Taxing the Queue: Hindering Middleboxes from Unauthorized Large-Scale Traffic Relaying

AbdelRahman Abdou, Ashraf Matrawy, and Paul C. van Oorschot
Carleton University

***Abstract*—When employed by online content providers, access-control policies can be evaded whenever clients masquerade behind a middlebox (MB) that meets the policies. An MB, commonly being the gateway of a virtual private network (VPN), typically contacts the content provider on behalf of the clients it colludes with, and relays the provider's outbound traffic to those clients. We propose a solution to hinder MBs from unauthorized relaying of traffic to a large number of clients. To the best of our knowledge, this is the first work to address this problem. Our solution increases the cost of collusion by leveraging client puzzles in a novel way, and uses network properties to help the content provider detect if its outbound traffic is being further relayed beyond a transport-layer connection. Our evaluation shows that the number of colluding clients follows a hyperbolic decay with the rate of creation of puzzles and the time required to solve a puzzle—both factors are influenced by the content provider, but grows almost linearly with the MB's computational resources.**

I. Introduction

ONLINE content providers, such as Hulu (hulu.com), often have access-control policies, which either customize or prevent content-delivery to certain classes of clients. By *client*, we mean the software used to communicate with the content provider, e.g., a web browser. For instance, an access policy may only allow access to clients within 300 km of where the site is hosted (e.g, for data sovereignty [1]), or to those with certain IP addresses [2]. Another policy may ban clients at a specific geographic location [3], [4], or clients whose devices have certain system fingerprints (operating system, user-agent, etc) [5]. A content provider (or *provider* for short) may also classify clients by their access networks [6], or their network distance from the server (in terms of hop counts, network latency, etc) [7].

When access policies are in effect, the motivation to bypass them may arise. A client, which does not meet the access policies, may try to bypass them using a *middlebox* (MB) that meets those policies. MBs are commonly transport-layer proxy servers, gateways of virtual private networks (VPNs) or anonymizing networks. The MB requests the provider's content and grants the client access to it by simply relaying the provider's outbound traffic. Many MBs claim to own thousands of IP addresses, which makes blocking them by enumerating their IP addresses almost infeasible. To detect an intercepting MB, a provider can collaborate with a *cooperative* client [8]. However, this is infeasible within our threat model as we address a client that aims to bypass the provider's access policies; i.e., the client is the provider's adversary. Solutions that aim to prevent MBs from intercepting a connection (such as Secure Socket Layer [9]) fail to prevent those MBs from relaying traffic because the client would be ready to share cryptographic credentials, such as encryption keys, with the MB to deceive the provider.

We propose to use client puzzles [10] to increase the cost of collusion per client on the MB. Our solution leverages network properties (average latency between network hosts) which, together with the puzzles, impose a *limit* on the number of simultaneous clients an MB can collude with. Exceeding the limit divulges the MB's relaying actions to the provider. We make the following contributions:

- We propose and study a solution that uses client puzzles to limit unauthorized traffic relaying (§II).
- We use a Markovian queueing model to evaluate our solution, and to find the upper limit of the number of clients the MB can collude with at a time (§III).
- We evaluate the rates of false rejects and false accepts through simulations [11].

II. Hindering Unauthorized Middlebox Relaying

Our objective is to enable a provider detect if a *content recipient*[1] is a *legitimate client* (i.e., connected to the provider without an MB and not relaying the provider's traffic anywhere else) or an MB. To achieve this objective, we use client puzzles [10] to increase the computation required by the MB per client; thus, increasing the round-trip time (RTT) the provider observes. The success of detecting an MB is dependent on the number of simultaneous clients receiving the relayed traffic from the MB. As the number increases, the detection success increases. If the number of clients reaches a certain threshold (§III), the provider realizes that the MB is relaying its traffic. The provider is assumed to be able to:

- Estimate an approximation to the average RTT from itself to a content recipient [12], [13]. Because wireless access networks have unique latency-estimation issues, they are beyond the scope of this letter.
- Estimate the mean time to solve a puzzle with certain difficulty across different client machines spanning a range of computational power (demonstrated in [10]).

For each connection made to provider w from content recipient d, w estimates $N_w(d)$, which is the average network RTT from itself to d. The provider w then periodically creates non-parallelizable puzzles [14], and sends them to d. To solve

[1]We use this term to refer to the machine intended by the provider as the final content destination.

a puzzle, d must allocate a portion of its resources for some time depending on the puzzle difficulty set by w. The resource demanded by the puzzle depends on the type chosen by w, which could be processing- [10] or memory-type [15] puzzles. We assume w uses processing-type puzzles throughout this letter. However, any type can be chosen as long as w is able to estimate the client's puzzle-solving time to some degree of certainty (second assumption above). Upon solving a puzzle, d is required to return the solution to w, which verifies it and bans d if the solution was incorrect. Verification happens in constant time independent of the puzzle difficulty [10].

Denoting t_c as the mean time to solve a puzzle across various clients, w expects to see a RTT of:

$$\text{RTT}_e = N_w(d) + t_c \tag{1}$$

When w receives a solution, it calculates the actual round-trip time, RTT_a, from the puzzle-arrival time and compares it with RTT_e. If $\text{RTT}_a \leq \text{RTT}_e$, the provider assumes that d is not an MB. Otherwise, it suspects that d is an MB because the existence of an MB between the provider and a client is likely to increase RTT_a—an explanation follows.

If d is an MB, it has two options: either relaying all of w's outbound traffic including the puzzles to client c, so that c solves them; or extracting the puzzles from the traffic and solving them on behalf of c. Relaying the puzzles to c costs an additional network RTT, $N_{\text{MB}}(c)$, between the MB and c. An analogous effect occurs if the puzzles were outsourced to a remote party. The actual RTT then becomes:

$$\text{RTT}_a = N_w(d) + N_{\text{MB}}(c) + t_c \tag{2}$$

We do not expect w to be able to estimate $N_{\text{MB}}(c)$. To satisfy $\text{RTT}_a \leq \text{RTT}_e$, the MB and c have to satisfy $N_{\text{MB}}(c)+t_c \leq t_c$, which happens when $N_{\text{MB}}(c) = 0$; that is, the colluding client and the MB are one physical machine, or very close to each other. We believe it is not a cost effective (scalable) attack for an MB to be close to a meaningful number of clients. Assuming proper estimations to t_c and $N_w(d)$ (i.e., RTT_e), it would be challenging for the MB to relay the puzzles to c, and satisfy $\text{RTT}_a \leq \text{RTT}_e$. We study the effect of inappropriate estimation of RTT_e in §III-A below.

To avoid the additional $N_{\text{MB}}(c)$, the MB will be inclined to choose the second option: solve the puzzles on behalf of the clients. An additional queueing time, q, is expected to contribute to RTT_a because the MB will solve many puzzles, which correspond to the number of clients it simultaneously colludes with. The actual RTT would then be:

$$\text{RTT}_a = N_w(d) + q + t_{\text{MB}} \tag{3}$$

where t_{MB} is the MB puzzle-solving time. Recall, the content recipient d is the MB. Again, we do not expect w to be able to estimate t_{MB}. To maintain $\text{RTT}_a \leq \text{RTT}_e$, the MB's computational resources must satisfy:

$$W \leq t_c \tag{4}$$

where $W = q+t_{\text{MB}}$, which is the average time a puzzle spends at the MB from the moment it arrives unsolved to the MB until it departs the MB solved. The queueing time q is affected by: the rate at which w sends puzzles to each client connection; the number of clients simultaneously colluding with the MB; the MB's processing capabilities; and the puzzles' difficulty. The last two factors also affect t_{MB}. Although this option seems more appealing to the MB than the previous one, it forces the MB to limit the number of simultaneous clients to avoid being caught by the provider.

If an MB chooses to combine both options, solving some puzzles by itself and relaying others, the provider will likely observe larger RTT for the relayed puzzles and hence reject the client. The provider may allow some proportion, τ, of RTTs to be larger than the expected RTT before rejecting a client to account for delay spikes. In such case, the benefit of relaying some puzzles will be limited by the provider's parametrization, which upper bounds the proportion of puzzles the MB can relay, without getting its clients rejected, by τ.

III. Evaluation and Analysis

In this section, we derive W (§II) as a function of the parameters affecting it, and analyze the maximum number of simultaneous client connections an MB can collude with (i.e., relay content to) to maintain W that satisfies Inequality 4. We use the following set of notations:

- n—the number of clients simultaneously *colluding with* (i.e., being relayed the provider's content from) the MB.
- t—(t_c in §II) the mean of an exponential distribution representing the time required to solve a single puzzle across different client machines, measured in seconds/puzzle. The provider is required to estimate this mean according to the chosen puzzle difficulty.
- r—the rate the provider generates puzzles to each client connection, measured in puzzles/second.
- b—the proportion of a client's time available to solve puzzles;[2] $b = rt$. If $b = 1$, the average client spends all of its time solving puzzles.
- k—the number of distinct puzzles the MB can solve simultaneously. It is possibly influenced by the number of available processing cores to the MB.
- g—the factor by which an MB processing core is faster than the average client. It is possibly influenced by the cores' clock rate.

We focus only on the MB's processing power (k and g) as needed to solve processing-type puzzles, and exclude from consideration resources (e.g., bandwidth, I/O, memory, etc) needed for the MB to relay content to clients. The motivation for this is to allow focus on how the puzzle rate and difficulty constrain the MB; i.e., this is the limiting factor. It follows that if the MB has sufficient resources to solve the puzzles sent to it, then we assume it will have sufficient additional resources to relay content to an arbitrary number of clients. We assume the MB does not store a local copy of the traffic it receives from the provider; it initiates a connection to the provider with each client connection request.

We use the $M/M/k$ queueing model [16] to represent the queueing system at the MB, where we assume the puzzle

[2]We assume a legitimate client uses all of its available computational resources to solve each puzzle it receives promptly.

arrival is modelled by a Poisson process, and the puzzle-solving time is exponentially distributed. This model considers k serving units, which in our case is the number of puzzles the MB is able to solve in parallel. The waiting time of this model is [16]:

$$W = \frac{1}{\mu} + \left(\frac{(k\rho)^k}{k!(1-\rho)} + \sum_{i=0}^{k-1} \frac{(k\rho)^i}{i!} \right)^{-1} \left(\frac{\rho (k\rho)^k}{\lambda (1-\rho)^2 k!} \right) \quad (5)$$

where

$$\rho = \frac{\lambda}{k\mu} \quad (6)$$

In the queueing terminology, λ is the customer arrival rate to the system and μ is the customer departure rate from each of the k serving units ($\mu = 1/(\text{service time/customer})$), both measured in customers/time unit. Customers arriving and departing the system resemble, in our case, unsolved puzzles arriving and solved puzzles departing the MB. Customer-service time at each serving unit resembles puzzle-solving time at each of the MB's cores.

To realize the maximum n that satisfies Inequality 4, we first need to represent W as a function of n. We use the waiting time of (5), and express λ and μ in terms of n, r, t and g. Because the provider sends puzzles at a rate of r puzzles/second to each client connection, the puzzle arrival rate at the MB is $\lambda = nr$ puzzles/second. The rate of solving puzzles at each of the k cores is g times faster than that of a client; hence, $\mu = g/t$. Substituting in (6), we get:

$$\rho = \frac{nrt}{kg} = \frac{nb}{kg} \quad (7)$$

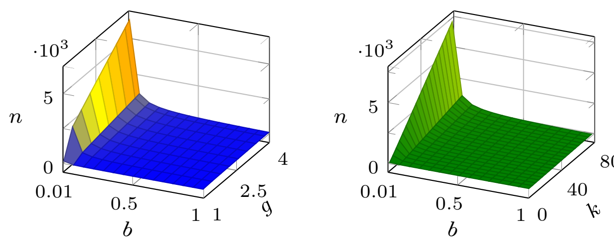
Note that the MB can prevent its queue from growing indefinitely by maintaining $\lambda < k\mu$ [16], which occurs if it keeps the number of simultaneous clients $n < kg/b$. However, only satisfying this inequality can still disclose the MB's relaying actions to the provider (as it does not ensure satisfying Inequality 4). By substituting ρ obtained as in (7) for that in (5), we express W in terms of n, t, r, k and g. Inequality 4 (which can be rewritten as $W/t - 1 \leq 0$) then becomes:

$$\frac{1}{g} + \left(\frac{(\frac{nb}{g})^k}{k!(1-\frac{nb}{kg})} + \sum_{i=0}^{k-1} \frac{(\frac{nb}{g})^i}{i!} \right)^{-1} \left(\frac{\frac{nb}{kg}(\frac{nb}{g})^k}{nb(1-\frac{nb}{kg})^2 k!} \right) - 1 \leq 0 \quad (8)$$

Using linear iterative root finding [17], we can find the maximum integer value of n that satisfies Inequality 8.

To study the behaviour of n with respect to b, k and g, we consider a range of values for each of those parameters in the intervals $[2^{-6}, 1]$, $[1, 80]$ and $[1, 4]$ respectively. Fig. 1(a) shows the change of n at $k = 25$, and Fig. 1(b) at $g = 1.5$. We ignore n when $b > 1$ because the provider should never set b in that range. Otherwise, unsolved puzzles start to accumulate at legitimate clients, increasing the RTT due to additional queueing delay, and falsely rejecting these clients.

From Inequality 8, we can see that n and b always occur multiplied together, hence by replacing all occurrences of nb with γ, we can express n in terms of γ and b as $n = \gamma/b$. That is, *n follows a hyperbolic decay with b (for all $b > 0$) with a scale factor of γ*. The maximum value of γ that makes



(a) At $k = 25$. (b) At $g = 1.5$.

Fig. 1. Maximum theoretical number of clients simultaneously-colluding with the MB. The lines on the surfaces are equally spaced on the b, g and k axes. The charts show that n responds to b (influenced by the provider) quicker than that to k and g (influenced by the MB). These results illustrate the potential of puzzles in limiting the number of colluding clients.

W satisfy Inequality 4 grows with k and g. For example, in Fig. 1(b)—where $g = 1.5$—every integer value, κ, on the k axis defines the scale factor, $\gamma = f(1.5, \kappa)$, of a hyperbolic decay of n with respect to b at κ.

The results plotted in Fig. 1 show that n follows an almost linear growth with g and k, versus a hyperbolic decay with b. The provider influences b through t and r, the MB controls k and influences g by investing in hardware. This puts the MB in a critical situation as the provider has a more significant impact on n than the MB has.

A. Simulation Results

The analytical evaluation showed how client puzzles affect the number of clients the MB could support in case the MB decides to solve the puzzles on behalf of the clients it colludes with. We now study the case where the MB decides to forward the puzzles to those colluding clients. We use the network simulator (ns-2) [11] to evaluate the rate of false rejects (FRs), where a legitimate client is rejected by the provider; and false accepts (FAs), where a client colluding with the MB is accepted. We assume the provider will endure some error while estimating RTT$_e$ in (1). This error scales RTT$_e$ by a factor β, such that:

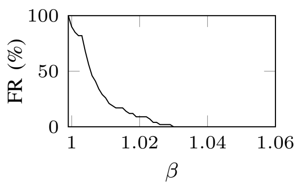
$$\text{RTT}_e = \beta \times \text{RTT}_a \quad (9)$$

See (2) for RTT$_a$. FRs tend to increase when $\beta < 1$, FAs tend to increase when $\beta > 1$.

Our simulation scenarios involved several runs with 100 nodes and random connectivity patterns. Nodes distribution and link latencies were designed to resemble networks distributed over a large geographic region. One node was set to be the provider, another was set to be the MB, while other nodes simulated clients. Some clients were connected directly to the provider (legitimate clients), others (colluding clients) were connected through the MB. FRs and FAs are shown in Fig. 2. For the runs we conducted, the error scale in the range $1.03 < \beta < 1.1$ yields 0% FRs and 2% FAs. We believe these results show promising potential for the solution we propose herein.

IV. Further Considerations

How many puzzles per second should the provider send to a client, and what should their difficulty be? Fig. 1 showed



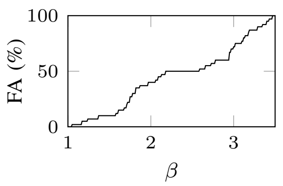


Fig. 2. False reject (FR) and false accept (FA) obtained from simulations; β represents the error of the provider's RTT estimation.

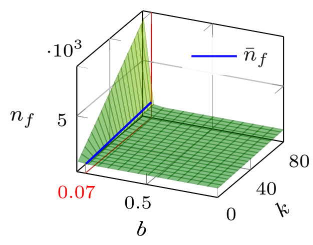


Fig. 3. Fitted surface at $g = 1.5$ represented by (10). The values of the constants in (10) are: $A = 5.64$, $B = -58.13$, $C = 3.9$ and $D = 4.37$. Normalized Root-Mean-Square-Deviation (NRMSD) over the displayed b and k intervals is 0.04 (or 4%).

a tradeoff between allowing more clients to collude with an MB, and overwhelming legitimate clients. To deal with this tradeoff, providers may set b to the value that satisfies a central tendency of n, such as the mean $\bar{n}$, over desired intervals of b, k and g.

One way to calculate $\bar{n}$ is to, first, approximate a function that mimics the behaviour of n. This can be done using curve fitting [18]. For example, at $g = 1.5$ and $2^{-6} \leq b \leq 1$, n_f can mimic the behaviour of n, such that:

$$n_f = k(Ae^{bB+C} + D) \tag{10}$$

where A, B, C and D are constants—their values are shown on Fig. 3. The mean, $\bar{n}_f$, in terms of k is:

$$\bar{n}_f = \frac{1}{1-2^{-6}} \int_{2^{-6}}^{1} k(Ae^{bB+C} + D)\,\mathrm{d}b = 8.9k \tag{11}$$

Substituting $\bar{n}_f$ for n_f in 10, and solving for b, we get:

$$b = \frac{1}{B}\left(\ln \frac{8.9k - Dk}{kA} - C\right) = 0.07 \tag{12}$$

That is, considering the abstraction given in §III and our queueing model, when b is restricted to the range $2^{-6} \leq b \leq 1$ and $g = 1.5$, the mean of n_f occurs at $b = 0.07$. Beyond this value of b, puzzles will overwhelm legitimate clients without significant drop in the number of colluding clients n whereas below, n rapidly increases with little reduction in the puzzle workload on legitimate clients. This highlights selection of an example value of b which may be of practical interest.

To set b, the provider adjusts r and t such that their product b results in the desired value. Because the network RTT is typically measured in ms [19], a puzzle that takes a relatively long time (e.g., 1 sec) to solve on an average client machine may overshadow the network RTT. Providers need to consider that when setting the puzzle difficulty, as it affects t.

Finally, providers may consider varying the puzzles' difficulty randomly, and discarding the observed RTT of puzzles that are harder than certain undisclosed threshold to avoid having their solving time overshadow the network RTT. This may penalize an MB significantly as it will not be able to distinguish *time-sensitive* puzzles (those where the provider will account for their RTT) from others, and will have to solve them in order of arrival. Having a number of relatively difficult puzzles in the MB's queue will raise the waiting time of all others behind them, making it easier for the provider to capture the highly-delayed responses of timed puzzles, thus, detecting the MB.

V. Conclusion

We proposed to use client puzzles and delay estimation to enable providers hinder unauthorized MB relaying of traffic. Our evaluation shows that the maximum number of clients an MB can collude with follows a hyperbolic decay with the rate of creation of puzzles and the time required to solve them; both factors are influenced by the content provider, versus an almost-linear growth with the MB's computational resources.

Acknowledgments. This work was supported by Canada's NSERC through the Canada Research Chairs program (third author) and Discovery Grants (second and third authors).

References

[1] Peterson et al., "A position paper on data sovereignty: The importance of geolocating data in the cloud," in *USENIX NSDI*, 2011.
[2] C. Dietrich and C. Rossow, "Empirical research of IP blacklists," in *ISSE 2008 Securing Electronic Business Processes*, 2009.
[3] E. Bertino, B. Catania, M. L. Damiani, and P. Perlasca, "GEO-RBAC: a spatially aware RBAC," in *ACM SACMAT*, 2005.
[4] *BBC News - US employee'outsourced job to China'*, http://www.bbc.co.uk/news/technology-21043693.
[5] Nikiforakis et al., "Cookieless monster: Exploring the Ecosystem of Web-based Device Fingerprinting," in *IEEE S&P*, 2013.
[6] Wei et al., "Classification of access network types: Ethernet, wireless LAN, ADSL, cable modem or dialup?" *Computer Networks*, vol. 52, no. 17, 2008.
[7] C. Jin, H. Wang, and K. G. Shin, "Hop-count Filtering: An Effective Defense Against Spoofed DDoS Traffic," in *ACM CCS*, 2003.
[8] G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet, "Revealing middlebox interference with tracebox," in *ACM IMC*, 2013.
[9] E. Rescorla, *SSL and TLS: Designing and Building Secure Systems*. Addison-Wesley Reading, 2001.
[10] A. Juels and J. G. Brainard, "Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks." in *NDSS*, 1999.
[11] Breslau et al., "Advances in network simulation," *Computer*, vol. 33, no. 5, 2000.
[12] Landa et al., "Measuring the Relationships between Internet Geography and RTT," in *IEEE ICCCN*, 2013.
[13] Wong et al., "Octant: a comprehensive framework for the geolocalization of Internet hosts," in *USENIX NSDI*, 2007.
[14] Tritilanunt et al., "Toward Non-Parallelizable Client Puzzles," in *Cryptology and Network Security*. Springer, 2007, vol. LNCS 4856.
[15] S. Doshi, F. Monrose, and A. D. Rubin, "Efficient memory bound puzzles using pattern databases," in *Springer ACNS*, 2006.
[16] D. Gross, J. Shortle, J. Thompson, and C. Harris, *Fundamentals of Queueing Theory*. Wiley, Hoboken, NJ, 2008.
[17] J. F. Traub, *Iterative Methods for the Solution of Equations*. AMS Bookstore, 1982.
[18] J. Philips, *Online Curve and Surface Fitting at ZunZun.com*, http://zunzun.com/, 2011.
[19] M. Crovella and B. Krishnamurthy, *Internet Measurement: Infrastructure, Traffic and Applications*. John Wiley & sons, 2006.