APPENDIX A
MONOSAT-BASED MODEL

A.1 MonoSAT Solver

Intuitively, whether the VPC meets reachability and security attributes can be converted to SMT problems. MonoSAT[1] is an SMT solver designed to verify the relevant properties of the graph, e.g., reachability. For many significant graph constraints, MonoSAT shows order-of-magnitude speed-ups and greatly improved scalability over MiniSAT, CLASP, and Z3 [1]. Specifically, the network reachability can be modeled as a symbolic graph of network components, along with symbolic packet headers. The symbolic graph consists of a set of nodes and directed edges, where the edges may be traversable or untraversable. Nodes represent components and edges represent possible paths that packets may go through these components. The functions and constraints of components are modeled as logical expressions and associated with the corresponding edges. The constraints between edges and packet headers define the packet's routes. For example, we encode an edge between one ECS instance node, ECS1, and its containing subnet node Subnet1, $edge(\text{ECS1}, \text{Subnet1})$. As shown in Fig. 1, we encode a constraint that force $edge(\text{ECS1}, \text{Subnet1})$ to be false if the packet's source address does not match the ECS instance's IP address. This ensures that packets leaving the ECS instance must have their IP address be their source address. A similar constraint ensures that packets entering the ECS instance must use that ECS instance's IP address as their destination address.

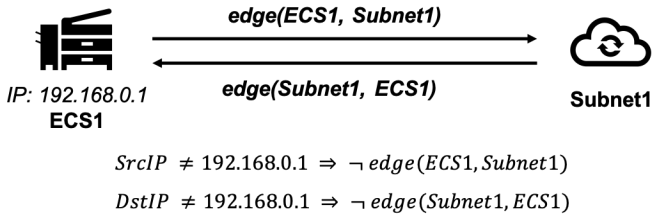

Fig. 1: A small example of MonoSAT.

A.2 Basic Encoding

When components process packets, they apply rules that match the packet according to the designed strategy. For example, security groups and ACLs allow or deny packets based on their priority. Routers transmit packets based on the longest matching principle and the routing table. All these components can be converted into a logical expression in the same way. Specifically, although ACL uses the first matching rule, it can be considered as a combination of all relevant rules because the priority is not repeated. For routing tables, we take the subnet mask number of CIDR as the priority parameter. The larger the subnet mask value, the higher the priority of the routing rule, which corresponds to the longest prefix matching.

Algorithm 1 shows the encoding process, where the $union$ operation aggregates the rules in $Rule$ according to their priority parameters. Line 7 records the scope of packets processed under each priority, and lines 8-11 implement statistical analysis on allowed and rejected packets, respectively. Lines 15-16 aggregate different priority rules, ensuring that low-priority rules do not affect high-priority rules. The result $restrict$ includes all restrictions for packets that the current component allows.

Algorithm 1: Basic encoding in MonoSAT

```
   Input: A set of rules R
   Output: Logical expressions Restrict
1  Let r_0, ··· , r_n be the rules in the R;
2  For each r, it includes elements {Pri, CIDR, Pro, Port, Act};
3  Union = union(R);
4  for p, u ∈ Union do
5      record = {};
6      for j ∈ p do
7          record['Domain'] = record['Domain'] ∧ (IP ∈
             CIDR ∧ port ∈ Port);
8          if j.Act =='Allow' then
9              record['A'] = record['A'] ∨ (IP ∈ CIDR ∧ port ∈
                 Port);
10         else
11             record['D'] = record['D'] ∧ ¬(IP ∈ CIDR ∧ port ∈
                 Port);
12     Union[p] = record;
13 record = None;
14 for p, u ∈ sorted(Union) do
15     Restrict = ¬record ∧ u['Domain'] ∧ u['A'] ∧ u['D'];
16     record = record ∨ u['Domain'];
17 return Restrict
```

A.3 Extended Encoding

Some components may modify and retransmit packets, which cannot be easily expressed through logical expressions. To solve this, we use segmentation to calculate packet reachability. We divide the component into multiple copies with separate packet header variables. For example, we can extend the encoding of CLB and NAT as shown in Fig. 2. In the case of CLB, only the destination IP and port are modified, so there is only one splitting layer shown in Fig. 2(a). However, for NAT, which involves both SNAT and DNAT, there is a fully connected two-layer network shown in Fig. 2(b). To handle situations where not every packet triggers both SNAT and DNAT, we introduce two white nodes to transmit such packets. More specifically, if a packet passing through the NAT gateway does not match any of the rules in the SNAT (DNAT), it will not be modified by the NAT gateway. Activating different component copies will trigger different packet modifications, which will affect the packet used in the next stage. According to Fig. 2, there are multiple groups of servers that are responsible for handling incoming traffic based on specific criteria, such as the port number of the TCP packet. If the port number of the TCP packet is 22 (which is often used for SSH connections), the next phase of the load balancing process would be directed towards the range of servers within Group 1.

It is important to note that the current models based on MonoSAT cannot provide state tracing, thus making them unsuitable for verifying consistency usability and consistency reachability.

1. https://www.cs.ubc.ca/labs/isd/Projects/monosat/

REFERENCES

[1] S. Bayless, N. Bayless, H. Hoos, and A. Hu, "Sat modulo monotonic theories," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

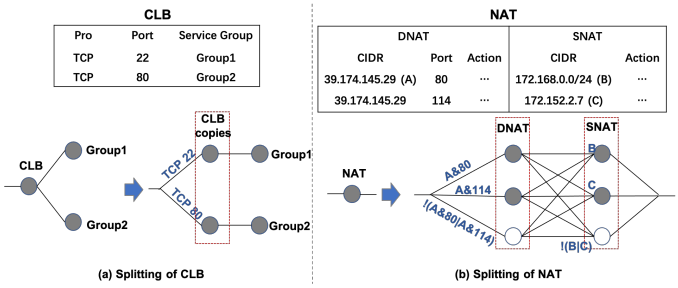

Fig. 2: Splitting of the CLB and NAT.