Ideas Surrounding Protocol Orderings *

Anna Fritz

Institute for Information Sciences
The University of Kansas
Lawrence, KS 66045
{authors}@ku.edu

Abstract. The abstract should briefly summarize the contents of the paper in 15–250 words.

Keywords: First keyword · Second keyword · Another keyword.

1 Goals

In this work we aim to introduce a means to produce some protocol ordering. When comparing two protocols, say a and b, we hope to be able to say

1. a $\leq$ b & a $\neq$ b $\implies$ a $<$ b
2. a $\leq$ b & a $=$ b $\implies$ a $=$ b
3. a & b are incomparable

2 Mathematical Concepts

2.1 Basics

Critical to our study is various relations R on a set X. We specifically consider the following relations:

- *reflexive* – forall $x \in X$, R x x
- *symmetric* – forall $x, y \in X$, R x y $\rightarrow$ R y x
- *transitive* – forall $x, y, z \in X$, R x y $\rightarrow$ R y z $\rightarrow$ R x z
- *antisymmetric* – forall $x, y \in X$, R x y & R y x $\rightarrow$ x $=$ y
- an *equivalence* relation is reflexive, symmetric, and transitive
- a *partial order* is reflexive, antisymmetric, and transitive
- a *preorder* is reflexive and transitive

We also may need to consider properties of functions. A function from set X to set Y is written $f : X \rightarrow Y$. Here X is the domain while Y is the codomain. A function may also be considered a relation for pairs (x, y) where $f(x) = y$. *Partial* functions may be undefined for elements in the domain while a *total* function is defined for all elements of the domain. We can say various things about properties of functions. We say a function $f : X \rightarrow Y$ is

* Supported by organization x.

- *injective* – forall $x, y \in X$, $f(x) = f(y) \rightarrow x = y$
- *surjective* – forall $y \in Y$, there exists some $x \in X$ with $f(x) = y$
- a *bijective* function is both injective (one-to-one) and surjective (onto)

2.2 Mathematical Structures

In this work, we are mostly concerned with directed, labeled, acyclic graphs as defined below.

Definition 1 (Directed graph). *A directed, labeled, acyclic graph is a tuple $G = (N, E, \ell)$ where N is a finite set of nodes, E is a finite set of edges represented as an edge relation $(N \times N)$, and ℓ is an node labeling function $\ell : N \rightarrow L$.*

Definition 2 (Transition System). *A transition system is a pair (S, T) where S is the set of states and T is the transition relation between states $(S \times S)$. We say some state p transitions to q if and only if $(p, q) \in T$.*
The transition from p to q is denoted $p \rightarrow q$.

what difference between directed graphs and labeled transition systems?

One may easily see that transition systems resemble graphs. In this way, we may say that two transition systems are equivalence if they are isomorphic (there exists a bijection between the states and transitions).

If the transitions have labeled, then the transition system becomes a labeled transition system.

Definition 3 (Labeled Transition System). *A labeled transition system is a tuple (S, L, T) where S is the set of states, L is the set of labels, and T is the transition relation between states $(S \times L \times S)$. We say some state p transitions to q with label a if and only if $(p, a, q) \in T$.*
The transition from p to q with label a is denoted $p \xrightarrow{a} q$.

A Kripke structure is another variation of transition systems. In this model, the nodes are reachable states and the edges are the state transitions. The labeleing function of a Kripke structure maps the nodes to the set of properties that hold within the state.

Definition 4 (Kripke Structure). *A Kripke structure is a 4-tuple (S, I, L, T) where S is the set of states, I is the set of initial states, L is a labeling from $S \rightarrow Prop$, and T is the transition relation between states $(S \times S)$. The transition relation is left-total meaning $\forall s \in S, \exists s' \in S$ such that $(s, s') \in R$*

2.3 Properties over structures

Definition 5 (Homomorphism). *A (graph) homomorphism $\eta : G \rightarrow H$ between graphs $G = (N_G, E_G, l_G)$ and $H = (N_H, E_H, l_H)$ is a function $\eta : N_G \rightarrow N_H$ on the nodes such that for every edge $(n_1, n_2) \in E_G$, $(\eta(n_1), \eta(n_2)) \in E_H$, and for every node $n \in N_G$, $l_G(n) = l_H(n)$. [1]*

A homomorphism is a relation between two graphs, G and H, where two graphs are homomorphic if there exists some function which maps the nodes of G to the nodes of H such that all edges in G have a corresponding edge in H and all labels in G have related labels in H. To prove two graphs are homomorphic, one must prove there exists a function not necessarily that every function produces a homomorphism. Therefore, to prove a homomorphism does not exists, one must prove no functions exists.

A homomorphism bestows a preorder i.e. it is reflexive and transitive. If the homomorphism is injective, then the preorder is a partial order (up to an isomorphism). This is because injective homomorphisms imply an isomorphism.

Definition 6 (Supports/Covers). *Given two sets of graphs S and T, we say that S supports T iff for every $H \in T$, there is some $G \in S$, such that $G \leq H$. We say that T covers S iff for every $G \in S$ there is some $H \in T$ such that $G \leq H$.[1]*

Rowe defines supports and covers to introduce a relation over sets of graphs. Supports states that, for every graph in T, there exists some homomorphism from a graph in S to a graph in T. Conversely, covers says that, for every graph in S, there exists a homomorphism to some graph in T. We imagine we can apply the ideas of supports and covers to compare sets of graphs using relations besides homomorphisms.

Through our study, we found that a homomorphism was not the correct way to capture the relationship between attack graphs.

Bisimulation is useful notion of equivalence which states that two processes are equal if they have the same behavior [2]. This research relies on labeled transition systems so we define bisimilarity over such systems.

Definition 7 (Bisimilarity). *A relation R is a bisimulation if, whenever $P \Re Q$, for all μ we have:*

- *forall P' with $P \xrightarrow{\mu} P'$, exists Q' such that $Q \xrightarrow{\mu} Q'$ and $P' \Re Q'$*
- *forall Q' with $Q \xrightarrow{\mu} Q'$, exists P' such that $P \xrightarrow{\mu} P'$ and $P' \Re Q'$*

Bisimilarities are an equivalence relation. That is, they are reflexive, symmetric, and transitive.

Definition 8 (Similarity). *A relation R is a simulation if, whenever $P \Re Q$, for all μ we have:*

- *forall P' and mu with $P \xrightarrow{\mu} P'$, exists Q' such that $Q \xrightarrow{\mu} Q'$ and $P' \Re Q'$*

Similarity mirrors the definition of bisimilarity sans the second clause. By this definition, similarity is a one way relation, akin to $\leq$ where as bisimilarity is akin to equivalence.

Processes may have internal actions. To account for this, we can manipulate the transition system to include internal actions with a special label. We call these actions *weak*. To abstractly reason about this type of system, we must

instantiate *weaktransition* relations. These relations are typically denoted $\Rightarrow$ and $\stackrel{\mu}{\Rightarrow}$ where $P \Rightarrow Q$ implies P evolves to Q by preforming some, possibly zero, internal actions. $P \stackrel{\mu}{\Rightarrow} Q$ communicates P evolves to Q by performing action μ.

In the typical weak transition relation, the transitions between states are either labeled or silent. However, in our case, the states themselves are either labeled or silent.

Definition 9 (Weak Bisimulation).

3 Examples

We motivate this work specifically in the context of attack trees.

3.1 Chase Model Finder

Protocol Name	Actual protocol
sys	*target: @p4 (vc p4 sys)
vc sys seq	*target: @p3 [(a1 p4 vc) + < + @p4 (vc p4 sys)]
ker vc sys seq	*target: @p4 [(ker p4 vc) + < + @p4 (vc p4 sys)]
rtm ker vc sys seq	*target: @p1 [(rtm p4 ker) + < + @p4 [(ker p4 vc) + < + @p4 (vc p4 sys)]]

We hypothesize that measuring more system components is better. Therefore, we wish to say vc sys seq supports sys.

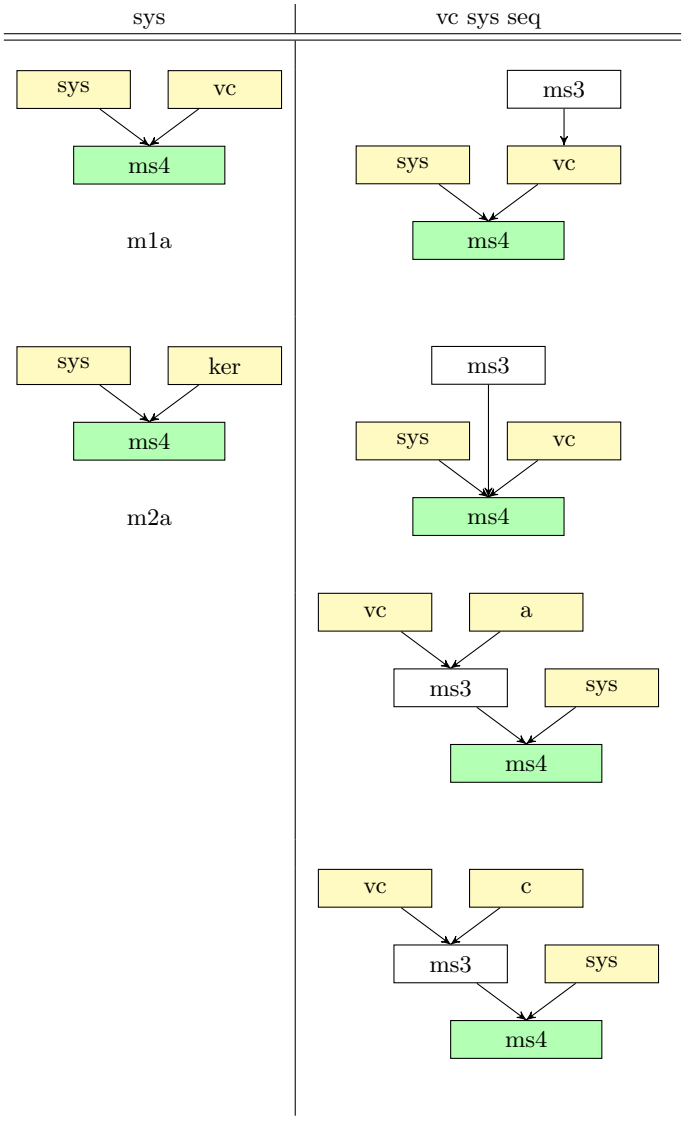

Fig. 1: All attack graphs for two protocols

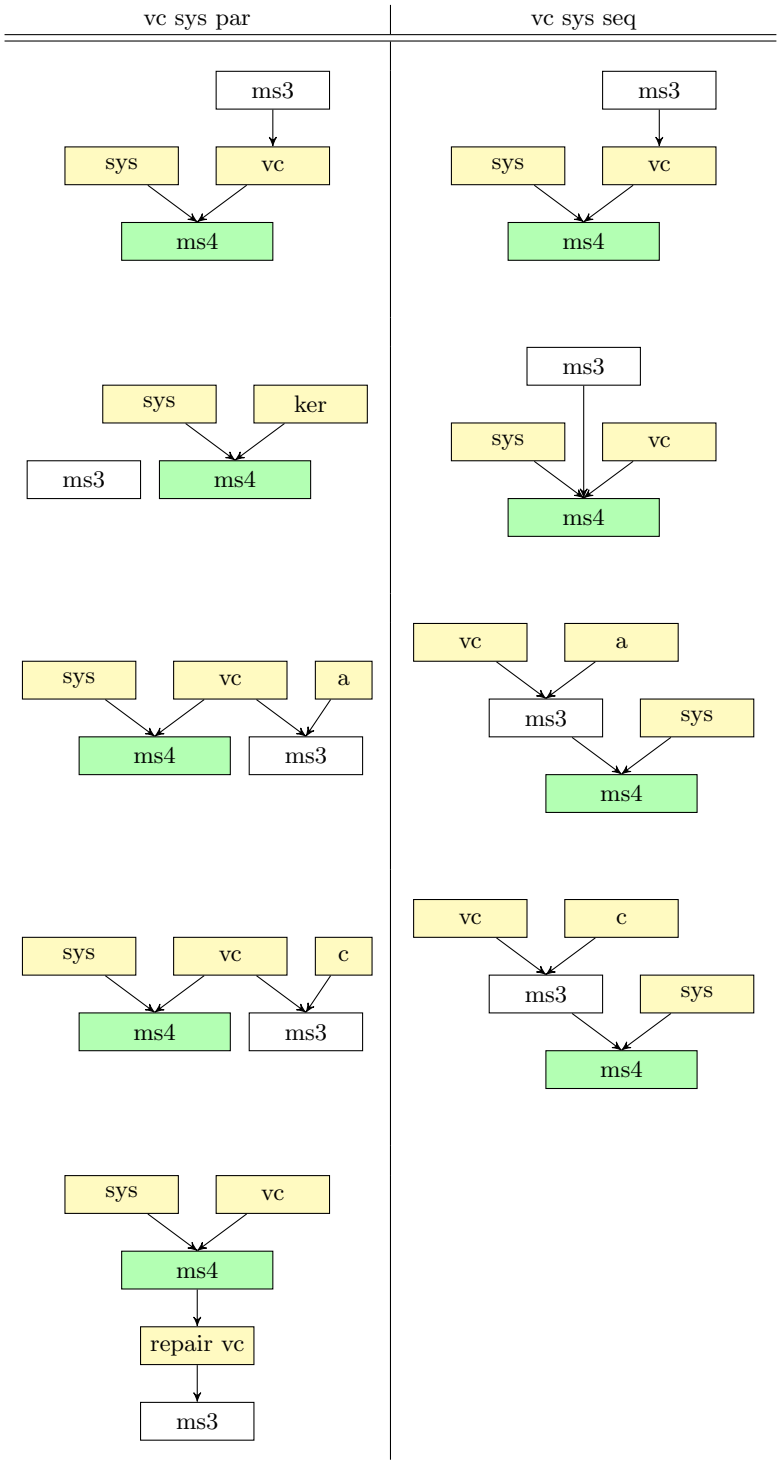

Fig. 2: All attack graphs for two protocols

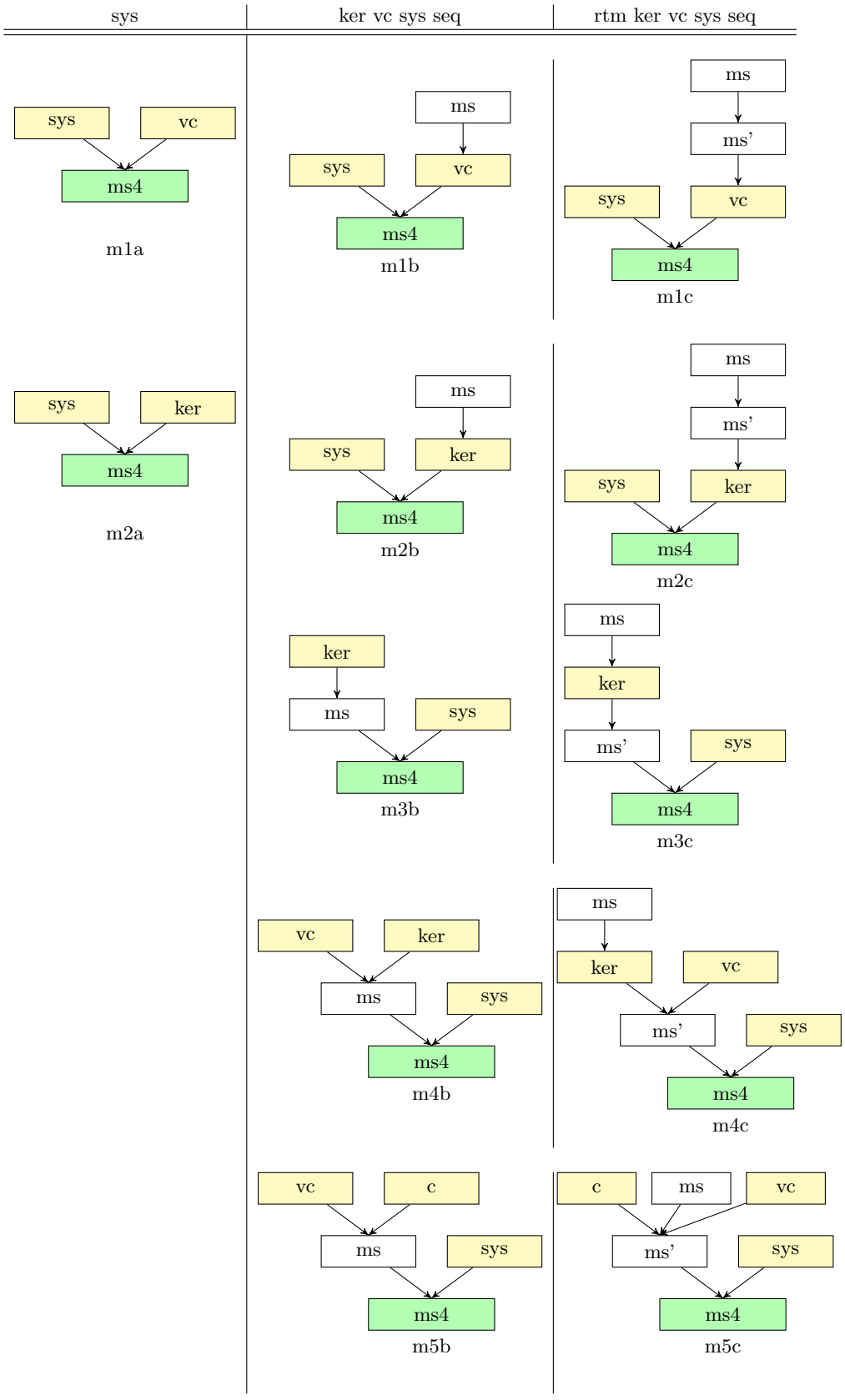

Fig. 3: All attack graphs for two protocols

Bibliography

[1] Rowe, P.D.: On Orderings in Security Models, pp. 370–393. Springer International Publishing, Cham (2021), https://doi.org/10.1007/978-3-030-91631-2_21

[2] Sangiorgi, D.: Introduction to Bisimulation and Coinduction. Cambridge University Press (2011)