Potential functions include keyed hashes or encrypting the flow information with the secret. The process can be further strengthened by having routers set a flag when they recognise a mismatch in the verifier code, as an alert to routers further down the road, that this packet has been tampered with and that it should be forwarded only when there is no congestion ("misbehavior detected bit"). Then, the efforts of all the routers are multiplicative, no longer just additive.

By including the current hop count together with the flow information as an input to the keyed hash or encryption we prevent `traceroute attacks`. This multiplicative effort allows us to limit the number of bits per RarQoS step to one, the impact of an attack will be reduced by a factor of 2^r, with each router using just a single bit. This allows us to get rid of all counters.

C. Path binding properties

To summarise, RarQos path binding properties. It securely binds packets to a path with a high probability. It needs no router storage and requires only minimal packet data.The established session can be piggybacked on existing transport or application layer setup or in a separate protocol and can receive IntServ-style path binding with minimal overhead (i.e., "DiffServ-style"). Additionally it

Distributed Systems Laboratory, Department of Computer and Information Science, University of Konstanz, 78457 Konstanz, Germany, <firstname>.<lastname>@uni-konstanz.de. Work was started at IBM Research, Zurich Research Laboratory [1].

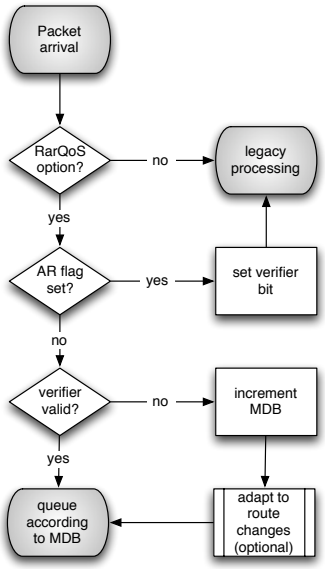

Fig. 2
Router message processing

does not rely on symmetric routing.

IV. Implementation considerations

We are currently implementing a prototype in the Scalable Simulation Framework, SSFNet,[1] to gain experience with the properties, refine open issues, and evaluate further applications.

A simple bidirectional RarQoS handshake could be integrated into TCP's three-way-handshake and the following data transfer [4]. Integration into TCP's setup is not necessary, any application messages or even out-of-band signalling can be used to set up the RarQoS association. On the first message, the source asks the routers to set the bits they would like to see on further messages, such as their path can be validated. The target then echoes these bits back (potentially only after authenticating the source as trustworthy), such that the sender can now use elevated priority. A potential function for the router to use could be

$$b = h(\text{routerSecret} \parallel \text{timeToLivetypeOfService} \parallel \text{sourceAddress} \parallel \text{destinationAddress}), \quad (1)$$

where h is a secure hash function returning a single bit, $\parallel$ is the concatenation operator, *routerSecret* is a per-router secret (no per-host or per-connection state), inclusion of *timeToLive* defeats hop-by-hop verifier secret gaining attempts, and the remaining variables are pieces of information from the packet which should be linked to this path.[2]

Fig. 2 shows the steps necessary in a router to process RarQoS messages, discriminating between legacy IP datagrams, setup and verification messages. The optional route adaptation process is described in Section B.

[1] `http://www.ssfnet.org`

[2] Source and destination ports could be included into the calculation but would prevent a client from reconnecting to a server under DoS.

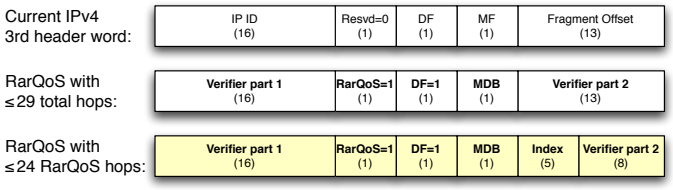

Fig. 3
Possible message format when avoiding IP options

A. To option or not to option

A clean and flexible way to implement RarQoS would be as an IP option (as described in [4]). It would be fully downward compatible for both routers and end-systems without requiring any extra packets. The most important field is the verifier field, which will be used by the source to prove the binding to the users. The introduction of an index field provides a further benefit, namely the opposite: allowing the introduction of virtual RarQoS routers. In the beginning, when only few routers will support RarQoS, only few hops will be verified, clearly not enough to bind paths with only a single bit per router. In this initial phase, a single router can extract several bits from its hash function to act as the local verifier "bit."

If verification of a single bit fails, a router will increment the value stored in the *Misbehaviour Detection Bits*. A single difference may not be an indication of cheating, as a router could have changed its secret key, e.g. through reboot, or a path component might have been replaced. Therefore, we propose that routers will gradually decrease the packet's priority when discrepancies in the verifier become apparent. Also, using the *Re-Probe* bit, a friendly receiver might tell the source that verifier information along the route has changed.

While the implementation as an IP option has several design advantages, as shown above, it has the practical disadvantage that many current routers will process packets with IP options very inefficiently along the "slow path" through the main CPU, bypassing the specialised forwarding hardware. RarQoS, being a co-operative protocol, unlike DDoS packet marking schemes, can use several fields from the stock IPv4 header. Especially convenient is the third word from the header, which contains IP ID, fragmentation flags, and fragmentation offset (Fig. 3). With the pervading use of Path MTU Discovery, many modern operating systems no longer require these fields. Therefore, they can be put to good use. The must-be-zero reserved bit would be modified to become a RarQoS indicator and the Don't Fragment flag would need to be set for legacy routers; the other bits could be used. Again, we have the option of using TTL indexing or putting aside a separate index field, to obtain more flexibility. Even in this space-constrained environment, the index solution shows its advantage.

The fields, as used here, do not provide an option for

establishment of the binding. This has been purposefully chosen, as legacy end-systems might become confused when receiving messages with non-zero fragment offset or must-be-zero bits set to one. In this case, the path would be bound using separate establishment datagrams sent before or with the actual connection-setup. This would be achieved using special ICMP messages or end-to-end packets with the IP option described above. This separation choice not only resolves the "slow-path" issue, it further does degrade gracefully if an overzealous yet RarQoS-ignorant middlebox is in the path.[3]

B. Dynamics options

There are always possibilities for a box to be replaced, a component failing and requiring an automatic or manual replacement, or even the path getting slightly longer or shorter. Another aspects of dynamism includes wilful changes of the router secret. While we expect RarQoS to react very quickly to path changes, especially if routers always update their verifier status in messages, not only in setup messages, the framework allows for the use of more sophisticated mechanisms to overcome small route changes at the local level. These changes will typically cause a single router to be replaced by it's hot standby or an alternative path being chosen, which might be slightly shorter or longer, in terms of hop count.

Our optional mechanism includes a router, which notices a mismatch to the verifier bit and tries to guess whether the path has been shortened or prolonged by a single hop, i.e., verifying whether the previous or next bit would have matched. It records this result in the *Before/After Match* bits, but also possible in Fig. 3). If the next hope finds this hint confirmed, it can set the *Shift/Direction* bits to indicate a single-bit offset from the original plan. Such guesswork slightly weakens the binding; our simulations will show whether this is worth the higher stability of the system.

A simple step to increase stability and which each router can implement independently, is to use a slightly different hash function input (Eq. (1)): Instead of taking the TTL, take the contents of the index field. This will only cause a change, if a RarQoS router in the path changes or is inserterd/deleted, not if a legacy router is inserted into or deleted from the path. When a router purposefully changes its secret to require old bindings to time out, it might also not flag new messages hard, but only set a flag indicating the usefulness of a reprobe to update the binding.

C. QoS properties

While still maintaining DiffServ-style storage and communications overhead, it is now also possible to bind a path to the DiffServ parameters. This allows to reserve resources along the path and potentially also rejecting DiffServ requests by changing the code point or setting a flag in the RarQoS setup message. The path binding not only allows to borrow some IntServ properties into the DiffServ world, it may also offer new options for fraud detection and prevention in general. This system can also be combined with (forward or reverse) charging mechanisms, requiring both parties' ongoing consent and thus making the system more transparent to the end users, combined with lower abuse potential due to the path binding.

[3] Such firewalls have been reported to impede Explicit Congestion Notification (ECN) deployment.

D. Anti-DDoS properties

First of all, it provides a QoS differentiation for accepted users at the receiver and the passing-on of QoS binding credentials to other hosts with a sufficiently different path. These two properties will already significantly reduce the feasibility and effect of a DDoS attack. Yet, it does not prevent the sudden change of an apparently benign user of the system into a DoS attacker. As RarQoS enforces the path binding, at least if the attacker packets should consistently be treated at high priority, it allows the victim to ask upstream routers to install filters which reduce or stop the attack traffic [7]. Unlike other filter systems, it has very low false-positive *and* false-negative rates: (a) The attacker can not easily switch source address, as it would lose the QoS binding and (b) it is ineffective to blame someone else with fake sender addresses and thus causing the victim to install a filter blocking a legitimate communications peer.

V. Conclusions and future work

We described a lightweight scheme where the parties with vested interest need to take action and then obtain better quality under heavy load. The change necessary for the network providers is minimal, their role is essentially limited to only act as a third-party verifier of the sources' claims. All the important decisions remain with the end systems and their users or administrators. It further allows incremental deployment, might be combined with future charging mechanisms, where already a small deployment will show benefits, something which is generally lacking in other approaches.

References

[1] Sean Rooney, Christopher J. Giblin, Marcel Waldvogel, and Paul T. Hurley, "Identifying a distributed denial of service (DDoS) attack within a network and defending against such an attack," European Patent Application EP04405438.5, 2004.

[2] Jelena Mirkovic, Janice Martin, and Peter Reiher, "A taxonomy of ddos attacks and ddos defense mechanisms," Tech. Rep. 020018, Computer Science Department, University of California, Los Angeles, 2002.

[3] Rich Pethia, Alan Paller, and Gene Spafford, "Consensus roadmap for defeating distributed denial of service attacks," `http://www.sans.org/dosstep/roadmap.php`, 2000.

[4] Marcel Waldvogel and Tobias Köck, "Light-weight end-to-end qos as dos prevention," `http://w3.ub.uni-konstanz.de/kops/volltexte/2007/2338/`, June 2007.

[5] Robert Braden, David Clark, and Scott Shenker, "Integrated services in the Internet architecture: An overview," Internet RFC 1633, June 1994.

[6] Steven Blake, David Black, Mark A. Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss, "An architecture for differentiated services," Internet RFC 2475, Dec. 1998.

[7] John Ioannidis and Steven M. Bellovin, "Implementing pushback: Router-based defense against DDoS attacks," in *Proceedings of Network and Distributed System Security Symposium*, Reston, VA, USA, Feb. 2002, The Internet Society.