Technical Report MS-CIS-12-10

Private and Verifiable Interdomain Routing Decisions

Proofs of Correctness

Mingchen Zhao
University of Pennsylvania

Wenchao Zhou
University of Pennsylvania

Alexander J. T. Gurney
University of Pennsylvania

Andreas Haeberlen
University of Pennsylvania

Micah Sherr
Georgetown University

Boon Thau Loo
University of Pennsylvania

Introduction

This technical report contains proofs of correctness for the VPref protocol described in "Private and Verifiable Interdomain Routing Decisions" [2].

Below, we use cursive notation to indicate faulty ASes — that is, a producer P that is faulty is written as $\mathcal{P}$; a consumer C that is faulty is written as $\mathcal{C}$; and an elector E that is faulty is written as $\mathcal{E}$.

1 Detection

The detection proofs make use of the following definitions and lemmas:

Definition 1. *We say that h is* well-formed *iff*

$$h = H(H(q_1, x_1)|| \ldots ||H(q_k||x_k))$$

for a sequence of k bits $q_1, \ldots, q_k$ and bit strings $x_1, \ldots, x_k$.

Note that a correct elector will always construct a well-formed h in step four of the Commitment Phase.

Definition 2. *We say that a message is* observable *iff it directly or indirectly affects at least one correct node.*

Our definition of *observable* is similar to the notion defined by Haeberlen et al. in PeerReview [1], and is necessary since we cannot expect faulty nodes to reveal telltale messages exchanged only between faulty nodes. Our goal is to detect malicious behavior that affects (either directly or indirectly) a correct node.

Lemma 1. *If (1) a faulty elector $\mathcal{E}$ does not send an identical $\sigma_{\mathcal{E}}(h)$ message to each producer and consumer where h is well-formed; (2) both the Commitment and Verification Phases of the protocol are executed; and (3) all the neighbors of $\mathcal{E}$ are correct, then at least one correct neighbor will detect the fault.*

Proof. If, in step five of the Commitment Phase, $\mathcal{E}$ does not transmit a commitment to each correct AS, then the correct AS that expects $\sigma_{\mathcal{E}}(h)$ will detect that $\mathcal{E}$ is faulty and raise an alarm.

If $\mathcal{E}$ does not properly sign h, then the receiving producer or consumer will discover that $\mathcal{E}$ is faulty.

If $\mathcal{E}$ does not transmit the same value $\sigma_{\mathcal{E}}(h)$ to all ASes, then at least one correct neighbor (who receives $\sigma_{\mathcal{E}}(h)$) will discover that $\mathcal{E}$ is faulty during the Verification Phase when a neighbor broadcasts a VERIFY$(\sigma_{\mathcal{E}}(h'))$ message where $h' \neq h$.

Finally, since the hash function is assumed to be collision resistant (Assumption 1), if $\mathcal{E}$ transmits an h that is not well-formed during step five of the Commitment Phase, then the receiving producers and consumers will discover during the Verification Phase that $\mathcal{E}$ is faulty since $\mathcal{E}$ cannot provide a bit proof for any bit. (A valid bit proof, by definition, contains sufficient information for the receiver to reconstruct the preimage of a well-formed commitment. If the commitment is not well-formed, then $\mathcal{E}$ cannot provide a valid bit proof for any bit, since this would imply that the receiver's digest, which is computed using the well-formed pre-image from the bit proof, is identical to the faulty elector's non-well-formed commitment.) □

Lemma 2. *If h is well-formed and constructed using the bit sequence $b_1, \ldots, b_k$, then a valid bit proof for bit i must indicate that the ith bit of the bitstring used to construct h is b_i.*

Proof. The proof is by contradiction. Assume that a valid bit proof indicates that the ith bit of the bitstring used to construct h is $\neg b_i$ (the negation of b_i). A bit proof for the ith bit consists of q_i (the claimed ith bit; by assumption, $q_i = \neg b_i$), x_i, and for all $j \neq i$, $H(q_j||x_j)$.

Let h' be the hash computed by the receiver of the bit proof, using the provided information from the bit proof. (Recall that the receiver computes its own hash h' and compares it with the elector's commitment.) Since $q_i \neq b_i$, the preimages of h and h' must differ. Since, by Assumption 1, the hash function H is collision-resistant, then h and h' must also differ. Hence, since $h \neq h'$, the bit proof is not valid, which contradicts our assumption, completing the proof. □

Lemma 3. *If (1) the elector is faulty during the Commitment Phase or breaks a promise, (2) both the Commitment and Verification Phases of the protocol are executed, and (3) all the neighbors of the faulty elector are correct, then at least one correct neighbor will detect the fault.*

Proof. Let $\mathcal{E}$ be the faulty elector. The neighbors of $\mathcal{E}$ are the producers $P_1, \ldots, P_r$ and the consumers $C_1, \ldots, C_q$. The proof proceeds by case analysis of $\mathcal{E}$'s possible misbehaviors:

- In any step of the protocol, if $\mathcal{E}$ improperly raises an alarm (signaling that its neighbor did not transmit a required message), then $\mathcal{E}$ is trivially discovered by its correctly operating neighbor.
- If $\mathcal{E}$ sends a message other than in steps two, five, and six of the Commitment Phase, then the receiving AS trivially discovers that $\mathcal{E}$ is faulty.
- In step two of the Commitment Phase, if the elector responds with an acknowledgment that is not properly signed, then the receiving producer trivially detects the fault. If $\mathcal{E}$ sends a properly signed acknowledgment $\sigma_E(q)$ where $q \neq \sigma_{P_i}(r_i)$, then producer P_i trivially detects that $\mathcal{E}$ is faulty.
- If $\mathcal{E}$ sends either a commitment that is not well-formed or sends non-identical commitments to its neighbors, then by Lemma 1, a correct neighbor discovers that $\mathcal{E}$ is faulty.
- In step six of the protocol, if $\mathcal{E}$ does not properly sign its message, its misbehavior is immediately detected by the correct consumer. If the message sent by $\mathcal{E}$ to C_j is neither of the form $\sigma_{\mathcal{E}}(C_j, \perp)$ nor $\sigma_{\mathcal{E}}(C_j, \sigma_{P_i}(r_i), \sigma_{\mathcal{E}}(r_i))$, then C_j trivially discovers that $\mathcal{E}$ is faulty.
- Without loss of generality, let C_w be a consumer that receives a non-null route c_w ($c_w \neq \perp$) from $\mathcal{E}$ in step six of the Commitment Phase. (Note that either such a consumer exists, or the theorem vacuously holds since no promises are broken.) That is, C_w receives $\sigma_E(C_w, \sigma_{P_i}(c_w), \sigma_E(c_w))$. If the signature $\sigma_{P_i}(c_w)$ is invalid, then C_w discovers that $\mathcal{E}$ is faulty (since $\mathcal{E}$ should have previously validated P_i's signature).

  A promise is broken if (i) c_w is the route chosen by the elector and given to C_w, (ii) there exists a route r_x that is in a different indifference class than c_w, and (iii) $c_w \leq_w r_x$ where $\leq_w$ is the promise given to C_w. Let R_h be the indifference class that contains c_w and R_x be the indifference class that contains r_x (and hence, $R_h \leq_w R_x$). During the Verification Phase, C_w will request a bit proof for all b_z such that $R_h \leq_w R_z$ and $R_h \neq R_z$. From the definition of the protocol, the bit proof must reveal that each b_z is `0`; if it does not, C_w discovers that $\mathcal{E}$ is faulty. To avoid being discovered by the consumer, $\mathcal{E}$ must provide a bit proof that b_x is `0`, and therefore must have set $b_x =$ `0` when computing its value of h in step four of the Commitment Phase (this follows as a direct consequence of Lemma 2). However, the producer P_x that sent route r_x will request a bit proof for b_x and expect the value to be `1`.

  Since $\mathcal{E}$ must transmit the same value of h to all parties or be detected (Lemma 1), then $\mathcal{E}$ cannot provide correct bit proofs for both $b_x =$ `0` and $b_x =$ `1` (as a direct consequence of Lemma 2). Hence, if $\mathcal{E}$ breaks a promise, then either a correct producer or consumer discovers that $\mathcal{E}$ is faulty.

□

Lemma 4. *If some subset of producers and consumers is faulty during the Commitment Phase (i.e., the faulty nodes do not execute the VPref protocol correctly) and the elector is correct, then the elector will detect the fault.*

Proof. In any step of the protocol, if a faulty producer or consumer improperly raises an alarm (that is, incorrectly signals that the elector did not transmit a required message), then the faulty AS is trivially discovered by the elector.

Otherwise, there are two cases of faults to consider:

- **A producer $\mathcal{P}$ is faulty.** The only neighbor of $\mathcal{P}$ is the elector E.

  In step one of the protocol, $\mathcal{P}$ may send the message with an invalid signature. (If $\mathcal{P}$ does not send a message, then the elector raises an alarm and the lemma holds.) By Assumption 5, E's verification of the signature will fail and E will detect $\mathcal{P}$'s misbehavior.

- **A consumer $\mathcal{C}$ is faulty.** The only neighbor of $\mathcal{C}$ is the elector E.

  A correct consumer does not send messages during the Commitment Phase. Any message sent by $\mathcal{C}$ during the Commitment Phase will cause its misbehavior to be immediately discovered by the message's receiver.

Note that the case in which both producers and consumers are faulty is subsumed by the above cases, as each producer's faults (case 1) and each consumer's faults (case 2) will be independently detected by the elector.

□

Theorem 1. *If (1) an AS is faulty during the Commitment Phase or breaks a promise, (2) both the Commitment and Verification Phases of the protocol are executed, and (3) all the neighbors of the faulty AS are correct, then at least one correct neighbor will detect the fault.*

Proof. The theorem follows as a direct consequence of Lemmas 3 and 4. □

The following theorem considers the case of multiple, colluding (i.e., faulty) nodes:

Theorem 2. *If (1) both the Commitment and Verification Phases of the protocol are executed and (2) some faulty behavior is observable during the Commitment Phase, then either: (a) at least one correct neighbor will detect the fault, or (b) no promises to correct ASes will be broken with respect to the correct inputs from correct producers and some valid set of inputs that could have been sent by the faulty producers.*

Proof. Note that clause (b) acknowledges that the elector and a subset of faulty producers can always agree on some fake input routes; VPref does not assess the veracity of advertised routes.

The proof proceeds by case analysis:

- **Only the elector is faulty.** If only the elector is faulty, then the theorem holds by Lemma 3 and Theorem 1.

- **Only some subset of producers is faulty.** If only some subset of the producers is faulty, then the theorem holds by Lemma 4.

- **Only some subset of consumers is faulty.** If only some subset of the consumers is faulty, then the theorem holds by Lemma 4.

- **The elector and some subset of the consumers are faulty and collude.** From the definition of the protocol, consumers do not interact with other consumers or any producers during the Commitment Phase. Interactions between the consumer and the elector are not observable by a correct AS (condition 2). If a faulty consumer otherwise interacts with any correct AS, then the misbehavior is trivially detected and the theorem holds.

  The remaining cases in which a faulty elector causes an observable fault to the consumers are covered by the case analysis in Lemma 3.

- **The elector and some subset of the producers are faulty and collude.** From the definition of the protocol, producers do not interact with other consumers or any producers during the Commitment Phase. Interactions between the producer and the elector are not observable by a correct AS (condition 2). If a faulty producer otherwise interacts with any correct AS, then the misbehavior is trivially detected and the theorem holds.

  An elector may choose (i.e., send to a correct consumer) a route that had not been previously offered in step 1 of the protocol. There are three subcases to consider:

  - If the choice is not properly signed by a producer (step 6), then the correct consumer will detect the fault.
  - If the choice is properly signed, but the choice's indifference class was not indicated in the commitment sent in step 5 of the protocol, then by Assumption 1 (the hash function H is collision resistant) and Lemma 2, the elector cannot provide a valid bit proof that indicates that the choice *was* indicated in the commitment, and hence the correct consumer will detect the default.
  - If the choice is properly signed and its indifference class was indicated in the commitment sent in step 5 of the protocol, then this is equivalent to having the producer offer the path in step 1 of the protocol, and therefore no promise is broken and clause (b) of the theorem holds.

- **Some subset of the consumers and some subset of the producers are faulty and collude.** If some subset of the consumers and some subset of the producers are faulty and collude, then the theorem holds by Lemma 4.

- **Some subset of the consumers and some subset of the producers and the elector are faulty and collude.** This case holds as a direct consequence of the three previous cases.

□

2 Evidence

We make use of the following lemma:

Lemma 5. *If the elector is correct, it can produce a bit proof for any bit of the commitment.*

Proof. A bit proof for any bit i of the commitment consists of b_i, x_i, and for all $j \neq i$, $H(b_j||x_j)$. From the definition of the commitment (steps three and four of the Commitment Phase), the elector selects b_i and x_i for each bit i. Since the elector has access to the hash function H (Assumption 1), it can compute $H(b_j||x_j)$ for any bit j, completing the proof. □

The VPref protocol achieves the following *evidence* guarantees:

Theorem 3. *If a correct AS X detects that another AS Y was faulty during the Commitment Phase (i.e., it did not execute the Commitment Phase of the VPref protocol correctly), then either (i) Y did not send an expected message or did not properly sign a message with its key σ_Y in which case X raises alarm, or (ii) X can convince any correct AS that Y is faulty.*

Proof. From the definition of the protocol, if Y does not transmit an expected message to X or does not properly sign the message, then the theorem trivially holds since X will raise an alarm.

The proof proceeds by case analysis. We consider all cases in which an AS X detects that Y is faulty during the Commitment Phase.

- **A correct producer ($X = P_i$) detects a fault in the elector ($Y = \mathcal{E}$).** We consider each case in which P_i detects a fault in $\mathcal{E}$.
  - P_i detects a fault in the elector in step two of the Commitment Phase if P_i receives a message $\sigma_{\mathcal{E}}(q)$ where $q \neq \sigma_{P_i}(r_i)$ and r_i is the route advertised by P_i in step one of the protocol. Since $\sigma_{\mathcal{E}}(q)$ is properly signed by $\mathcal{E}$ and could not have been produced by another party (Assumption 3), $\sigma_{\mathcal{E}}(q)$ is a *proof of misbehavior* (PoM) against $\mathcal{E}$. That is, any correct AS that receives the PoM $\sigma_{\mathcal{E}}(q)$ can verify that q has not been properly signed by P_i and determine that $\mathcal{E}$ is faulty.
  - P_i detects that $\mathcal{E}$ was faulty during the Commitment Phase if P_i received any messages from $\mathcal{E}$ other than the acknowledgement in step two and the commitment in step five. If such a message exists and was not properly signed, then P_i raises an alarm. Otherwise, the message itself serves as a PoM against $\mathcal{E}$ since the message must have been produced by $\mathcal{E}$ (since no other AS knows $\sigma_{\mathcal{E}}$).
  - P_i detects that $\mathcal{E}$ is faulty if it previously received the commitment $\sigma_{\mathcal{E}}(h)$ from $\mathcal{E}$ in step five of the Commitment Phase and subsequently receives a $\text{VERIFY}(\sigma_{\mathcal{E}}(h'))$ message (where $h \neq h'$) during the Verification Phase. P_i can convince any correct AS that $\mathcal{E}$ is faulty by broadcasting the PoM $\text{INVALIDCOMMIT}(\sigma_{\mathcal{E}}(h), \sigma_{\mathcal{E}}(h'))$, since signatures cannot be forged and by the definition of the protocol, the elector should send the same commitment to each producer and consumer.
  - P_i detects that $\mathcal{E}$ is faulty if (i) P_i challenges $\mathcal{E}$ to provide a bit proof $b_x = 1$ for a previously sent route r_i (transmitted in step one of the Commitment Phase) where r_i is in indifference class R_x, and (ii) $\mathcal{E}$ either refuses or provides an incorrect proof. (By Lemma 5, a correct elector can

always provide a valid bit proof.) If $\mathcal{E}$ does not provide a proof, then P_i will raise an alarm, and the theorem holds.[1]

Otherwise, let $\sigma_{\mathcal{E}}(\text{PROOF}(q))$ be the signed (but incorrect) proof sent by $\mathcal{E}$ to P_i during the Verification Phase. P_i will broadcast

$$\text{PROOFCHALLENGE}(\sigma_{\mathcal{E}}(\sigma_{P_i}(r_i)), \sigma_{\mathcal{E}}(\text{PROOF}(q)))$$

to all other ASes as a PoM against $\mathcal{E}$. A correct AS that receives the PoM can verify that b_x should be `1` and that the elector's bit proof is invalid. The correct AS will therefore discover that $\mathcal{E}$ is faulty.

- **A correct elector ($X = E$) detects a fault in a producer ($Y = \mathcal{P}$).**
  - E detects a fault iff either $\mathcal{P}$ provides an invalid signature in step one or transmits a forged message in any other step of the Commitment Phase (recall that a correct producer participates only in step one of the Commitment Phase). In such cases, from the definition of the protocol, E raises an alarm and the theorem trivially holds.
- **A correct consumer ($X = C_i$) detects a fault in the elector ($Y = \mathcal{E}$).** We consider each case in which C detects a fault in $\mathcal{E}$.
  - C_i detects that $\mathcal{E}$ is faulty if it either does not receive a message from $\mathcal{E}$ in step six of the Commitment Phase or receives a message that has not been properly signed by $\mathcal{E}$. Here, C_i raises an alarm and the theorem trivially holds.
  - C_i detects that $\mathcal{E}$ is faulty if it receives a message of the form $\sigma_{\mathcal{E}}(q_1, q_2, q_3)$ in step six of the Commitment Phase, where either (i) $q_1 \neq C_i$, (ii) $q_2 \neq \sigma_{P_w}(r_j)$ for some producer P_w and route r_j, or (iii) $q_3 \neq \sigma_{\mathcal{E}}(r_j)$. The message $\sigma_{\mathcal{E}}(q_1, q_2, q_3)$ serves as a PoM against $\mathcal{E}$ since it could only have been produced by $\mathcal{E}$ and any correct AS can verify that it is not a correct announcement.
  - C_i detects that $\mathcal{E}$ is faulty if it previously received the commitment $\sigma_{\mathcal{E}}(h)$ from $\mathcal{E}$ in step five of the Commitment Phase and subsequently receives a $\text{VERIFY}(\sigma_{\mathcal{E}}(h'))$ message (where $h \neq h'$) during the Verification Phase. C_i can convince any correct AS that $\mathcal{E}$ is faulty by broadcasting the PoM $\text{INVALIDCOMMIT}(\sigma_{\mathcal{E}}(h), \sigma_{\mathcal{E}}(h'))$, since signatures cannot be forged and by the definition of the protocol, the elector should send the same commitment to each producer and consumer.
  - C_i detects that $\mathcal{E}$ was faulty during the Commitment Phase if C_i received any messages from $\mathcal{E}$ other than the commitment in step five and the elector's choice in step six. If such a message exists and was not properly signed, then C_i raises an alarm. Otherwise, the message itself serves as a PoM against $\mathcal{E}$ since the message must have been produced by $\mathcal{E}$ (since no other AS knows $\sigma_{\mathcal{E}}$).
  - C_i detects that $\mathcal{E}$ is faulty if (i) C_i challenges $\mathcal{E}$ to provide a bit proof $b_x = \texttt{0}$ for some b_x such that $R_x \leq_i c_i$ and $c_i \notin R_x$, and (ii) $\mathcal{E}$ either refuses or provides an incorrect proof. (By Lemma 5, a correct elector can always provide a valid bit proof.) If $\mathcal{E}$ does not provide a proof, then C_i will raise an alarm, and the theorem holds.[2]

[1] Additionally, P_i can broadcast the message $\text{PROOFCHALLENGE}(\sigma_{\mathcal{E}}(\sigma_{P_i}(r_i)), \emptyset)$, and a correct AS that receives the message will ask $\mathcal{E}$ for a bit proof; if $\mathcal{E}$ cannot provide such a proof, then the correct AS learns that $\mathcal{E}$ is faulty.

[2] Additionally, C_i can broadcast the message $\text{PROOFCHALLENGE}(\sigma_{\mathcal{E}}(C_i, \sigma_{P_j}(r_j), \sigma_{\mathcal{E}}(r_j)), \emptyset, \sigma_{\mathcal{E}}(\leq_i))$, where $\sigma_{\mathcal{E}}(C_i, \sigma_{P_j}(r_j), \sigma_{\mathcal{E}}(r_j))$ is the message it received in step six of the Commitment Phase. A correct AS that receives the message will ask $\mathcal{E}$ for a bit proof; if $\mathcal{E}$ cannot provide such a proof, then the correct AS learns that $\mathcal{E}$ is faulty.

Otherwise, C_i will broadcast

$$\text{PROOFCHALLENGE}(\sigma_{\mathcal{E}}(C_i, \sigma_{P_j}(r_j), \sigma_{\mathcal{E}}(r_j)), \sigma_{\mathcal{E}}(\text{PROOF}(q)), \sigma_{\mathcal{E}}(\leq_i))$$

to all other ASes, where $\sigma_{\mathcal{E}}(C_i, \sigma_{P_j}(r_j), \sigma_{\mathcal{E}}(r_j))$ is the message it received in step six of the Commitment Phase. A correct AS that receives the message can verify that b_x should be `0` and that the elector's bit proof is invalid, and that $\mathcal{E}$ is faulty.

- **A correct elector ($X = E$) detects a fault in a consumer ($Y = \mathcal{C}$).**
  - The elector E detects that $\mathcal{C}$ is faulty if E receives any message from $\mathcal{C}$ during the Commitment Phase, since a correct consumer does not send messages during the Commitment Phase. If the message is not properly signed, then E raises an alarm. Otherwise, the received message is itself a PoM against $\mathcal{C}$, and can be used to convince a correct AS that $\mathcal{C}$ is faulty.
- **A correct producer ($X = P$) detects a fault in a consumer ($Y = \mathcal{C}$).**
  - The producer P detects that $\mathcal{C}$ is faulty if P receives any message from $\mathcal{C}$ during the Commitment Phase, since a correct consumer does not send messages during the Commitment Phase to the producer. If the message is not properly signed, then P raises an alarm. Otherwise, the received message is itself a PoM against $\mathcal{C}$, and can be used to convince a correct AS that $\mathcal{C}$ is faulty.
- **A correct consumer ($X = C$) detects a fault in a producer ($Y = \mathcal{P}$).**
  - The consumer C detects that $\mathcal{P}$ is faulty if C receives any message from $\mathcal{P}$ during the Commitment Phase, since a correct producer does not send messages during the Commitment Phase to the consumer. If the message is not properly signed, then C raises an alarm. Otherwise, the received message is itself a PoM against $\mathcal{P}$, and can be used to convince a correct AS that $\mathcal{P}$ is faulty.

□

3 Accuracy

Theorem 4. *If an AS X is correct during the Commitment Phase, then no correct AS will detect a fault in X, and no valid evidence can exist against X.*

Proof. The proof proceeds by case analysis. We consider all cases in which either a correct AS Y detects a fault in another AS X or there exists some evidence against Y, and show that no case can hold when X is correct.

- **Detection/Evidence against Producer P_i.**
  - *Detection:* The elector E detects that P_i is faulty if either P_i does not send an expected message (step one) or provides an invalid signature. But since P_i is correct, it will transmit a properly signed message, and hence no correct AS will report a fault in P_i.

    A correct AS detects that P_i is faulty if P_i sends an unexpected message during the Commitment Phase (i.e., any message other than the route message sent in step one). However, since P_i is correct, it will not send any messages other than in step one of the protocol.
  - *Evidence:* An AS obtains a PoM against P_i iff P_i sends an unexpected and signed message during the Commitment Phase (i.e., any message other than the route message sent in step one). Since P_i is correct, it will not send any messages other than in step one of the protocol, and hence no valid evidence can exist against P_i.
- **Detection/Evidence against Consumer C_i.**
  - *Detection:* A correct AS detects that C_i is faulty iff C_i transmits any message during the Commitment Phase. Since C_i is correct, it will not transmit any messages during the Commitment Phase, and hence no correct AS will report a fault in C_i.
  - *Evidence:* An AS obtains a PoM against C_i iff C_i transmits any signed message during the Commitment Phase. Since C_i is correct, it will not transmit any messages during the Commitment Phase, and hence no valid evidence can exist against C_i.
- **Detection/Evidence against Elector E.**
  - *Detection:* There are several subcases to consider:
    - A correct AS detects that the elector is faulty if the elector improperly raises an alarm. However, since E is correct, it will not improperly raise an alarm.
    - If in any step of the protocol, E responds with a message that is not properly signed with σ_E, then the receiver will detect that E is faulty. However, since E is correct, it will properly sign each message with σ_E.
    - A correct producer P_i detects that E is faulty if either E does not send an acknowledge of P_i's route r_i or sends an acknowledgment that is not $\sigma_E(\sigma_{P_i}(r_i))$. However, since E is correct, it will properly acknowledge the route with $\sigma_E(\sigma_{P_i}(r_i))$.
    - A correct producer P_i detects that E is faulty if it received any message from E other than the acknowledgment in step two or the commitment in step five. Similarly, a consumer C_j detects that E is faulty if it received any message from E other than the commitment in step five and E's choice is step six. However, since E is correct, it will not send any such unexpected messages to either P_i or C_j.

* As a consequence of Lemma 1, a correct AS will detect that E is faulty if E transmits either a commitment that is not well-formed or non-identical commitments to its correct neighbors. However, since E is correct, it will send the identical well-formed commitment to all neighbors.
* A correct producer P_i that advertises a route r_i detects that the elector is faulty if the elector is unable to provide a bit proof that $b_x = \texttt{1}$, where R_x is the indifference class that contains r_i. However, by Lemma 5, a correct elector can produce a valid bit proof.
* A correct consumer C_i detects that E is faulty if E does not provide a bit proof that $b_z = \texttt{0}$ for all b_z such that $R_h \leq_w R_z$, where $R_h \neq R_z$ and R_h is the indifference class that contains c_w (the route that E sends to C_i). However, by Lemma 5, a correct elector can produce a valid bit proof.

– *Evidence:* There are several subcases to consider:

* A producer P_i obtains a PoM against E if P_i receives a message $\sigma_E(q)$ in step two of the protocol where $q \neq \sigma_{P_i}(r_i)$. However, since E is correct, it will send an acknowledgment where $q = \sigma_{P_i}(r_i)$.
* A producer P_i or a consumer C_j obtains a PoM against E if E sends a properly signed message $\sigma_E(q)$ to P_i other than in steps two and five of the Commitment Phase or if E sends a properly signed message $\sigma_E(q')$ to C_j other than in steps five and six of the Commitment Phase. However, since E is correct and obeys the protocol, it will only send messages to P_i in steps two and five, and to C_j in steps five and six.
* A producer P_i or a consumer C_j obtains evidence (in the form of an INVALIDCOMMIT message) if it previously received a commitment $\sigma_E(h)$ from E and subsequently received a VERIFY$(\sigma_E(h'))$ message (where $h \neq h'$) from another AS during the Verification Phase. However, since E is correct, it will send the same commitment ($\sigma_E(h)$) to each AS, and since signatures cannot be forged (Assumption 3), neither P_i nor C_j can obtain a message $\sigma_E(h')$ where $h' \neq h$.
* A producer P_i or a consumer C_j obtains a PoM against E if, when asked to provide a bit proof, E provides a signed but invalid proof (i.e., that $b_z \neq \texttt{1}$ in the case of P_i or, for C_j, that $b_w \neq \texttt{0}$ for some w where $R_w \leq_j R_h$, $R_w \neq R_h$, and R_h is the indifference class that contains the route received from the elector). However, by Lemma 5, a correct elector will always provide a valid bit proof.
* A consumer C_i obtains a PoM against E if it receives a message of the form $\sigma_{\mathcal{E}}(q_1, q_2, q_3)$ in step six of the Commitment Phase, where either (i) $q_1 \neq C_i$, (ii) $q_2 \neq \sigma_{P_w}(r_j)$ for some producer P_w and route r_j, or (iii) $q_3 \neq \sigma_{\mathcal{E}}(r_j)$. However, since E is correct, from the definition of the protocol, E will send either $\sigma_E(C_i, \perp)$ or $\sigma_E(C_i, \sigma_{P_w}(r_j), \sigma_E(r_j))$.

□

4 Privacy

Theorem 5. *If all ASes are correct, then no AS can learn anything from running VPref that it could not already have learned from running BGP.*

Proof. By Lemmas 6 and 7, no consumer can obtain additional information about the routes available to an elector E at any given time, nor about the policy of E (beyond what is specified in its promise).

By Lemma 8, no producer can obtain any information about which routes were available to E, nor about the policy of E.

Finally, E does not obtain any additional information about the BGP configuration or state of its neighbors: no VPref messages are sent to E, in the case when all ASes are correct, except for route advertisements (step one) and VERIFY requests, both of which contain only information that E already knows. □

In the following, let E be the elector, and C be a consumer. Assume that E is correctly implementing its promise to C, and that in so doing, it delivers a route r, for prefix p.

Lemma 6. *Let s be any route from E to p, such that s is not strictly preferred to r according to the promise. Then the consumer C is unable to deduce whether s was available to E.*

Proof. Note that if s were strictly preferred to r, then C could deduce from receiving r, and from the fact that E is correct, that s could not have been in E's input. (Theorem 1 ensures that if E had misbehaved, it would have been detected.)

Recall that C is unable to invert the hash function, and therefore can only observe the bit values b_i which have been explicitly revealed to it by E.

Let R_r and R_s be the classes of r and s respectively. If $R_r < R_s$, or if R_r and R_s are incomparable, then C will receive no bit proof for b_s, and will not learn the value of this bit. Consequently, it cannot tell whether s was present for E. Similarly, if $R_r = R_s$, then C will receive a proof for $b_r = 1$; but this happens whether or not s was present as well as r. Therefore C cannot deduce the presence or absence of s in this case either. □

An immediate corollary of this lemma is that C cannot deduce any information about the availability of routes from E, excepting those that are strictly preferred to r. The following lemma shows that the private policy of E is also hidden: if it does not promise any preference between two routes, then C cannot tell which one it would actually adopt, given the choice.

Lemma 7. *If r and s are two routes in the same indifference class, or in incomparable indifference classes, then C cannot tell what E's preference is between them.*

Proof. By Lemma 6, when C sees r, it cannot tell whether s was not present, or if it was present but not adopted. It is only in the latter case that C could deduce that r was preferred over s; as far as C knows, s could be preferred over r, but was simply not available at the time.

Symmetrically, if C sees s at some other time, it still cannot deduce whether r or s is most preferred: as far as C knows, it could be that one route vanished and the other appeared, without any comparison taking place between them. □

For producers, we can prove stronger versions of these results. In ordinary BGP, a producer P would be able to learn nothing at all about E's state or policy, other than that E has any route which P has given it. We show that from running VPref, P still does not gain any additional knowledge.

Lemma 8. *Let r be any route that E could have for a prefix p. No producer P can deduce whether E has a BGP RIB entry for r. Nor can P deduce any route preference implemented by E.*

Proof. In the commitment phase, P receives acknowledgements for any routes it has sent to E; no such acknowledgement is proof that the route has passed through E's ingress filter, or been installed into a BGP Adj-RIB-In or Loc-RIB, but only that the message was received. It also receives a signed hash value, but cannot use this to determine the values of any bits b_i. In no case does P receive information about routes that it did not itself send.

In the verification phase, if P had sent a route r_i in class R_i, it will receive a bit proof that $b_i = 1$. However, it cannot deduce that this bit was set because of the presence of its route r_i: it could be that E has actually adopted a more preferred route, in a class $R_j \leq R_i$. Indeed, P cannot deduce that E has any routes at all, since it cannot rule out the possibility that the more preferred route is actually the null route $\perp$.

Consequently, P also cannot deduce E's route preferences, since the information it sees does not depend on the outcome of E's routing decisions. □

Note that a producer P can discover facts about E's routing policy from observing the network in other ways; in particular, it could monitor incoming traffic to determine whether an offered route had been adopted. It could also offer two different routes at different connection locations, to see which one was chosen and ended up carrying traffic. However, this can all be done without using any of the output of VPref.

5 Inability to Support Inconsistent Promises

Theorem 6. *If an elector makes inconsistent promises, i.e., there exist indifference classes R_i and R_j and consumers C_a and C_b such that $R_i \leq_a R_j$ and $R_j \leq_b R_i$, then there exists a set of inputs such that the elector either must choose $e = \perp$ or break at least one promise.*

Proof. If the elector chooses $e = \perp$, then the theorem trivially holds.

Otherwise, a promise is broken to a consumer C_w if there exists a route r_x such that $c_w \leq_w r_x$ where c_j is the route sent by the elector to C_j, c_j and r_x are in different indifference classes, and $\leq_w$ is a partial order that encodes the promise between the elector and C_w.

Let r_i and r_j respectively be routes in indifference classes R_i and R_j (and hence $r_i \neq r_j$) that are (resp.) sent in step one of the protocol. The elector will select $r_q \in \{r_i, r_j\}$.

If $r_q = r_i$, then the elector breaks its promise to C_a since $r_i \leq_a r_j$. Otherwise, if $r_q = r_j$, then the elector breaks its promise to C_b since $r_j \leq_b r_i$, completing the proof.

□

Note that by Theorem 1, at least one correct neighbor will detect the broken promise. Additionally, from the case analysis of Theorem 3, the consumer whose promise has been broken can broadcast a PROOFCHALLENGE message that will cause correct ASes to detect that the elector is faulty (i.e., makes inconsistent promises).

References

[1] A. Haeberlen, P. Kuznetsov, and P. Druschel. PeerReview: Practical accountability for distributed systems. In *Proceedings of the Symposium on Operating Systems Principles (SOSP'07)*, Oct. 2007.

[2] M. Zhao, W. Zhou, A. Gurney, A. Haeberlen, M. Sherr, and B. T. Loo. Private and Verifiable Interdomain Routing Decisions. In *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 2012.