A Content-oriented Networking Approach for Security and Accountability

Ted ”Taekyoung” Kwon*
Seoul National University, Seoul, Korea
Phone: 732-668-6015 (USA)
Email: tkkwon@snu.ac.kr or tkkwon@winlab.rutgers.edu

I. Introduction

Recently, there are burgeoning efforts to redesign the Internet architecture to solve many problems, to name a few, security, mobility, routing scalability, accountability (e.g. [1], [2]). One of the noticeable approaches is the content-oriented networking (e.g. [3]–[5]) as opposed to the original host-based (or address-based) Internet design. The rationale behind the content-oriented network designs is that users of the majority of the Internet traffic (e.g. video on demand, peer-to-peer, CDN) are oblivious to the addresses of the serving hosts.

The motivation of this work is that the content-based networking can be exploited to enhance the security and accountability. First, the most of the security attacks rely on the core Internet transport mechanism that allows a host to send an arbitrary number of packets to any destination. And often the source address of a packet can be spoofed. This mechanism makes it extremely difficult to block the malicious packets, to trace back the origin of offending packets, or to find out who is accountable for Internet anomalies. Second, we rely on the name (e.g. http URI) to check the trustworthiness of the content. However, e.g. by subverting the DNS entry, an HTTP GET request can be forwarded to a wrong site. In general, we have no trustable measures to authenticate the downloaded contents, which makes the Internet vulnerable to phishing and pharming. Third, when a flash crowd to a popular content happens, the corresponding server may show severe degradation. The reason of the above problems is that the network just forwards the packets since it is unaware of the content context.

In this study, we take a clean slate approach to address the security and accountability issues. We first propose a content-based networking architecture in which a user (or its host) exchanges the content files with its ISP, and ISPs exchange the content files among themselves, rather than merely forwarding IP packets. Then we explain how the proposed content-based networking can solve or mitigate the security and accountability problems. Finally, we will discuss the implementation and prototyping issues on top of the current TCP/IP protocol suite.

II. Content-oriented Networking Assumptions

A. Host Identifiers

We assume that every host has its own human-readable identifier, a domain name. Its identifier is registered with its access router, which provides the internet connectivity to the hosts attached to the router’s subnet. Henceforth, we call the access router of a host “an agent.” The reason is that an agent will solicit the contents on behalf of its hosts instead of blindly forwarding IP packets from the hosts. An agent has a domain name and a globally-routable IP address on the egress link. A publisher (or a server) that holds content files has its domain name, and its agent’s IP address is registered to a mapping infrastructure (say, the DNS). Simply put, agents can communicate with each other by the current IP protocol.

B. Content Identifiers

A content is named (or specified) by an HTTP URI[1]. Its publisher (actually, the agent of the publisher) can be located by sending a DNS query. (Note that an agent will send the DNS query on behalf of its host; in this way, a host cannot contact the DNS infrastructure, which helps fortify the security.) Then the DNS will reply with the corresponding entry that contains the IP address of the agent to which the publisher is attached. If the publisher wishes to ensure the authenticity of the content file, she can attach her digital signature and her certificate, to be detailed later.

C. Internet connectivity

When a host first tries to set up Internet connectivity, it first receives an agent advertisement message (similar to a router advertisement message) from an agent. An agent advertisement message includes the domain name of the agent and its certificate as well as the network configuration information. Hence, a host can verify the identities (and their authenticity) of agents[2]. As a content file can be large, it should be divided into multiple IP packets. To deliver IP packets between the agent and its host, we assume that the host can configure some locally unique IP address for communications with the agent (e.g. DHCP), which could be a private address in IPv4 or a link local address defined in IPv6.

* Prof. Kwon is now with WINLAB at Rutgers University on his sabbatical leave from Seoul National University.

[1] As more and more internet traffic is delivered over HTTP [6], we consider web contents only in this study.

[2] Especially in wireless environments, there can be multiple agents.

III. Content-oriented Networking Architecture

A. User-ISP interaction

After the user's host sets up local connectivity with the agent, the host will send an HTTP GET message to the agent to request a content. The agent will first check its cache to find out whether the requested content is stored locally. If not, it will contact the DNS to find out the IP address of the agent of the publisher. The host's agent will contact the the publisher's agent to download the content. At this moment, let us assume both agents are in the same domain.

When the agent (of the host) finishes downloading the content, it forwards the content to the host. Depending on its policy, it may cache the content or not. In this way, if there is a flash crowd for a particular HTTP URI, the agents of the requesting hosts can directly return the content efficiently.

B. ISP-ISP interaction

let us describe the scenario when the agent of the content requesting host and the agent of the publisher belong to different ISPs. Now the border routers that connect the two ISPs are called gateways due to their functionalities other than packet forwarding. As similar to User-ISP interactions, the gateways will deliver the contents between each other.

On receipt of the content request for a publisher outside the ISP, the agent looks up to a routing server[3] to figure out the gateway connected to the neighbor ISP toward the publisher. Denote the gateway selected by the routing server in the given ISP and the next hop gateway in the neighbor ISP by G_s and G_n, respectively. So G_s will send the content request message to G_n. If the publisher belongs to the same ISP as the G_n, G_n will contact the agent of the publisher to download the content, which in turn is relayed to G_s. If the publisher does not belong to G_n's ISP, G_s will contact its own routing server and perform the same procedure.

There will be a lot of contents downloaded in parallel over the link between two gateways (of the adjacent ISPs). For traffic monitoring and accountability purposes, we propose to prepend a short-term flow label (similar to that of MPLS) in front of the IP packets. There is a one-to-one correspondence between the content and the label. In this way, each ISP can keep track of which contents are delivered and how long is each content and so forth.

Like agents, gateways can also store the contents by their own policy[4]. Thus, if the content request message results in a cache hit, the gateway can directly send the content without further forwarding the request.

IV. Security Measures

A. DDoS attacks

Suppose a large number of bots across multiple ISPs are activated to launch an attack to a particular publisher[5]. Denote the agent and the ISP of the publisher under a DDoS attack by A_t and I_t, respectively.

Once the DDoS attack starts, A_t will first detect the storming requests to the target publisher and inform all the gateways of I_t by some intra-domain signaling mechanism (e.g. iBGP can be extended for this purpose). Denote a particular gateway of I_t by G_t. On receipt of DDoS notification, G_t performs two tasks: (i) it starts checking incoming flows onto the attacked publisher from the counterpart gateway of the neighbor ISP, say G_n, and (ii) it solicits G_n to limit the content request rate to the publisher[6]. This rate limiting in terms of content request rate can be easily checked at gateways and propagated across multiple ISP boundaries. In this way, the DDoS attack traffic will be mitigated by a backpressure mechanism.

B. Trustworthiness

When a user downloads a content from a publisher over the Internet, she normally verifies the authenticity of the content by looking at the its name (e.g. the domain name in the HTTP URI). However, the content may not be transmitted from the one which the user wishes to download from. For instance, if the DNS cache entry for the publisher is "poisoned" (i.e. the DNS cache returns a different IP address from the authentic one), the HTTP GET request will bring back a wrong web page. This may lead to phishing. Even without DNS poisoning, a web site forgery can happen in the form of a man-in-the-middle attack.

In order to address the trustworthiness issue, many proposals (e.g. [3], [8], [11]) have been studied in the literature. Here, we take a practical approach – the self-certifying content. So, the content means the content file itself along with the metadata for authentication, which includes the publisher's signature on the content as well as the publisher's certificate. For simplicity, we assume that a host can keep most of the popular certificates and hence it can easily verify the hierarchical chain of certificates[7]. If needed, the host can download a certificate just by using the proposed content-networking mechanism.

V. Implementation and Performance Issues

In this section, we focus on how to substantiate a DDoS countermeasure scenario among ISPs on the testbed made up of OpenFlow switches with NetFPGA cards. In this scenario, ISPs will react to DDoS attacks in a cooperative fashion by

[3] Looking at the domain name of the publisher, a routing server (e.g. a BGP router with some extension) returns the the IP address of the gateway within the ISP toward the publisher. If there are multiple candidate neighbor ISPs for the publisher, some policy (similar to the BGP policy) is needed to select a gateway connected to the best neighbor ISP, which is out of the scope.

[4] We believe that an ISP can orchestrate the caching policy over its agents and gateways (e.g. [10]).

[5] If every agent allows the transport of contents of well known services only, bots might not be activated. However, we conservatively assume that there are vulnerable points in distributed environments; for example, agents can be compromised, or commands to bots can be spread by other mechanisms.

[6] As for (ii), we can think of a passive approach (e.g. [7]) that simply discards the DDoS attack traffic locally (at G_t or some local clearing center [9]) without cross-domain signaling; however, we believe that some active approach across multiple ISPs can effectively solve these resource-exhaustive attacks.

[7] Gateways and agents can also check the authenticity; however, we try to place as much processing overhead as possible on the hosts

regulating the content request rate toward a target publisher. Also, for the purpose of accountability, each ISP will keep track of what contents are requested and delivered.

First, we need to embody the content-aware transport mechanism while retaining much of the the current TCP/IP protocol suite. To this end, we use a flow label prepended to every IP packet per content. As for the link between two adjacent gateways (connecting two different ISPs), we assume that short-term labels are readily managed between two gateways. In this way, two gateways connecting two ISPs can count and track the content files for each publisher, their sizes and so on. For traffic prioritization or traffic engineering purposes, we may need to use flow labels inside an ISP. As to intra-domain label-related signaling in the testbed, we will leverage the NOX[8] [12].

The second issue is how to make agents and gateways figure out what content is requested. There are two choices: (i) we can make a host and intermediate nodes[9] to construct a new content request message and send to the next hop node, and (ii) we can extract the HTTP URI from an ordinary IP packet containing an HTTP GET message. Here, we take the latter approach to minimize the modifications at hosts. So, OpenFlow switches (serve as agents or gateways) should be able to inspect the packets and extract the HTTP URI. Fortunately, there is a similar effort [13], which we will adopt in prototyping. However, unlike [13], OpenFlow switches hash the HTTP URI and search the cache by the hashed value.

The third issue is how to implement the signaling to react to the onslaught of DDoS attack packets. There are two signaling functionalities needed for the countermeasure: intra-domain signaling and inter-domain signaling. The inter-domain signaling is easy. The two neighbor gateways need to have some connectivity for notification of DDoS events and rate limiting. However, intra-domain signaling requires all the gateways to be involved. Again, we rely on the NOX for intra-domain signaling over the testbed[10]. We will make a hub-and-spoke signaling connectivity among agents and gateways, where the NOX serves as the hub. In this way, intra-domain signaling for DDoS can go to the NOX first, and then can be distributed to relevant entities.

The last issue is the delay performance. In the content-based networking architecture, the agents and gateways along the route from the publisher to the host relay the contents on a hop-by-hop basis, which results in significant delay. To reduce the content transfer delay, we use parallelism in content transfer. For instance, while the packets of a content is transmitted from gateway A to gateway B (A is the upstream node), gateway B will start transferring the packets to its downstream node.

VI. Concluding Remarks

In this extended abstract, we proposed a novel content-oriented networking architecture in which users exchange contents with agents rather than sending any packets to any destination. The proposed content-oriented networking can react to DDoS attack traffic effectively and provide content authenticity. In addition, due to traffic monitoring in terms of contents between ISPs, accountability is enhanced. Also, we discussed how the proposed architecture can be implemented using OpenFlow switches with NetFPGA cards while retaining most of the operations in the current TCP/IP protocol architecture.

References

[1] NSF FIND Project, http://www.nets-find.net/

[2] EU FIRE Project, http://cordis.europa.eu/fp7/ict/fire/

[3] Teemu Koponen et al., “A Data-Oriented (and Beyond) Network Architecture,” ACM SIGCOMM 2007.

[4] Van Jacobson et al., “Networking named content,” ACM CONEXT 2009.

[5] Sanjoy Paul et al., “The Cache-And-Forward Network Architecture for Efficient Mobile Content Delivery Services in the Future Internet,” Proceedings of the First ITU-T Kaleidoscope Academic Conference on Innovations in NGN: Future Network and Services, 2008

[6] Craig Labovitz et al., NANOG 47, “2009 Internet Observatory Report,” Oct. 2009.

[7] Zhenhai Duan et al., “Constructing Inter-Domain Packet Filters to Control IP Spoofing Based on BGP Updates,” IEEE INFOCOM 2006.

[8] John Kubiatowicz et al., “OceanStore: An Architecture for Global-Scale Persistent Storage,” ASPLOS 2000.

[9] Sharad Agarwal et al., “DDoS Mitigation via Regional Cleaning Centers,” Sprint Technical Report RR04-ATL-013177, 2004.

[10] Jeffrey Erman et al., “Network-Aware Forward Caching,” WWW 2009.

[11] Diana Smetters and Van Jacobson, “Securing Network Content,” PARC Technical Report, 2009.

[12] Natasha Gude et al., “NOX: Towards an Opearting System for Networks,” ACM CCR, July 2008.

[13] Michael Ciesla et al., “URL Extraction on the NetFPGA Reference Router,” NetFPGA Developers Workshop, 2009.

[14] Ericsson Research, Project OpenFlow-MPLS, http://www.openflowswitch.org/wk/index.php/OpenFlowMPLS

PLACE
PHOTO
HERE

Ted “Taekyoung” Kwon is an associate professor at Seoul National University (SNU). Before joining SNU in 2004, he was the post-doctoral researcher at UCLA and CUNY. He got B.S, M.S., Ph.D. at SNU in 1993, 1995, 2000, respectively. He is a founding member of Future Internet Forum (http://fif.kr) in Korea, which was founded in 2006 to coordinate R&D efforts on the future Internet architecture in Korea. Since 2007, he has participated in a few future internet projects funded by Korean funding agencies. He is also a steering group member in Asia Future Internet (http://www.asiafi.net). He also hosted the NetFPGA workshop in Korea in Feb. 2009 at SNU. Recently, he serves as the co-chair of AsiaFI Winter School 2010 and Workshop on Identifiers in Future Internet 2010. His group has achieved the second best demo award in ACM MobiCom 2007 and the third place student best paper in IEEE CCNC 2006.

[8] NOX stands for network operating system; however, the NOX is essentially a central controller of a network.

[9] Nodes include agents and gateways.

[10] Even though the signaling for DDoS notification is not so relevant to original NOX functionalities (e.g. flow management), we decide to co-locate the DDoS signaling functional module in the NOX controller for simplicity.