Automated Attack Graph Generation

Dixit Kumar (18111405)

1. INTRODUCTION

Over the years, cyberspace has also become an important constituent of national power and as the advancement in cyber-attack is increases drastically day by day, we need some mechanism to generate attack graph from different possible attack scenario.

This project is mostly focus on how to convert a threat model into finite state model, to encode various security policies into invariant and to generate attack graph automatically and efficiently.

2. DESCRIPTION

Symbolic model checking can be used to generate attack graph in an efficient automated way. This section will describe how exactly symbolic model checking helps us in generating attack graph.

2.1 Bounded Model Checking

Bounded Model Checking (BMC) is a technique in which we exhaustively traverses all the reachable states of the system to verify the desired properties. In BMC we construct a Boolean formula which is satisfiable iff the underlying state transition system can realize a finite sequence of state transitions that reaches certain states of interest. In case if the desired property in not satisfiable, it gives a counterexample. Bounded Model checking is based on SAT procedure.

$$Init(S_0) \wedge R(S_0,S_1) \wedge R(S_1,S_2) \wedge \neg P(S_2)$$

above boolean formula can be given to SAT Solver, which inn case of satisfiablity gives a assignment to support this formula, where

$$S - set\ of\ states$$
$$R - S \times S$$
$$P - desired\ property$$

2.2 Finite State Model

Let n_size be number of nodes in network, and each node is represented as set of various attributes.

$$node - \{id, type, unsafe, critical\}$$

type of node could be pc, mobile, server, firewall or any typical network component; unsafe parameter tells whether node is in compromised state or not, critical parameter states the importance of that node.

Edges in network can be represented as,

$$edge - \{(node_i, node_j), true\}$$

2.3 State Transition and Safety Property

In BMC, at each transition step (unrolling), we are creating another set of variable to store modified unsafe value. At every transition step, unsafe value get modified as according to these constraint,

$$unsafe'_i = If\ ((\exists j)\ unsafe_j \wedge edge_{ji})\ \{true, unsafe_i\}$$

Unsafe updation can be explained as, if there exist any neighbour *node_j* to *node_i*, such that *node_j* is in unsafe state then it will make *node_i* also to be in unsafe state, else *node_i* will hold same value as it has in previous transition state.

Cyber-attack on netowork can be detected using safety property. Safety property states that if there exists any node in network which is critical in nature and currenlty in compromised (unsafe) state, then network is surely be in attack.

$$safety = (\exists i)\ is\_critical_i \wedge unsafe_i)$$

Invariant related to safety property can be extended depending upon the requirements.

3. EVALUATION

In our typical network configuration, below figures

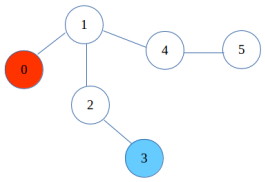

Figure 1: base step

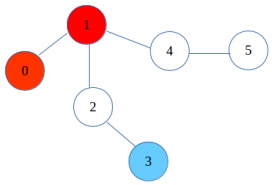

Figure 2: base step

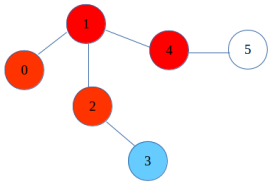

Figure 3: base step

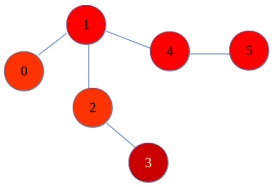

Figure 4: base step

explain the evaluation of attack with transition state, red color is used to mark comprised node, blue color is used to mark critical node.

In case when network is under attack then, the SAT Solver will return sat with possible assignment to node which helps us in generating one of possible attack graph, as shown in figure 5.

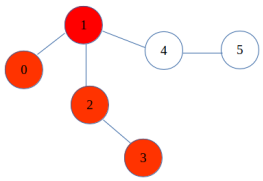

Figure 5: Attack Graph

So, inorder to get all possible attack graph at each unrolling step, we added one extra invariant, untill we get all possible attack graph.

4. FUTURE WORK

There are many possibilities of improvement in this work. Strong Threat model has to implement by including more number invariant. Number of edges and there respective connection in network can be updated dynamic, like in case of an attack, attacker can easily creates, delete connection in the network. Various different attack scenario,with different system configuration (e.g. network connectivity, firewall rules etc) need to be considered and according to it, different type of constraint has to be added.

For ease of end-user, well-designed graphical interface can be used along with Automated Graph. This will helps end user, like network administrator to get better insights, also it will enhance network administrator capability to enforce various security policies.

5. ACKNOWLEGEMENT

I specially thank Pramod Subramanyan for continued inspiring discussion on this project. I also thanks Sandeep Shukla for giving me a chance to work on this project. [1]

[1]https://github.com/node104/AttackGraph-2018