Computer Communications 106 (2017) 75–85

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Design and analysis of optimization algorithms to minimize cryptographic processing in BGP security protocols

Vinay K. Sriram[a,1,*], Doug Montgomery[b,*]

[a] Stanford University, Stanford, CA, USA
[b] National Institute of Standards and Technology, Gaithersburg, MD, USA

ARTICLE INFO

Article history:
Received 17 July 2016
Revised 4 March 2017
Accepted 20 March 2017
Available online 24 March 2017

Keywords:
Routing security
Border Gateway Protocol security
BGPSEC
Performance optimization
Route processor efficiency

ABSTRACT

The Internet is subject to attacks due to vulnerabilities in its routing protocols. One proposed approach to attain greater security is to cryptographically protect network reachability announcements exchanged between Border Gateway Protocol (BGP) routers. This study proposes and evaluates the performance and efficiency of various optimization algorithms for validation of digitally signed BGP updates. In particular, this investigation focuses on the BGPSEC (BGP with SECurity extensions) protocol, currently under consideration for standardization in the Internet Engineering Task Force. We analyze three basic BGPSEC update processing algorithms: Unoptimized, Cache Common Segments (CCS) optimization, and Best Path Only (BPO) optimization. We further propose and study cache management schemes to be used in conjunction with the CCS and BPO algorithms. The performance metrics used in the analyses are: (1) routing table convergence time after BGPSEC peering reset or router reboot events and (2) peak-second signature verification workload. Both analytical modeling and detailed trace-driven simulation were performed. Results show that the BPO algorithm is 330% to 628% faster than the unoptimized algorithm for routing table convergence in a typical Internet core-facing provider edge router.

1. Introduction

A brief review of vulnerabilities of the Border Gateway Protocol (BGP), the problem statement and a summary of our results are presented in this section.

1.1. Border Gateway Protocol vulnerabilities

Tens of thousands of Autonomous Systems (ASes) in the Internet use Border Gateway Protocol (BGP) [19,24] to convey reachability information for Internet Protocol (IP) prefixes. When BGP was first developed, the main design goal was scalability; little consideration was given to security. One of the most significant security vulnerabilities in BGP is the ability of malicious or misconfigured BGP routers to falsely announce a prefix, and attract the traffic destined for that prefix away from its legitimate destination [2,13]. This is known as prefix hijacking, and results in a denial of service (DoS) for the legitimate prefix owner. A recent example of this is the YouTube subprefix hijack by Pakistan Telecom,[2] described below in Fig. 1a. In this incident, the perpetrators leveraged the fact that BGPs path selection algorithm always prefers longer, more specific, prefixes over shorter, less specific, announcements. An attacker AS might also perform a DoS attack by falsely announcing a connection to an AS with which it is not actually peering, in order to intercept data packets intended for a prefix which is legitimately originated by the victim AS (Fig. 1b). This is one type of man-in-the-middle (MITM) attack on routing [6]. In an MITM attack, the attacker manipulates a BGP announcement so as to attract traffic towards its AS. Once diverted, an MITM attacker may eavesdrop on, or deny, the target data traffic. The goal might be as simple as disrupting established peering business relationships.

1.2. Problem statement and summary of results

Over the years, several security enhancements have been proposed for BGP [1,14,23] to mitigate these types of attacks. One specific example of such an enhancement is BGPSEC (BGP with SECurity extensions), which is currently under development in the

* Corresponding authors.
E-mail addresses: vsriram@stanford.edu, vsriram25@gmail.com (V.K. Sriram), dougm@nist.gov (D. Montgomery).

[1] The work documented here was performed while Vinay Sriram was an intern at the National Institute of Standards and Technology.

[2] All references to specific products, services, and well known Internet incidents are provided for descriptive purposes only and are not to be interpreted as conveying any endorsement, or non-endorsement, of the parties or products involved.

http://dx.doi.org/10.1016/j.comcom.2017.03.007
0140-3664/© 2017 Published by Elsevier B.V.

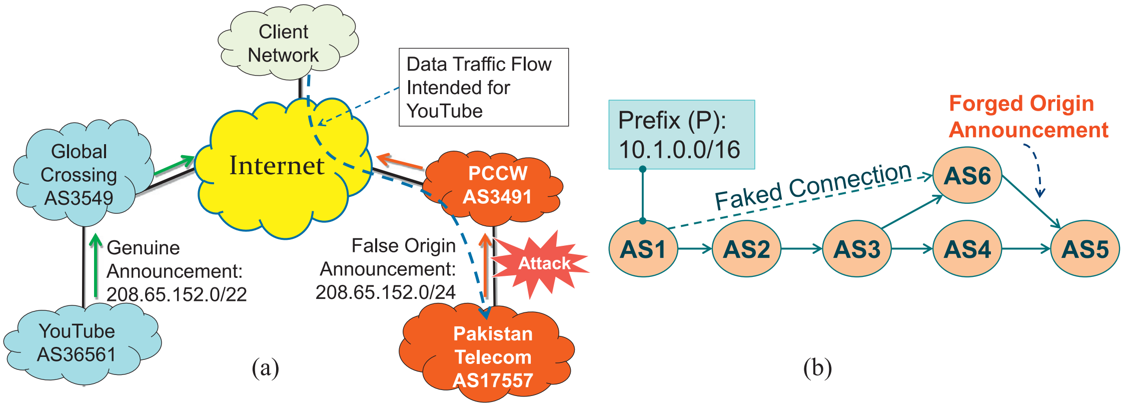

Fig. 1. (a) Pakistan Telecom's false-origin announcement resulted in a DoS attack on Youtube for over 2 h in February 2008. (b) Illustration of an MITM attack, where attacker (AS6) can stealthily attract or eavesdrop on victim's (prefix P at AS1) data traffic.

Internet Engineering Task Force (IETF) [1]. One issue that invites particular attention regarding the BGPSEC protocol is its processing costs (or CPU workload). The cryptographic requirements of BGPSEC would impose a significantly increased workload on the route processors as compared to current BGP. This increased processing cost may prove to be an impediment to near-term deployment of BGPSEC. However, there are ways to optimize cryptographic processing of BGPSEC updates. To date, some optimization algorithms for BGPSEC have been informally discussed in the IETF, but they have not been formally documented. In this paper, we formally describe these algorithms, and also propose new algorithms that are significant enhancements over the previously known algorithms. We present detailed simulation and analytical modeling of these algorithms. We study the performance of the optimization algorithms, to both quantify their relative efficiencies and determine which algorithm is most efficient in terms of CPU workload. We propose and analyze three different optimization algorithms: Unoptimized, Cache Common Segments (CCS), and Best Path Only (BPO). We then extend these algorithms to examine the detailed cache management schemes necessary for their implementation. The performance metrics used for the comparisons are: (1) routing table convergence time after BGPSEC peering reset or router reboot events[3] and (2) peak-second signature verification workload. Both analytical modeling and detailed trace-driven simulation were performed. The results show that the BPO algorithm is 3.3–6.3 times faster than the unoptimized algorithm for routing table convergence in a typical Internet core-facing BGPSEC-enabled provider edge router. Comparisons based on a peak-second workload metric indicate that BPO with extended cache (BPO-EC) algorithm reduces peak CPU workload by about 10 times as compared to the unoptimized method.

1.3. Prior work: description of cryptographically enhanced BGP

The first step towards securing BGP is to ensure that only explicitly authorized ASes can originate specific prefixes. The IETF is nearing completion of a standardized approach to prefix-origin validation, based on a Resource Public Key Infrastructure (RPKI) [10,11,16]. In this scheme, every prefix owner receives a digital certificate, and must register in the RPKI a digitally signed object called a Route Origin Authorization (ROA) [11]. The ROA authoritatively asserts that the AS listed in the ROA can legitimately originate a prefix or a set of prefixes. The RPKI-aware BGP routers in the Internet can use ROAs to validate the route-announcements, and thus provide protection against false origination.

While ROAs alone can mitigate simple attacks and misconfigurations, Fig. 1b shows that it is trivial to forge a valid origin and still misdirect traffic. It is evident from Fig. 1b that in spite of a ROA for the pair $\{P, AS_1\}$, AS_5 would trust a forged-origin announcement from AS_6, and will send traffic intended for prefix P via AS_6. The BGPSEC protocol builds on prefix-origin validation, and extends the security to AS path validation as well [1]. The system of AS hop-by-hop cryptographic signing and verification in BGPSEC prevents prefix hijacks, subprefix hijacks, and MITM path modification attacks (Section 1.1).

In one possible implementation of BGPSEC, each AS, and therefore each external BGP (eBGP) router within the AS, is assigned four digital certificates, each with a {public, private} key pair [5]. Two of the certificates are "current" and "next" originating certificates, meant for signing prefix-updates that are originated by the AS. The other two are "current" and "next" transit certificates, meant for signing prefix-updates transited by the AS [5]. The "current" certificate is used for signing updates, and the "next" certificate is kept in reserve. If and when a key or certificate rollover takes place, then the "next" certificate becomes "current" and a new "next" certificate is generated [5]. Key rollover, described in Section 1.3.3, is a technique used for BGPSEC update freshness and replay protection. Brief descriptions of the signing and verification processes in BGPSEC follow.

1.3.1. Signing process

Here, with the help of Fig. 2, we explain the principles of signing and verification in BGPSEC that are relevant to this optimization study. Other details of BGPSEC can be found in [1]. In Fig. 2 the originating eBGP router in AS_1 first runs a hash function (e.g., SHA-256 [3]) over specific attributes and information in the BGP message $\{P, AS_1, AS_2\}$ to obtain a hash value. The router then uses its current origination private key to produce a cryptographic signature, Sig_{12}, over the hash value using the ECDSA-P256 signature algorithm [3,23]. The BGPSEC update propagated from AS_1 to AS_2 includes: P, $[AS_1, SKI_1]$, $[Sig_{12}]$. SKI_1 represents AS_1's Subject Key Identifier (SKI), the purpose of which is explained in the subsequent section. In general, AS_n (for $n \geq 2$) in the path signs a transmitted update over the following fields:

$$P, [AS_1, SKI_1, AS_2, SKI_2, \ldots, AS_{n-1}, SKI_{n-1}, AS_n, SKI_n, AS_{n+1}], [Sig_{12}, Sig_{23}, \ldots, Sig_{n-1,n}] \quad (1)$$

[3] In this study, a router reboot is simply the simultaneous reset of all peering sessions.