# Design and Implementation of Cross-Domain Cooperative Firewall

Jerry Cheng[1], Hao Yang[2], Starsky H.Y. Wong[1], Petros Zerfos[3], Songwu Lu[1]
UCLA Computer Science Department, Los Angeles, CA 90095[1]
IBM T.J. Watson Research Center, Hawthorne, NY 10532[2]
Deutsche Telekom Laboratories, Ernst-Reuter-Platz 7 D-10587, Berlin, Germany[3]
Email: {chengje,hywong1,slu}@cs.ucla.edu[1], haoyang@us.ibm.com[2], petros.zerfos@telekom.de[3]

***Abstract*— Security and privacy are two major concerns in supporting roaming users across administrative domains. In current practices, a roaming user often uses encrypted tunnels, e.g., Virtual Private Networks (VPNs), to protect the secrecy and privacy of her communications. However, due to its encrypted nature, the traffic flowing through these tunnels cannot be examined and regulated by the foreign network's firewall, which may lead the foreign network widely open to various attacks from the Internet. This threat can be alleviated if the users reveal their traffic to the foreign network or the foreign network reveals its firewall rules to the tunnel endpoints. However, neither approach is desirable in practice due to privacy concerns.**

**In this paper, we propose a *Cross-Domain Cooperative Firewall* (CDCF) that allows two collaborative networks to enforce each other's firewall rules in an oblivious manner. In CDCF, when a roaming user establishes an encrypted tunnel between his home network and the foreign network, the tunnel endpoint (e.g., a VPN server) can regulate the traffic and enforce the foreign network's firewall rules, without knowing these rules. The key ingredients in CDCF are the distribution of firewall primitives across network domains, and the enabling technique of efficient *oblivious membership verification*. We have implemented CDCF and integrated it with the OpenVPN software, and evaluated its performance using extensive experiments. Our results show that CDCF can protect the foreign network from encrypted tunnel traffic with minimal overhead.**

## I. Introduction

Security and privacy are two major concerns in supporting roaming users across administrative domains. Nowadays many organizations have deployed Virtual Private Networks (VPNs) [4] to protect their users when they roam into foreign networks. Once a roaming user establishes a VPN tunnel with her home network, she can access not only the private resources within the home network, but also redirect her Internet traffic through the VPN tunnel, which is typically encrypted to protect the secrecy of the user traffic.

While roaming users enjoy the security protection offered by VPNs, little consideration has been given to the impact of such encrypted tunnels on the foreign network. In particular, the foreign network's firewall cannot effectively regulate such tunneled traffic, because it is unable to examine the encrypted connection properties, such as destination IP addresses and ports. As a result, certain connections that are normally prohibited by the foreign network, for either security or policy reasons, can now circumvent the firewall regulation. The existence of such unregulated tunnels not only weakens the security protection for the roaming users, but more importantly leaves the foreign network widely open to various security threats from the public Internet.

At first glance, this problem may be alleviated by having the roaming user expose her decrypted traffic to the foreign network. Alternatively, the foreign network could also publish its firewall rules for the roaming user to self-regulate her traffic at the tunnel endpoint. However, neither approach is desirable in practice due to privacy concerns. On one hand, it is unlikely that users are willing to reveal their traffic (or their decryption keys) to the foreign network, which is exactly the motivation for deploying VPNs in the first place. On the other hand, network administrators are also reluctant to publish the firewall rules in use, which can expose sensitive information about the internal network topology and the administrative policies. With these conflicting security and privacy requirements, it is very difficult to regulate the encrypted tunnels using conventional firewall techniques, because they all require a single entity to possess knowledge on both the connection characteristics and the firewall rules.

In this paper, we present the design and implementation of CDCF, a *Cross-Domain Cooperative Firewall* that allows two networks to collaboratively enforce each other's firewall rules in an oblivious manner. As a result, CDCF can properly regulate the encrypted tunnel traffic of a roaming user, yet preserve the privacy of all parties involved (i.e., the roaming user, the home network and the foreign network). The key ingredients in CDCF are the distribution of firewall primitives across network domains, and the enabling technique of efficient *oblivious membership verification*.

As an example, consider the scenario in which a user roams into a foreign network $F$ and establishes an encrypted tunnel with her home network $H$. In CDCF, each network chooses a secret key individually and never reveals it. During the bootstrapping phase, the foreign network $F$ encrypts its firewall rules using its own key $K_F$. It sends the encrypted rules to the home network $H$, which applies another encryption function on them with its key $K_H$. These double-encrypted rules are then sent back to $F$. Whenever the roaming user attempts to establish a new connection within the tunnel, the VPN server at $H$ encrypts the connection descriptor using

1-4244-1588-8/07/$25.00 ©2007 IEEE

$K_H$ and sends it to the firewall at $F$, which then encrypts it again using $K_F$. Thanks to the use of a commutative cipher [8], the firewall at $F$ can use the oblivious membership verification algorithm (Section III) to match the double-encrypted connection descriptor against the double-encrypted rules. The resulting verdict is sent to the VPN server, which then filters the connection traffic accordingly. Note that in this process, the rule matching is done at the foreign network's firewall, while the verdict is enforced by the home network's VPN server. Such cross-domain interaction occurs only at the time of connection setup. The VPN server can cache the received verdicts and then perform the per-packet filtering task for the subsequent data packets using only local information.

We have implemented a prototype system and integrated it with OpenVPN [20], an open-source VPN software. We have also evaluated its performance using extensive experiments. Our results show that CDCF can enforce the firewall policy on the encrypted tunnels at an affordable cost. With CDCF, a roaming user experiences no extra delay in packet processing, except for 0.4 seconds of delay for the *first* packet in a new connection. Our oblivious comparison algorithm can be executed in real-time even by software modules on commodity PCs. The performance of CDCF also scales well as more roaming users participate in the system or the firewall ruleset grows larger. These results indicate that CDCF can be readily deployed in a practical mobile networking environment.

In summary, our contributions in this work are four-fold:

- The identification of the needs for and the challenges in protecting networks from encrypted tunnels;
- An architecture of cross-domain, cooperative firewall for regulating the tunneled traffic in a distributed manner;
- A novel *oblivious membership verification* algorithm that enables rule matching in an oblivious manner;
- A prototype implementation with desirable practical performance.

The rest of this paper is organized as follows. Section II discusses our system settings as well as the privacy issues in cross-domain firewalls. Section III presents the design of our proposed CDCF architecture in detail. Section IV describes our prototype implementation and presents performance evaluation results using both experiments and analysis. Section V discusses several issues surrounding the design of CDCF, and Section VI compares with the related work. Finally, Section VII concludes the paper.

## II. Issues for Cooperative Firewall

In this section, we describe the system settings and elaborate on the privacy issues that we seek to address.

### A. System Settings

We consider a home network that provides permanent network access to its associated users and also a foreign network that offers temporary connectivity to its visiting mobile users. Both networks deploy firewalls to monitor and regulate the traffic entering or leaving their respective network boundary. We consider a static packet-filtering firewall that filters traffic based on IP header of the data packets. The rules used by these firewalls are specified by the 4-tuple of *[src_IP, src_port, dest_IP, dest_port]*, where each field can be either a distinct value or a range of values. Each rule is associated with a verdict, either *accept* or *reject*, that instructs the firewall to pass through or block the corresponding traffic. In addition, a firewall can employ the *default-deny* policy to block all traffic which is not explicitly allowed, so that it can maximize the degree of protection against unknown or newly emerged threats.

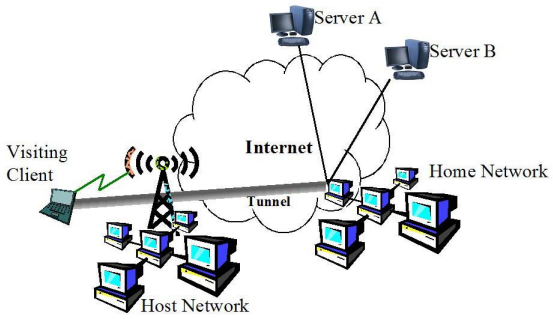


Fig. 1. The system settings for cooperative firewall

As illustrated in Figure 1, a roaming user can establish an encrypted tunnel between her home network and the foreign network where she is currently visiting. In practice, this is often done by using the widely deployed VPN technology. In order to protect her privacy, the user can redirect her traffic to/from the Internet, as well as accessing the home network's internal resources, through the encrypted tunnel. However, such traffic encryption and redirection bring non-trivial security challenges to the foreign network. In particular, the redirected traffic, with its connection descriptor encrypted, cannot be examined by the foreign network's firewall. As a result, it may be abused to circumvent the firewall regulation and pass through certain traffic that is normally prohibited by the foreign network for security or policy reasons.

The existence of unregulated traffic can create substantial security problems for network administrators. For instance, a VPN-enabled back-door into the private network can facilitate attackers in stealing information or injecting harmful viruses and trojans. Moreover, many universities have opted to ban P2P software in an effort to curb the illegal downloading of pirated software or MP3 content through campus network access. Public libraries and schools often block content that is deemed inappropriate for children. While firewall is not a panacea to all these problems, it is still one of the most effective lines of defense in practice. Therefore, joint regulation on the visitors' redirected traffic becomes a desirable security feature for network firewalls.

### B. Privacy Issues

To enable cooperative filtering across administrative domains, one fundamental challenge is to preserve the privacy of different parties. In general, three pieces of information are needed to perform joint traffic filtering: the home network's firewall rules, the foreign network's firewall rules, and the user connection descriptors. Ideally, if all parties openly share these information, then a firewall placed in either the home or the foreign network suffices to regulate the redirected traffic. However, in practice, both the user and the networks are reluctant to reveal their respective information to outside parties. On one hand, users tend not to trust foreign networks and do not want information on their traffic to be exposed. On the other hand, network administrators wish to restrict access to their firewall policies as they are usually considered a network asset, which can be potentially abused to exploit security vulnerabilities in the network configuration.

Therefore, we must carefully consider how much information is needed in order to ensure joint firewall enforcement. In particular, we must identify how to limit information exposure. For example, how can we keep the firewall rules confidential? Does every packet need to be revealed? In addition, we also need to consider the direction in which information is shared. For example, should a user reveal its traffic information (in a disguised form) to the foreign network, or should the foreign network disclose its firewall rules (again, in a disguised form)?

## III. Design

In this section, we present our design of the Cross-Domain Cooperative Firewall (CDCF). We start with an overview of CDCF in Section III-A, then describe the CDCF operations and the underlying oblivious comparison algorithms in Section III-B. Finally, we present several techniques that can further enhance the privacy protection of CDCF in Section III-C.

### A. Design Overview

Our proposed CDCF architecture protects the privacy of the user traffic as well as the network firewall policies through two mechanisms. First, we limit information exposure by distributing the firewall's basic operations, namely *rules-matching* and *verdict enforcement*, to the foreign network and the home network respectively. Secondly, we obfuscate the minimally exchanged information and perform oblivious rules-matching using the cryptographic technique of commutative encryption. The resulting CDCF architecture achieves cooperative filtering in three phases of operations: bootstrapping, per-connection evaluation, and per-packet enforcement.

**Bootstrapping** - This phase involves the preparation of the firewall rules into encrypted format, which can be used for oblivious comparison. The encrypted rule set is stored at the foreign network (for rules-matching of firewall operations, see Section III-B.1)

**Per-connection evaluation phase** - In this phase, each newly initiated connection is first compared against the home network's firewall rules when it exits the tunnel. The traffic allowed by the home network will be compared against the foreign network firewall rules (using oblivious comparison of Section III-B.2). The comparison result and the firewall verdict are stored at the home network (for verdict enforcement of the firewall operation, see Section III-B.1).

**Per-packet enforcement phase** - In this phase, each subsequent packet in the connection is either passed through or dropped by the home network, according to the verdict that was stored from the previous phase.

### B. Limiting Information Sharing

The enforcement of the home network firewall rules on user traffic is straightforward since the home network knows the connection details of user traffic. For this reason, we primarily focus on how the foreign network firewall rules can be enforced, while limiting the amount of information that needs to be shared. To achieve this, we decompose the firewall operations into smaller, autonomous steps and distribute them to the home and foreign networks, depending on which network the information that is needed by each step is available. Commutative encryption is also employed to ensure oblivious comparisons against singular values and also ranges of values.

*1) Decoupling Firewall Operations:* The firewall operation is decomposed into two steps: *rules-matching* and *verdict enforcement*. Usually, these steps are coupled, which means that knowledge of both rules and connection information is needed to complete them. Moreover, such coupling limits flexibility in the placement of the firewall functionality in either network. Once these steps are decoupled, only the firewall rule and initial packets are needed for rules-matching, and subsequent packets will simply follow the initial match. Similarly, verdict enforcement will only require the corresponding verdict and the subsequent data packets.

Placement of the two firewall operations across the home and foreign networks also needs to be considered among the possible combinations. We choose to assign the rules-matching operation to the foreign network and the verdict enforcement to the home network, to avoid having the home network possess the foreign network's firewall ruleset. Such a decision requires the initial packet of each new connection to be compared at the foreign network, potentially resulting in the foreign network learning of user traffic information. We address this issue in the next section using oblivious comparison.

*2) Oblivious Comparison:* Oblivious comparison is a critical component of CDCF used for blind comparison of distinct values and ranges. The notion of "oblivious comparison" implies that the actual numbers, or the ranges of numbers, need not be known to the entity carrying out the comparison. This function is achieved through a pair of cryptographic techniques: *commutative cipher* and *oblivious membership verification*.

**Oblivious Comparison of Singular Values** To perform oblivious comparison of two singular values, we make use of commutative encryption ciphers [2], [3], [5], [11], [15], [19]. A cipher $CE$ is commutative if and only if it possesses the

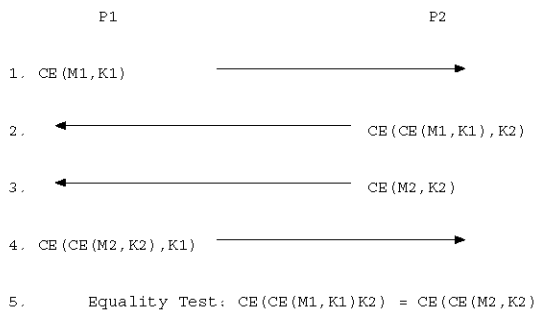


Fig. 2. Two parties P1 (holding key $K_1$ and message $M_1$) and P2 (holding key $K_2$ and message $M_2$) perform oblivious singular value comparison. The keys and messages are never revealed to the other party.

following property: For any message $M$ and two given keys $K_1$ and $K_2$, we have

$$CE(CE(M, K_1), K_2) = CE(CE(M, K_2), K_1) \quad (1)$$

That is, when one uses the commutative cipher to apply two encryption operations on a message using two different keys, the order of these encryptions does not change the resulting cipher-text. Additionally, the order of the decryption does not affect the resulting plain text. We defer the cryptographic implementation of such commutative ciphers to Section IV.

The commutative property of Equation 1 allows for two parties to obliviously compare two values by following the simple protocol illustrated in Figure 2. If the equality comparison fails, no information about the values is learnt by the other party. In CDCF, by using oblivious comparison, the user connection information can be matched, field by field, to relatively simple rules where each field of the rule is a singular value.

**Oblivious Membership Verification** In practice, firewall rules are usually expressed over ranges of IP addresses and ports for efficiency purposes, and the firewall needs to decide whether the user connection, which is a singular value, falls in these ranges. This problem is known as *membership verification*.

The naive approach for oblivious membership verification would be to first enumerate all the values in the given range, then employ the previous technique on each of these values to check for equality. However, this method is prohibitively expensive for firewall rules, where each field has a large domain (e.g., $2^{32}$ for IP address fields). Thus, we propose a novel, *oblivious membership verification* algorithm with only $O(b)$ complexity for both storage and computation, as compared to $O(2^b)$ in the naive approach, where $b$ is the number of bits to encode the numbers (e.g., 32 for IPv4 addresses).

Given a $Range$ and a singular $Value$, the basic idea of our oblivious membership verification algorithm is to search for the existence of a common range $R$ that satisfies both $R \subseteq Range$ and $Value \in R$. Based on results from Set Theory, the existence of such an $R$ implies a positive membership verification result, i.e., $Value$ falls within $Range$.

The search for $R$ is equivalent to generating two sets of ranges, $\alpha$ and $\beta$, and performing set intersection $\alpha \bigcap \beta$; where $\alpha$ is the set of all possible ranges containing the $value$ and $\beta$ is the set of all possible subranges that satisfy the subset relationship with the $Range$. The search space is large, but can be reduced by limiting the ranges in $\alpha$ and $\beta$ to those that follow the binary prefix format[1]. For $\alpha$, it is shown in [14] that the minimum number of binary prefixes that an arbitrary range can be converted to is at most $2b - 2$, where $b$ is the number of the bits covering the domain, and the union of these prefixes is equal to the original range. An example of such a transformation is shown in Figure 4, where the given range is decomposed into four subranges represented in binary prefix format. For $\beta$, each discrete value in the $b$-bit domain, there exist exactly $b + 1$ binary prefixes containing the value as their member. Algorithm 1 illustrates the pseudo code for generating these $b + 1$ binary prefixes, while an example is shown in Figure 3.

**Algorithm 1** DiscreteValueDecomposition($val$)

**Require:** $val$ is a discrete value
1: var $mask, i, range[b+1]$
2: $mask \leftarrow 2^b - 1$
3: **for** $i = 0$ to $b$ **do**
4: $\quad mask \leftarrow$ shiftLeftOneBits($mask$)
5: $\quad range[i] \leftarrow val \ \& \ mask$
6: $\quad$ /* $range[i]$ is a range with $i$ wildcards */
7: **end for**

By requiring the binary prefix format, the number of subranges generated in $\alpha$ and $\beta$ is limited to the order of $O(b)$. From $\alpha$ and $\beta$ we can determine whether a $Value$ falls into a $Range$ by examining whether a common range $R$ exists between the two sets of ranges generated. In Figures 4 and 3, the existence of a common subrange [72, 79] in both decompositions indicates the success of membership verification as the value 74 does fall within the given range [69, 81].

To preserve privacy, we need to hide the identity of each range of $\alpha$ and $\beta$ while they are being compared. The commutative cipher technique can again be applied here as the only comparison needed is the equality testing. Although each range in $\alpha$ and $\beta$ is a single binary prefix, which is suitable for equality comparison, further transformation is necessary for the binary prefix to be encrypted. More specifically, the wildcard (*) needs to be removed from the binary prefix representation, yet its function (specifying the size of the range) must be retained. We achieve this by prepending each binary prefix with a pre-defined binary preamble.

A binary preamble is a fixed bit pattern that helps transform each unique binary prefix into a unique encryptable format (i.e. without the wild cards). By prepending the binary prefix[2], the

[1]Binary prefix format consists of a binary bit pattern concatenated with zero or more wildcards representing the range.

[2]It is important that the leading zero(s) of the binary prefix are also preserved for a unique transformation.

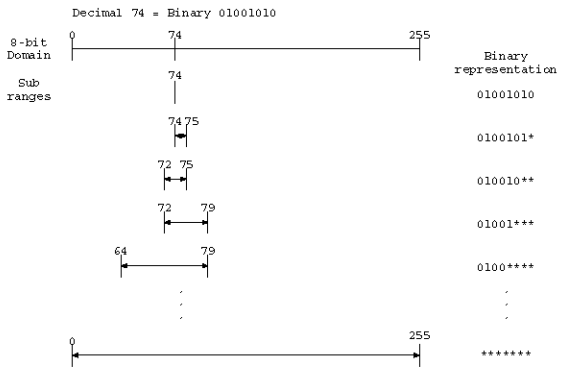


Fig. 3. An example of generating $b + 1$ binary prefixes that encompasses the decimal value of 74, where $b$ is the size of the domain (8)

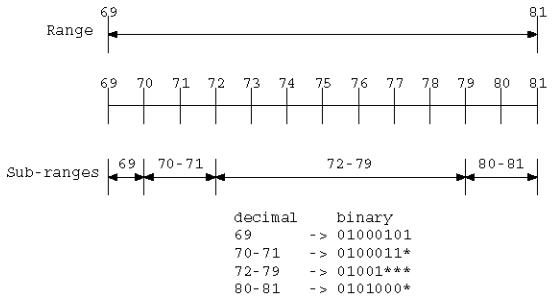


Fig. 4. An example of decomposition of a range into subranges. Each subrange follows the binary prefix format and as a set they exactly partition the original range

wildcards can be removed, resulting in a unique bit string that distinctively identifies a specific binary prefix. For example, consider the subrange [72, 79] in Figure 4 is represented as a binary prefix format of 01001*** in an 8-bit domain. Prepending a preamble of 1011 and removing the wild card (*) results in 101101001. Note that this string uniquely represents *one and only one* binary subrange (i.e. 72 to 79). No other binary prefix, using the same binary preamble, will share this unique string. Having generated each of the binary prefixes (as shown in Figure 3) and prepended them with the same preamble, the two sets of binary prefixes can be compared to identify whether a given value falls within a particular range.

Finally, by (1) applying the decomposition algorithm and permuting the generated subranges and (2) following the encryption sequence similar to Figure 2 on the entire batch of subranges, two parties can achieve oblivious membership verification between a $Value$ and a $Range$. For our proposed membership verification algorithm, the number of encryptions needed is linear ($O(b)$) instead of exponential ($O(2^b)$) as in the naive approach. The straightforward search for a common subrange in the two sets of $\alpha$ and $\beta$ would require $O(b^2)$ equality tests, and we can further improve it to linear overhead with techniques such as hashing.

### C. Privacy Enhancement Techniques

In this section, we consider the potential causes of privacy leakage in the basic CDCF design, and develop several techniques to further enhance the privacy for both the mobile user and the foreign network.

*1) User Connection Privacy:* In CDCF, the foreign network is designated to perform rules-matching, which may lead to privacy leakage for the mobile users. In particular, if the foreign network can identify the rules in their double encrypted form, then it can infer, at least partially, the user connection by checking which rule the connection is matched to. There are two possible methods for the foreign network to identify the commutatively encrypted rules. First, it can study the representation of the rule. For example, it may keep track of the rules that have produced 6 subranges in destination port field. Second, it can determine the identity of the rules through their logical order[3]. To address these issues and enhance the user traffic privacy, we propose the use of the technique of dummy fields/rules to make the rules indistinguishable, and the dummy connections technique to obfuscate the user traffic.

**Dummy Fields/Rules** We define a dummy value as a number that is beyond the domain of the field (i.e. any number greater than $2^{16}$ is a dummy value for the port number field). Dummy values never appear in a valid rule. However, when commutatively encrypted, they are indistinguishable from the legitimate values. There are two ways to use the dummy values for privacy enhancement purposes. First, for each field in the firewall rules, we can pad the decomposed subranges with dummy values. The idea here is that some rules may contain fields that do not decompose into multiple subranges. These fields can be identifiable even after the double encryption operations, as there may be only one rule that satisfies such property. Padding the resulting subranges with dummy fields would prevent such form of identification. These *dummy fields* can thus be used to effectively eliminate the structural artifact of range decomposition. Second, we can organize a set of dummy values into *dummy rules* and randomly insert them between the original rules, which can hide the ordering of the rules.

The dummy values can optionally be negotiated during the bootstrapping phase as follows: the home network requests the foreign network to supply a set of single encrypted dummy values; these values are then double encrypted and inserted into the ruleset by the home network before returning them to the foreign network.

**Dummy Connections** Alternatively, we can also obfuscate user traffic through the use of dummy connections. These connections are normal queries with valid values in each field[4], but randomly generated by the home network instead of being extracted from the user traffic. This way, the foreign network cannot distinguish a real user connection from the artificially

[3] It may utilize first-match policy, for which the order of the rules has to be preserved.

[4] This is different from dummy values and dummy rules, where the values fall outside of the domain of the field.

generated dummy ones.

*2) Foreign Network Privacy:* CDCF achieves strong privacy for the foreign network because its firewall ruleset is only released in an encrypted form. However, a dishonest home network may intentionally probe the firewall in an attempt to learn its policies. In particular, the foreign network can perform *range probing* as follows: instead of submitting valid queries with decomposed discrete values, it can probe the firewall with specially constructed queries that contain random sets of disjointed ranges. A match in this case may potentially reveal more information about the firewall rules than a properly formulated query. However, we show in Appendix that such range probing is no more efficient than brute force probing, i.e., the home network can never exploit CDCF to better probe the firewall.

## IV. Implementation and Evaluation

In this section, we briefly describe our prototype implementation and evaluate its performance through experiments.

### A. Implementation

To demonstrate the practicality of CDCF, we implemented a prototype system and integrated it with OpenVPN [20], an open-source VPN software. In what follows, we elaborate on several key issues in the implementation and integration.

*1) System Components:* As shown in Figure 5, our system consists of several modules on both home network and foreign network sides[5]. The *Range Decomposition* module implements our proposed algorithm for decomposing ranges and values into subranges with the binary prefix format. The *Encryption* module implements a specific commutative cipher based on the Pohlig-Hellman algorithm [21] as follows:

$$CE(M, K) = M^K \mod P, \tag{2}$$

where $M$ is the message, $K$ is the key and $P$ is a large prime modulus. Clearly, this cipher satisfies the commutative property of Equation 1. We use the GNU Multiple Precision Bignum library [12] for handling large integers.

The *Communication* module uses TCP as the transport protocol. There are five types of messages exchanged between the foreign network and the home network. As shown in Figure 6, the first three types are used for firewall rule exchange, while the last two are used for verdict queries and responses. The *Rule Manager* maintains a table of firewall rules, in the double-encrypted form, which are used by foreign networks. It also performs oblivious comparison between these encrypted rules and the encrypted connection tuple. The *Verdict Enforcement* module keeps track of all the verdicts received from foreign networks and performs packet filtering based on the associated verdict. For efficient lookup, these verdicts are stored in a multi-layer AVL tree structure, with each layer corresponding to one field in the connection information.

[5]Note that one network may act as the home network for its own users and, in the same time, the foreign network for its guest users. As such, it needs to implement functions on both sides.

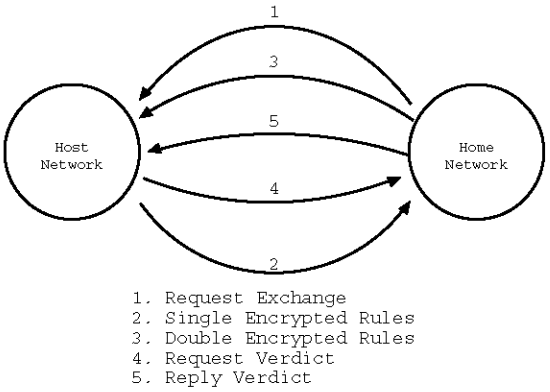


Fig. 6. Messages used between home and foreign network CDCF

*2) Integration with OpenVPN:* We integrated our prototype system with OpenVPN [20], which relies on the TUN/TAP virtual network driver in Linux for IP tunneling. Our integration requires no modification to the VPN client, and minimal modifications to the VPN server. Specifically, we insert a hook into the forwarding path inside the VPN server, so that CDCF can intercept packets from the tunnel and enforce the verdicts. Depending on its associated verdict, a packet is either returned to the VPN forwarding path (with "allow" verdicts) or dropped (with "reject" verdicts) by CDCF.

### B. Experiment Results

We evaluate the performance of the CDCF by measuring the latency through experimentation and analyzing the storage requirement. The testbed for the experiment consists of two machines in US to serve as the foreign network. One machine (Pentium 4, 1.5GHz, 128MB) plays the role of foreign network CDCF, while the other machine (Pentium 4, 800MHz, 128MB) acts as the visiting user's laptop. The home network CDCF (Pentium 4, 1.5GHz, 256MB) is located in Hong Kong.

*1) CDCF Overhead:* We are interested in the delays associated with the three phases of the CDCF operations as shown in Figure 7. We have selected the Pohlig-Hellman algorithm [21] for commutative encryption. The size of the prime modulus is a 1024 bits safe prime. Modular exponentiation is considered an expensive operation when the size of the exponent (encryption key) is large. To improve the performance of the CDCF system and reduce the delay experienced by the users, we limit the size of the exponent (encryption key) to be 160 bits following the recommendation in [25]. Each experiment varies from 10 to 100 firewall rules following the average number of firewall rules shown in [24], [26]. The firewall rules are randomly generated and padded with dummy fields to achieve indistinguishability. Traffic information matched against the rules, in the worst case scenario such that every field and every rule are compared against the traffic information before returning a verdict.

**Overhead of Bootstrapping Phase** The bootstrapping phase incurs a one-time cost and only needs to be re-initiated if the firewall policies are modified. We measure the computation

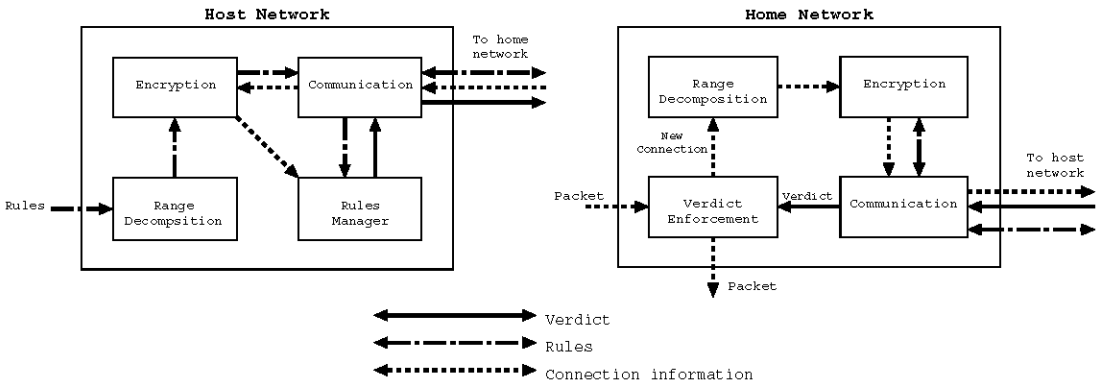


Fig. 5. Components in CDCF and their interactions

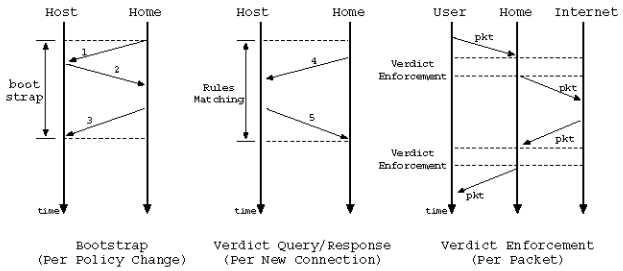


Fig. 7. Time delays associated with the three phases of the CDCF operation. The number attached to the arrows represents message type

delay involved, from the moment the foreign network receives the request until it has the commutatively encrypted firewall rules stored in the rules manager and is ready to answer verdict queries. The result of the overhead is shown in Figure 8 as a function of the number of rules in the firewall rule set. Each data point consists of the average of one hundred runs, using randomly generated rules, encryption keys and prime modulus. We observe an approximate linear increase in overhead with the increase in number of firewall rules.

**Overhead of the Verdict Query/Reply Phase** The Verdict Query/Response phase takes place for every *new* connection that is forwarded through the tunnel by the mobile user. We measure the delay that is experienced by the users and the latency incurred from rules-matching operation at the foreign network.

Figure 9 shows the computational delay as experienced by the user (top line), which included the processing latency for the foreign network (lower line). This graph indicates that the mobile user would experience approximately 0.4 seconds of computational delay per *new* connection. With a small optimization during the bootstrap phase, decrypting the double encrypted firewall rules using the foreign network's key before storing, each query would only take less than 0.01 second to process. Figure 10 shows a comparison of delay (with/without CDCF), as experienced by mobile users, for a new connection initiated at the foreign network (USA), tunneled to the home network (Hong Kong), and further forwarded to a Yahoo web server after the Verdict Query/Response phase. The user would experience approximately 0.6 second of additional delay with the application of CDCF, with approximately 250 ms contributed by the roundtrip time for verdict query between Hong Kong and the USA.

Figures 9 and 10 show an almost negligible increase in latency with increasing number of firewall rules. Majority of the overhead, as we have identified, are due to encryption (approximately 0.39 second per query). This is a limitation of utilizing software encryption. However, in an enterprise server, encryptions can be handled by dedicated hardware. [16] has product available for purchase that can perform 4,000 1024-bit RSA transactions (modular exponentiation operation) per seconds, which would approximately reduce our encryption overhead for each query from 0.39 second to 0.03 second.

**Overhead of the Verdict Enforcement Phase** The Verdict Enforcement phase consists in performing lookups for the verdicts stored in the multi-layer verdict tree, and consequently applying this verdict. We randomly inserted verdicts into the multi-layer AVL tree and then generated packets with connection information that matched these verdicts. Figure 11 shows the average delay for the home network to perform verdict enforcement. The delay grows logarithmically with the number of verdicts as the AVL tree provides efficient verdict lookup through binary search. Subsequent packets of an existing connection would experience only negligible delay on the orders of sub-microseconds.

*2) Storage Requirements:* In our design, the storage requirement for firewall rules is augmented due to the application of range decomposition and encryption.

The range decomposition generates 2b sub-ranges per rule in the worst case, where b is the number of bits used for the representation To evaluate the average case, we sampled 1,000,000 randomly generated ranges within the increasing size of the domain (number of bits b). Figure 12 shows the average number of subranges generated through range decomposition approaches b.

A large prime modulus results in a larger cipher text, since the encrypted field has the same number of bits as the prime modulus. For the CDCF, each encrypted field requires 1024 bits of storage, the same size as our chosen prime modulus p.

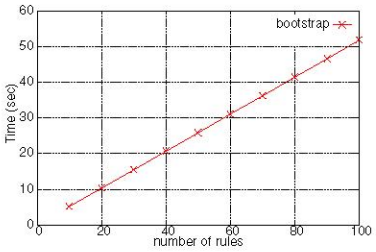


Fig. 8. Overhead of Bootstrapping Phase (RTT not included)

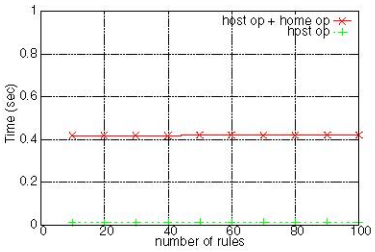


Fig. 9. Overhead of performing Verdict Query and Reply as measured from the home network (RTT not included)

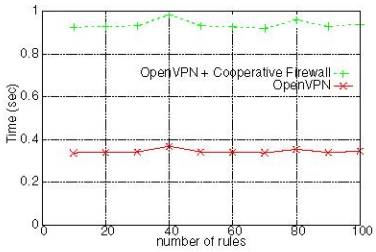


Fig. 10. Comparison of overhead per new connection, with and without the CDCF (RTT included)

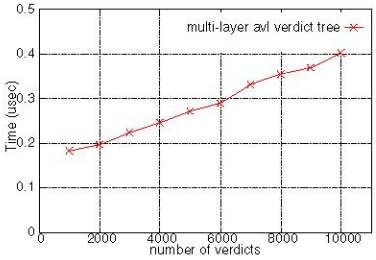


Fig. 11. The overhead of using multi-layer AVL verdict tree to perform verdict enforcement

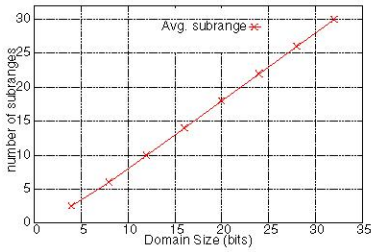


Fig. 12. The average number of subranges generated for arbitrary range within a domain

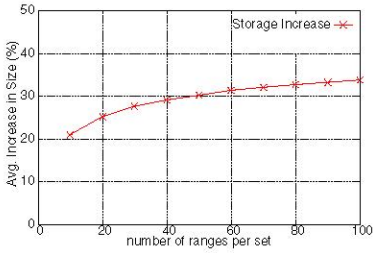


Fig. 13. The average size increase for the use of dummy value to achieve rules indistinguishability

To achieve rules indistinguishability, dummy values can optionally be used to pad the number of subranges appearing in each field. It is sufficient to increase the number of subranges appearing in a field to match the rules having the most subranges in the respective field. For the average increase in size due to dummy value, we randomly generated sets of ranges, varying from 10 to 100 ranges per set, for a 32-bit value domain. Figure 13 shows the average size increase with increasing number of ranges per set.

*3) Privacy Evaluation:* Now we evaluate the effectiveness of our cooperative firewall in preserving the privacy from two perspectives: the privacy of firewall rules, and the privacy of user traffic.

In our design the firewall rules are never disclosed in plain text. The home network never reveals its firewall rules. On the foreign network side, the only time that its firewall rule set leaves its custody is in the bootstrapping phase. However, these rules are revealed only in an encrypted form, and the encryption key is never revealed. Without the foreign network's encryption key, the home network can never decrypt the original rule set from its single- or double- encrypted form. One may think that even though the home network does not know exactly foreign network's firewall rule set, it may cache the encrypted rules and use them to unilaterally predict the foreign network's decisions on other traffic. However, this is not possible in our design, because rules-matching requires the double encrypted traffic information, which is known only to the foreign network. From each answer, the home network knows only the verdict binding to the query, and no additional information can be inferred.

The privacy of the user traffic, on the other hand, depends on both the form of firewall rules used by the foreign network and the user usage pattern. For example, if the firewall has only one rule which specifies only one specific connection, then a match against the firewall rules clearly indicate the user traffic information (as the one specified in the rule). Our design addresses this issue from two aspects as follows. First, we leverage the expressiveness of the typical firewall rule set. In practice, a firewall often has multiple rules, and to improve the filtering performance, each rule specifies a wide range of potential connections. In such cases, the foreign network can infer little information from whether the user traffic matches the entire rule set or not. Second, the home network can optionally send multiple dummy queries, and the foreign network cannot differentiate them from the real user traffic.

We also provide a simple quantitative approach to evaluating the achievable privacy given a set of firewall rules and specific user traffic. We define a privacy metric as the uncertainty in inferring the user traffic information. Specifically, given a user connection, if the foreign network cannot differentiate it from a set of $\gamma$ candidate connections, the uncertainty is $\gamma$. Because the user traffic is denoted by a tuple of five fields, the uncertainty on each field contributes to the overall privacy level.

Consider a user connection that matches to a rule. Each field of the rule contains an interval of integers, with a length $l_i$. In this case, the uncertainty of the user traffic is:

$$\gamma = l_1 \times l_2 \times l_3 \times l_4 \times l_5 \quad (3)$$

By utilizing the privacy enhancing dummy queries, the

query may match with up to $k$ rules with $k$-1 dummy connection, its uncertainty becomes the sum of the uncertainty due to each individual rule:

$$\gamma = \sum_{i=1}^{k} l_{i1} \times l_{i2} \times l_{i3} \times l_{i4} \times l_{i5} \quad (4)$$

where $l_{i1}$ is the interval length in the first field of the first matched rule, and so on.

## V. Discussion

In this section, we discuss several issues in our design.

**Firewall Rules** Our current design focuses on firewall rules specified by 4-tuples in a packet's IP header. In principle, our oblivious comparison technique is generic for handling any types of numeric values. As such, it can be readily extended for more sophisticated firewall rules, such as scanning the packet payload for certain byte signatures. However, the practical downside is the increased processing overhead, as we need to perform encryption on every byte boundary; Otherwise, we may miss a byte pattern due to mis-alignment. If the payload has 1000 bytes, then roughly 1000 encryptions are needed from each participating party, which will significantly slow down the processing capability of the firewall.

**Transit Network Firewalls** So far our design considers only the firewall interaction between the home network and the foreign network. However, it can be extended to incorporate transit networks and enforce their policies as well. Specifically, the home network can treat each transit network as an additional foreign network that regulates the user traffic. While one additional bootstrapping phase is needed for each transit network, the per-connection evaluation overhead can be minimized by having the home network send out queries to all transit networks in parallel. Finally, the home network can still cache all received verdicts and enforce them on each data packet locally.

**User Traffic Privacy** When the number of concurrent connections is small, our dummy connection technique may not be very effective in preserving the user traffic privacy. The reason is because in such cases, even a random guess can have a non-trivial success probability. Nevertheless, we believe that in most enterprise-level networks where firewall operations are critical, the number of concurrent connections is large enough to defeat such blind guesses.

**Firewall Ruleset Privacy** Our design does not seek perfect ruleset privacy for the foreign network's firewall in the presence of probing. Note that probing is possible with or without CDCF. In general, after a mobile user is admitted into the foreign network, she becomes an "insider" and thus can very well probe and learn the firewall ruleset through brute-force trial and error. Rather than preserving perfect privacy, we seek to preserve the degree of difficulty to probe the rule set, such that the home network cannot do better than brute-force probing as what a mobile "insider" can always launch.

## VI. Related Work

Firewalls have been widely deployed as the frontier security defense against malicious attacks and unwanted traffic. In recent years, the cooperation between firewalls across multiple domains has attracted much attention. For example, Shaer and Hamed have studied the identification and modeling of rule conflicts between different firewalls [22], [23]. However, such conflict analysis requires the firewalls to disclose their rules to each other, which may not be practical when they are within different administrative domains. In contrast, our design allows multiple firewalls to perform cooperative filtering yet preserves their ruleset privacy. Such cross-domain collaboration is also fundamentally different from the distributed firewall [17] or the multilayer firewall [17], which extends the firewall capability within a single network. Recently Lee et al. [18] propose to protect the secrecy of firewall rules using encryption. However, it focuses on safekeeping the firewall rules locally and does not consider any collaboration between the firewalls. In this paper, we have considered a static firewall for its simplicity, which differs from the stateful firewall studied in [13]. Yet many suggestions in [13] can potentially be adopted to extend our design at a cost of some additional overhead.

While VPNs have been increasingly popular nowadays, their interaction with firewalls becomes an important security issue for mobile users. RFC 2356 [10] discusses the necessary firewall support for enabling encrypted tunnels, and commercial products integrating VPN and firewall [7] have already been released. However, these efforts focus on the issues associated with the home network's firewall. To our best knowledge, the impact of VPN tunnels on the foreign network's firewall has never been studied before.

Oblivious comparison has been well studied in the cryptography community. It was first introduced by Yao [1] in the *Two-Millionaire Problem*, where two millionaires try to compare who is richer, without revealing their actual wealth to each other. The solution provided by Yao has the complexity of $O(2^b)$, where $b$ is the number of bits for representing the domain, and is subsequently improved to $O(b)$ using secure circuit evaluation [9]. These techniques have been successfully applied in many areas, such as private information retrieval and privacy-preserving auctioning [2], [3], [5], [6], [15]. They can also be applied to evaluate whether a value falls in a given range, by comparing the value with the range's upper and lower bounds. However, when the value falls outside the range, this approach has the drawback that it reveals whether the value falls above or below the range. Such information can be used to design effective binary probing (i.e. by guessing higher or lower in the next query). In contrast, our proposed oblivious membership verification algorithm only performs equality tests. Therefore, no additional information is leaked even if the value does not fall into the range in the query.

## VII. Conclusion

In this paper, we have presented the design and implementation of CDCF, a *Cross-Domain Cooperative Firewall* architecture that can enable the collaborative security in terms of joint

traffic filtering without exposing much of the shared information. The novelty of CDCF is the distribution of the firewall primitives of rule matching and verdict enforcement, as well as the enabling technique of efficient oblivious comparison through commutative cipher and a novel range comparison technique. Our prototype implementation and experimental results have shown that CDCF can readily be deployed to greatly enhance the mobile network security at marginal cost.

## VIII. Acknowledgments

We deeply appreciate the insightful and constructive comments from the anonymous reviewers. We would also like to thank Dr. John C. S. Lui for providing the equipment to run the experiment in Hong Kong.

## References

[1] A. Yao, "Protocols for secure computations," in *Proc. FOCS*, 1982.
[2] D. Agrawal and C. Aggarwal, "On the design and quantification of privacy preserving data mining algorithms," in *Proc. PODS*, 2001.
[3] R. Agrawal, A. Evfimievski, and R. Srikant, "Information sharing across private databases," in *Proc. SIGMOD*, 2003.
[4] B. Gleeson, et. al, "A framework for IP based virtual private networks," February 2000, RFC 2764.
[5] M. Bawa, R. Bayardo, and R. Agrawal, "Privacy-preserving indexing of documents on the network," in *Proc. VLDB*, 2003.
[6] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," in *Proc. ACM CCS*, 1999.
[7] Cisco VPN Concentrator, http://www.cisco.com/en/US/products/hw/ vpndevc/ps2284/.
[8] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. on Information Theory*, vol. 22, no. 6, pp. 644–654, November 1976.
[9] M. Fischlin, "A cost-effective pay-per-multiplication comparison method for millionaires," in *Proc. Conference on Topics in Cryptology (CT-RSA)*, 2001.
[10] G. Montenegro and V. Gupta, "Sun's SKIP Firewall Traversal for Mobile IP," RFC 2356, 1998.
[11] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin, "Protecting data privacy in private information retrieval schemes," in *STOC '98*, 1998, pp. 151–160. [Online]. Available: citeseer.ist.psu.edu/579393.html
[12] GNU Multiple Precision Arithmetic Library, http://www.swox.com/gmp/.
[13] M. G. Gouda and A. X. Liu, "A model of stateful firewalls and its properties," in *Proceedings of the IEEE International Conference on Dependable Systems and Networks (DSN)*, June 2005, pp. 320–327. [Online]. Available: http://www.cse.msu.edu/~alexliu/publications/Stateful/stateful.pdf
[14] P. Gupta and N. McKeown, "Algorithms for packet classification," *IEEE Network*, vol. 15, no. 2, pp. 24–32, 2001.
[15] B. Hore, S. Mehrotra, and G. Tsudik, "A privacy-preserving index for range queries," in *Proc. VLDB*, 2004.
[16] Interface Masters: Niagara 2100B, http://www.interfacemasters.com/products/nic/niagara2100b/.
[17] S. Ioannidis, A. Keromytis, S. Bellovin, and J. Smith, "Implementing a distributed firewall," in *Proc. ACM CCS*, 2000.
[18] J. Lee, J. Jeon, and K. Yoo, "A security scheme for protecting security policies in firewall," *SIGOPS Operating System Review*, vol. 38, no. 2, pp. 69–72, 2004.
[19] G. Liang, "Privacy-preserving inter-database operations." [Online]. Available: citeseer.ist.psu.edu/660634.html
[20] OpenVPN, http://openvpn.net/.
[21] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance," *IEEE Transactions on Information Theory*, vol. IT-24, pp. 106–110, 1978.
[22] E. Shaer and H. Hamed, "Firewall policy advisor for anomaly detection and rule editing," in *Proc. Integrated Management (IM)*, 2003.
[23] E. Shaer and H. Hamed, "Discovery of policy anomalies in distributed firewalls," in *Proc. IEEE INFOCOM*, 2004.
[24] E. Shaer and H. Hamed, "Modeling and management of firewall policies," *IEEE Trans. Network and Service Management*, vol. 1, no. 1, April 2004.
[25] P. van Oorschot and M. Wiener, "On Diffie-Hellman key agreement with short exponents," in *Proc. Eurocrypt*, 1996.
[26] A. Wool, "A quantitative study of firewall configuration errors," *Computer*, vol. 37, no. 6, pp. 62–67, 2004.

## Appendix

**Proposition:** *Range probing is no more efficient than brute force probing.*

*Proof:* We denote the home network's range as $R_1$ and the foreign network range as $R_2$. Only equality test can be performed between $R_1$ and $R_2$; no subset relationship can be determined. Therefore, the failure of exact matching provides *no guidance* for further querying the subranges of $R_1$. For a domain of $b$-bits, there are $2^b$ distinct values. The number of distinct binary prefixes is expressed by the following equation:

$$\sum_{i=0}^{b} 2^i = 2^{b+1} - 1 > 2^b \tag{5}$$

Furthermore, even if $R_1$ is equal to $R_2$, this fact does not guarantee that all values within this range share the same verdict of the rule that specifies $R_2$. In particular, it is common in firewall to have two rules with different verdicts to overlap in their specified range. □

### Handling Range Probing

We further propose a method for the foreign network to identify range probing attempts through a challenge and response mechanism.

Any two of the ranges generated by Discrete Value Decomposition should share a subset relationship. However, the set of ranges in a range probing query does not satisfy this property and can be detected as follows:

1) The foreign network can challenge of a query by randomly choosing a pair of encrypted ranges for a particular field and ask the home network to reveal them.
2) The two ranges in clear should exhibit the subset property where one range contains the other range. If not, then the foreign network concludes that the home network is performing range query.
3) Otherwise, the foreign network will encrypt these two unencrypted ranges then ask the home network to double encrypt it and return the result to the foreign network.
4) Upon reception, the foreign network will perform a final check against the previous version of the double encrypted range. If the two versions of encrypted ranges do not match, then the foreign network can conclude that the home network is performing range query.
5) Non-repudiation can be achieved through the logging the messages exchanged to be used as a proof.

This technique does not require additional messages being sent because the exchange can be piggy-backed in the query/response messages. The validity of the initial query can be verified upon the reception of the third query.