AN APPROACH TO INCORPORATING UNCERTAINTY IN NETWORK SECURITY ANALYSIS

BY

HOANG HAI NGUYEN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Adviser:

Professor David M. Nicol

ABSTRACT

Attack graphs used in network security analysis are analyzed to determine sequences of exploits that lead to successful acquisition of privileges or data at critical assets. An attack graph edge corresponds to a vulnerability, tacitly assuming a connection exists and tacitly assuming the vulnerability is known to exist. In this thesis, we explore use of *uncertain graphs* to extend the paradigm to include lack of certainty in connection and/or existence of a vulnerability. We extend the standard notion of uncertain graph (where the existence of each edge is probabilistically independent) however, as significant correlations on edge existence probabilities exist in practice, owing to common underlying causes for disconnectivity and/or presence of vulnerabilities. Our extension describes each edge probability as a Boolean expression of independent indicator random variables. This thesis (i) shows that this formalism is maximally descriptive in the sense that it can describe any joint probability distribution function of edge existence, (ii) shows that when these Boolean expressions are monotone then we can easily perform uncertainty analysis of edge probabilities, and (iii) uses these results to model a partial attack graph of the Stuxnet worm and a small enterprise network and to answer important security-related questions in a probabilistic manner.

In loving memory of my grandmother, Le T. Ngo (1925-2018).

ACKNOWLEDGMENTS

I would like to express my gratitude to my thesis advisor Professor David M. Nicol for his support, guidance, and insightful remarks through the learning process of this master's thesis. Professor Nicol granted me the freedom to explore my own research interest, but steered me in the right the direction whenever he thought I needed it. During my time at UIUC, I have learned a lot from him.

Furthermore, I would like to thank my research group which includes Rakesh Kumar, Kartik Palani, Vignesh Babu, and Elizabeth D. Reed, for the invaluable discussions surrounding my research topic. Thank you for sharing the wonderful times both at work and outside.

Last but not least, I would like to thank my father Tuan V. Nguyen, my mother Xuan T. Hoang, and my younger sister Duong T. Nguyen, for their unconditional love and support. Thank you for giving me the opportunity to pursue my dream through higher education.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY OF NOTATIONS

Symbols	**Definitions**
V	set of vertices
s, t	vertices in V, start and end point of attack
n	size of V
E	set of edges
m	size of E
G	deterministic graph
$E(G)$	set of edges in G
Γ_V	set of all det. graphs with vertex set V
N	size of Γ_V which is $2^{n(n-1)}$
p	probability assignment vector
X	set of random variables
r	size of X
$\wedge, \vee, \neg$	logic operator AND, OR, NOT
q	function that assigns Boolean exp. to edges
$\mathcal{G}$	uncertain graph (basic and extended)
$w_{G,\mathcal{G}}$	the probability of G in $\mathcal{G}$
f	stochastic mapping
$R_{s,t}$	reachability of deterministic graph
$\mathcal{R}_{s,t}$	reachability of uncertain graph
$[0,1]^m$	unit hypercube of dimension m
$\mathcal{H}_{p,\epsilon}$	hyperrectangle containing p

CHAPTER 1

INTRODUCTION

As computers become more ubiquitous in critical infrastructures, evaluating the effect of vulnerabilities becomes increasingly important. In order to make decisions about defense measures, it is vital to study the paths that an attacker might take to intrude into a target network and disrupt services. The attack graph formalism [1] is a representation of the possible ways in which an attacker can get to the desired target host by exploiting vulnerabilities on network hosts while gaining the required privileges at each step. The first step in attack graph generation is analyzing the connectivity of the network components and is termed *reachability analysis* [2]. This information is used to determine if a target host is reachable by an attacker from his current host. Ideally, information about the network topology of the target network, applications running on network hosts, access control rules for the network, and the trust relationships between hosts is known to the modeler. Accuracy and exhaustiveness of network configuration information directly affects accuracy of the generated attack graph [3].

Despite being a useful and well-developed tool, attack graphs have deterministic semantics and hence are not capable of expressing uncertainty [4], which is inherent to any model. For our interests, uncertainty arises mainly from three sources: there is the uncertainty about (i) the attacker (e.g. his skill set, goal, and knowledge), (ii) the system being modeled (e.g. the versions and configuration details of network services and their associated vulnerabilities), and (iii) the environment in which the system is operated (including the legitimate users and administrators). In each category, uncertainty may also come in different shapes, due to either the lack, inadequacy, inaccuracy, or sometimes inconsistency of information. Ideally we should be able to use an attack graph not only to identify possible pathways of attack but also to quantify uncertainty about those pathways.

This thesis aims to integrate uncertainty into security modeling and analy-

sis of computer systems. As a first step, we choose to focus only on studying uncertainty about the system. Hence, uncertainties about the attacker and the environment (and their implications) will not be considered. Under this setup, we propose to use uncertain graphs – graphs where potential edges are labeled with an existence probability. Uncertain graphs have been successfully applied to various problems in different domains [5], [6], [7], [8] while being deeply rooted in network reliability [9]. We use them to analyze uncertainty of the existence of stepping stone attacks encoded in data structures like attack graphs. However, the usual definition of *uncertain graph* assumes edges exist independently of each other [10], [11], [12], a major issue when applied to security modeling, e.g., as one vulnerability may simultaneously enable attacks from and/or to multiple hosts. Furthermore, existing uncertain graph research does not consider the precision of connectivity subjected to change or to uncertainty in edge existences; in other contexts, uncertainty analysis tells us in what cases a robust conclusion can be made in the face of model input uncertainty.

A major portion of this work aims to address these two issues. For the first issue, we extend the uncertain graph formalism and model the correlation between edge existence due to common underlying causes. We describe common causes using independent Bernoulli random variables and use Boolean expressions of the random variables to express the edge existence probabilities. For the second issue, we show how uncertainty analysis of uncertain graphs can be easily done when the Boolean expressions are monotone [13], i.e. they do not use negation of random variables. In summary, our contributions are fourfold:

1. To the best of our knowledge, we are the first to propose uncertain graphs for security modeling and analysis of systems with uncertainty.

2. We extend the traditional uncertain graph formalism to model the correlation between edge existence and prove theoretical results about the expressiveness of uncertain graphs.

3. We perform uncertainty analysis of uncertain graphs by leveraging the monotonicity of reachability.

4. We show how to use uncertain graphs to model systems with uncertainty and how the graphs help answer different security-related ques-

tions about the modeled systems in a probabilistic manner.

The rest of the thesis is organized as follows. Chapter 2 discusses the related work, Chapter 3 discusses background on uncertain graphs, Chapter 4 extends the uncertain graph formalism and proves some theoretical results, Chapter 5 describes uncertainty analysis of uncertain graphs, Chapters 6 and 7 show two case studies with numerical results, and Chapter 8 concludes the thesis.

CHAPTER 2

RELATED WORK

2.1 Uncertain graphs

Uncertain graphs, also known as probabilistic graphs, have been applied to modeling of problems from various domains like interaction between proteins using noisy and error-prone experimental data [5], entity resolution for inexact machine learned models [6], optimal reachability in intermittently connected networks with known routing algorithms [7], path queries on road networks with unexpected traffic jams [8], and many others. The power of uncertain graphs comes from their capability of modeling systems with uncertainty, whether due to lack of knowledge about certain parts of the systems [7], [8] or to noisy model data [5], [6]. Uncertain graphs are deeply rooted in classical network reliability [9], [14], [15].

Reasoning with uncertain graphs is challenging since most problems in uncertain graphs are computationally hard. For example, counting the number of possible worlds of an uncertain graph in which vertex s reaches vertex t is #P-complete (ST-CONNECTEDNESS [15]). Potamias et al. [16] derived sampling-based approximation algorithms for the k-nearest neighbor problem of uncertain graphs. Jin et al. [10] formulated the distance-constraint reachability (DCR) problem and introduced efficient recursive sampling schemes to estimate DCR of large uncertain graphs. Khan et al. [11] studied reliability search problems of uncertain graphs, i.e. finding all vertices reachable from some query vertices with probability no less than a given threshold, using RQ-tree. Recently, [12] proposed recursive stratified sampling-based estimators to reduce the variance of the standard Monte Carlo approach in estimating uncertain graph properties.

2.2 Scenario graphs

The operation of systems can be modeled to be in different states at different instants of time. While most states might be benign, there exist critical states that can lead the system to failure. A failure scenario is described as a sequence of events that violate a correctness property defined for the system. A scenario graph [17] is an exhaustive and succinct representation of all failure scenarios. A special case of the scenario graph is an attack graph.

2.3 Attack graphs

An attack graph models the possible ways an attacker might get access to a critical asset by exploiting a set of vulnerabilities on the services running on the hosts. The vertices of the graph represent the privilege levels of the attacker on the network hosts, and the edges represent the vulnerabilities that the attacker could exploit [18]. Traditionally, teams of experts have looked at the services running on hosts to determine vulnerability information and have coupled this with network information, such as the connectivity of hosts, to build out these attack graphs. Due to the manual nature of the construction of such attack graphs, they are prone to error and often not exhaustive. Automated attack graph generation using model checking was introduced by Ritchey and Ammann [19]. The model check, however, provided just a single attack scenario. Sheyner et al. [20] use model checking on heterogeneous networks to provide an exhaustive list of attack scenarios. A more scalable solution for larger networks has been proposed in [21]. Another optimization using the monotonicity property has been proposed by Ammann et al. [22]. Work in [23] and [24] uses Bayesian networks to capture the uncertainty of information in attack graphs. However, we believe that the acyclic nature of Bayesian networks limits their ability to model the possible cyclic relationships that arise in many practical situations.

Another related aspect is the process of reachablity analysis. Reachability analysis of a network investigates the conditions under which a target host can be reached by an attacking host. Network scanners [25] and vulnerability discovery tools [26] can be leveraged to derive the configuration of the target network.

CHAPTER 3

UNCERTAIN GRAPHS

3.1 Overview of uncertain graphs

Uncertain graphs extend the definition of a deterministic graph by ascribing to each of a deterministic graph's edges an existence probability [10], [12], [16]. Formally, let $G = (V, E)$ denote a deterministic graph[1] where $V = \{V_1, \ldots, V_n\}$ and $E = \{E_1, \ldots, E_m\}$ are the set of vertices and edges. The uncertain graph $\mathcal{G} = (V, E, p)$, where $p = (p_1, \ldots, p_m) \in [0, 1]^m$, allows each edge $E_i \in E$ to exist independently of the others and with probability p_i for $i = 1, \ldots, m$. We call p the probability assignment vector of E. An uncertain graph may contain both certain edges – edges that exist with probability zero or one – and uncertain edges – edges that exist with probability strictly greater than 0 and less than 1. When all edges are certain edges, the uncertain graph degenerates to a deterministic graph. In the literature, uncertain graphs are sometimes treated as generative models of deterministic graphs [10], [16]. In this view, every deterministic graph $G = (V, E')$ where $E' \subseteq E$ is called a possible world (or possible outcome) of $\mathcal{G}$. Slightly abusing the notation, we denote this as $G \in \mathcal{G}$. Similar to saying a fair coin generates the head outcome with probability 0.5, we say the $\mathcal{G}$ generates G with probability:

$$w_{G,\mathcal{G}} = \prod_{E_i \in E'} p_i \prod_{E_i \in E \setminus E'} (1 - p_i) \tag{3.1}$$

The uncertain graph $\mathcal{G}$ generates an exponential number of 2^m possible worlds, out of which only $2^{m'}$ exist with nonzero probabilities, where $m' \leq m$ is the number of uncertain edges in $\mathcal{G}$. For example, the probability of G_i in Figure 3.1 is $w_{G_i,\mathcal{G}} = p_1 p_2 (1 - p_3)(1 - p_4) p_5$. Obviously $w_{G,\mathcal{G}} \in [0, 1]$ for all

[1] We only consider simple directed graphs.

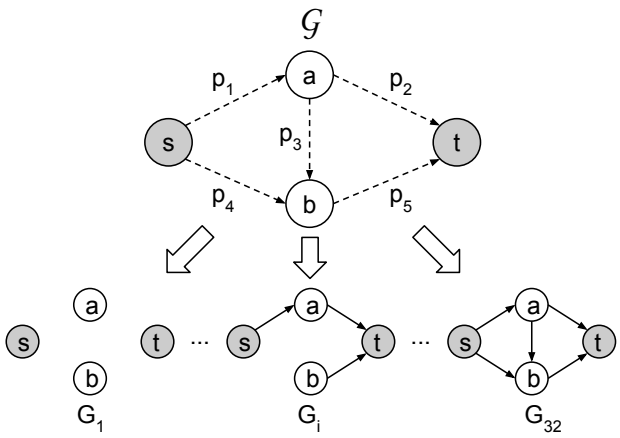

Figure 3.1: A 4-vertex, 5-edge uncertain graph and three of its 32 possible worlds. In security modeling, s denotes the starting point (e.g. a compromised computer in the network) and t the ending point of the attack (e.g. a critical computer that the attacker wants to gain access to).

$G \in \mathcal{G}$ and the law of total probability dictates that $\sum_{G \in \mathcal{G}} w_{G,\mathcal{G}} = 1$.

An uncertain graph is distinguished from a Bayesian network [23], [24], which was originally designed to summarize joint probability distributions. While Bayesian networks are acyclic, cyclic relationships arise from many practical situations and are allowed in uncertain graphs. Wang et al. [23] circumvented the problem with cycles, but the technique relies on metric-dependent properties. Uncertain graphs also do not assume the state transition modeled in transition systems [27] (e.g. Markov decision processes, probabilistic automata.) Such transitions have a subtle drawback in security modeling of computer networks since an attacker does not "jump" from one place to the other. Instead, he gains access to more and more places as the attack progresses and is capable of showing up at multiple places at the same time.

3.2 Reachability of uncertain graphs

For any given graph property, a deterministic graph either has the property or does not have it. Since edges in uncertain graphs are random, we will speak of the probability that an uncertain graph has a given property, as the sum of the probabilities of its possible worlds that possess that property. We are particularly interested in reachability.

Using mathematical symbols, given a deterministic graph G and two vertices s and t, let function $R_{s,t}(G)$ denote the s,t-reachability of G, formally:

$$R_{s,t}(G) = \begin{cases} 1, & \text{if there is a path from } s \text{ to } t \text{ in } G \\ 0, & \text{otherwise} \end{cases}$$

For example, in Figure 3.1, $R_{s,t}(G_i) = R_{s,t}(G_{32}) = 1$ and $R_{s,t}(G_1) = 0$. The two vertices s and t remain mostly unchanged throughout the paper. When the context is clear, we will refer to s,t-reachability simply as reachability. Let $\mathcal{G} = (V, E, p)$ be an uncertain graph, the reachability of $\mathcal{G}$ is defined as the reachability of an average possible world G in $\mathcal{G}$ weighted by its probability:

$$\begin{aligned} \mathcal{R}_{s,t}(\mathcal{G}) &= \sum_{G \in \mathcal{G}} w_{G,\mathcal{G}} \; R_{s,t}(G) \\ &= \sum_{G \in \mathcal{G}} \left(\prod_{E_i \in E(G)} p_i \prod_{E_i \in E \setminus E(G)} (1 - p_i) \; R_{s,t}(G) \right) \end{aligned} \tag{3.2}$$

where we use $E(G)$ to denote the set of edges in G. Using Equation 3.2, the reachability of the uncertain graph in Figure 3.1 can be computed as follows (after simplification):

$$\begin{aligned} \mathcal{R}_{s,t}(\mathcal{G}) = {} & p_1p_2 + p_4p_5 + p_1p_3p_5 - p_1p_2p_3p_5 \\ & - p_1p_2p_4p_5 - p_1p_3p_4p_5 + p_1p_2p_3p_4p_5 \end{aligned} \tag{3.3}$$

Although this thesis only focuses on reachability, other graph-related properties can be translated to uncertain graphs in a similar fashion. For example, for a deterministic graph G and two vertices s and t, if we denote $D_{s,t}(G)$ the length of shortest path (in term of hop count) from s to t in G, then we can talk about the *expected* length of the shortest path from s to t in an uncertain graph $\mathcal{G}$, denoted as $\mathcal{D}_{s,t}(\mathcal{G})$, which can be defined similarly to Equation 3.2 as follows:

$$\begin{aligned} \mathcal{D}_{s,t}(\mathcal{G}) &= \sum_{G \in \mathcal{G}} w_{G,\mathcal{G}} \; D_{s,t}(G) \\ &= \sum_{G \in \mathcal{G}} \left(\prod_{E_i \in E(G)} p_i \prod_{E_i \in E \setminus E(G)} (1 - p_i) \; D_{s,t}(G) \right) \end{aligned} \tag{3.4}$$

3.3 Estimating reachability of uncertain graphs

Most problems in uncertain graphs are #P-complete, including the problem of computing reachability [15]. For that reason, sampling techniques have been proposed to solving problems of large uncertain graphs [10], [12], [16], [28] as the alternative to direct computation. A basic Monte Carlo method for estimating the reachability of an uncertain graph $\mathcal{G}$ works as follows. First, sample k possible worlds $G_1, \ldots, G_k$ from $\mathcal{G}$. This can be achieved by sampling edges in $\mathcal{G}$ independently according to their existence probabilities. Then, compute the reachability $R_{s,t}(G_i)$ for each G_i, $i = 1, \ldots, k$. The reachability of the uncertain graph is estimated as:

$$\widehat{\mathcal{R}}_{s,t}(\mathcal{G}) = \frac{1}{k}\left(\sum_{i=1}^{k} R_{s,t}(G_i)\right) \tag{3.5}$$

The estimator $\widehat{\mathcal{R}}_{s,t}(\mathcal{G})$ is a random variable whose mean is $\mathcal{R}_{s,t}(\mathcal{G})$ (for this we say the estimator is unbiased) and variance $\frac{1}{k}\mathcal{R}_{s,t}(\mathcal{G})(1 - \mathcal{R}_{s,t}(\mathcal{G}))$ [10], [28]. Advanced sampling techniques have been proposed to reduce the estimator variance while requiring fewer samples [10], [12]. Those techniques rely on the factor theorem [14] to recursively compute $\mathcal{R}_{s,t}(\mathcal{G})$ by conditioning on the existence of an edge.

CHAPTER 4

EXTENDED UNCERTAIN GRAPHS

While a promising tool, the existing uncertain graph formalism does not support modeling of the correlation between edge existence. Such correlation arises naturally from modeling various systems (Chapter 6 and 7). Here is an example. Assume in a certain network, host 0 and host 1 can freely communicate with all services running on host 1 and host 2, respectively. Furthermore, both host 1 and host 2 run a similar set of services. If an attacker from host 0 can gain access to host 1 by exploiting some vulnerability of a service running on host 1, then surely he is also able to do so from host 1 to host 2. As we model the possibility of attacks in the network using an uncertain graph, edge $(0, 1)$ existence guarantees that edge $(1, 2)$ also exists. In other words, there is no possible world in which edge $(0, 1)$ exists while edge $(1, 2)$ does not. This behavior cannot be modeled using the described uncertain graphs where edges exist independently of one another (Section 4.4). As the result, an altered and more powerful formalism is required.

This chapter is organized as follows. First, we formally define the correlation between edges and extend the basic uncertain graph formalism to model such property (Section 4.1). Next, we show how to compute reachability of extended uncertain graphs using a slightly modified version of Equation 3.2 (Section 4.2). Then, we define the concept of stochastic mapping (Section 4.3) and use it to prove that modeling the correlation between edges indeed expands the expressiveness of basic uncertain graphs, in the sense that there exists an extended uncertain graph that has no equivalent basic uncertain graph (Section 4.4). Lastly, we prove an important result which says that extended uncertain graphs can model an arbitrary stochastic mapping, making the two models equivalent in term of expressiveness (Section 4.5).

4.1 Formal definition

Define a tuple $\mathcal{G} = (V, E, X, p, q)$ where

- $V = \{V_1, \ldots, V_n\}$ and $E = \{E_1, \ldots, E_m\}$ are the set of vertices and edges as before,
- $X = \{X_1, \ldots, X_r\}$ is the set of independent Bernoulli random variables, i.e. $X_i \in \{0, 1\}$ for all $X_i \in X$,
- $p = (p_1, \ldots, p_r) \in [0, 1]^r$ is the probability assignment vector of X, i.e. $p_i = Pr(X_i = 1) = 1 - Pr(X_i = 0)$ for $i = 1, \ldots, r$, and lastly
- q is the assignment function that associates each edge $E_i \in E$ with a Boolean expression of the random variables in X using three Boolean operators AND ($\wedge$), OR ($\vee$), and NOT ($\neg$). The existence of edge E_i directly depends on the evaluation of its Boolean expression $p(E_i)$, i.e. edge E_i exists if and only if $q(E_i) = 1$.

We refer to this tuple as the extended uncertain graph, in contrast to the basic uncertain graph $\mathcal{G} = (V, E, p)$ defined in Section 3.1. By sharing random variable(s) across different Boolean expressions in the assignment function q, dependency between edges can be *explicitly* modeled. An example of an extended uncertain graph is shown in Figure 4.1. When the context is clear, we use the term *uncertain graphs* to refer to both basic and extended uncertain graphs. Every basic uncertain graph $\mathcal{G} = (V, E, p)$ has an equivalent extended uncertain graph representation $\mathcal{G} = (V, E, X, p, q)$, which uses m random variables and $q(E_i) = X_i$ for $i = 1, \ldots, m$; however, as we shall see the reverse is not true.

4.2 Reachability of extended uncertain graphs

To compute the s, t-reachability of an extended uncertain graph, we sum up the probabilities of all of its possible worlds in which s reaches t. Unlike before, however, computing the probability of a given possible world is not so easy since edges no longer exist independently of each other. An easier approach is to iterate through every realization of X_i's, and for each realization construct the corresponding possible world by evaluating the Boolean

Algorithm 1 Computing the reachability of an extended uncertain graph.

Input: $\mathcal{G} = (V, E, X, p, q), s, t$
Output: $\mathcal{R}_{s,t}$

```
1: function REACHABILITY(G, s, t)
2:     R_{s,t} ← 0
3:     for X' ∈ 2^X do
4:         w ← 1
5:         for i ∈ [1, 2, ..., r] do
6:             if X_i ∈ X' then
7:                 X_i ← 1
8:                 w ← w p_i
9:             else
10:                X_i ← 0
11:                w ← w(1 − p_i)
12:        E' ← {E_i ∈ E : q(E_i) = 1}
13:        G_{X'} ← (V, E')
14:        if s reaches t in G_{X'} then
15:            R_{s,t} ← R_{s,t} + w
       return R_{s,t}
```

expression of every edge. As a result, computing the probability of a possible world now reduces to computing the probability of a realization of X_i's, which is simply a product of probabilities since X_i's are independent by definition. Specifically, let X' be a subset of X and $G_{X'}$ the deterministic graph realized from $\mathcal{G}$ by evaluating X_i to 1 if $X_i \in X'$, otherwise $X_i = 0$. It is easy to see that $\mathcal{G}$ generates $G_{X'}$ with probability:

$$\begin{aligned} w_{G_{X'},\mathcal{G}} &= \prod_{X_i \in X'} Pr(X_i = 1) \prod_{X_j \in X \setminus X'} Pr(X_j = 0) \\ &= \prod_{X_i \in X'} p_i \prod_{X_j \in X \setminus X'} (1 - p_j) \end{aligned} \tag{4.1}$$

The reachability of $\mathcal{G}$ can then be computed as

$$\begin{aligned} \mathcal{R}_{s,t}(\mathcal{G}) &= \sum_{X' \in 2^X} w_{G_{X'},\mathcal{G}}\ R_{s,t}(G_{X'}) \\ &= \sum_{X' \in 2^X} \left(\prod_{X_i \in X'} p_i \prod_{X_j \in X \setminus X'} (1 - p_j) \right) R_{s,t}(G_{X'}) \end{aligned} \tag{4.2}$$

where 2^X denotes the superset of X. A straightforward implementation of Equation 4.2 is given in Algorithm 1. For other methods to compute the reachability using cutsets and pathsets, please refer to [9].

4.3 Stochastic mapping

Before proving the expressiveness of basic and extended uncertain graphs, we first define the concept of stochastic mapping. If we consider uncertain graphs as generative models of deterministic graphs, then each uncertain graph defines a mapping from the set of possible worlds to the unit interval $[0, 1]$. Let Γ_V denote the set of all deterministic graphs with vertex set V and $N = |\Gamma_V| = 2^{n(n-1)}$ the size of Γ_V (i.e. we consider all possible directed edges except self-loops). Define a mapping $f : \Gamma_V \to [0, 1]$ that associates with each deterministic graph $G \in \Gamma_V$ a real number $w_{G,\mathcal{G}} \in [0, 1]$. If the mapping f satisfies the condition $\sum_{G \in \Gamma_V} f(G) = 1$, then we call it a stochastic mapping. A stochastic mapping is then a joint probability distribution function over the space of deterministic graphs whose edges are a subset of E. For a given uncertain graph $\mathcal{G}$ and stochastic mapping f defined on the same edge set as $\mathcal{G}$, if $f(G) = w_{G,\mathcal{G}}$ for all $G \in \mathcal{G}$, then we call f the equivalent stochastic mapping of $\mathcal{G}$; we denote that $\mathcal{G} \equiv f$. Every uncertain graph has an equivalent stochastic mapping and two uncertain graphs are equivalent if they have the same stochastic mapping.

4.4 Expressiveness of basic uncertain graphs

In this section, we prove the following theorem:

Theorem 4.4.1. *Extended uncertain graphs strictly expand the expressiveness of basic uncertain graphs, i.e. there exists an extended uncertain graph that has no equivalent basic uncertain graph.*

Proof. We prove this theorem by giving an example. Consider the extended uncertain graph $\mathcal{G}$ in Figure 4.1. It has only two possible worlds G_1 and G_2 with $w_{G_1,\mathcal{G}} = Pr(X_1 = 0) = 0.5$ and $w_{G_2,\mathcal{G}} = Pr(X_1 = 1) = 0.5$. We will show that this extended uncertain graph has no equivalent basic uncertain graph representation. Define the basic uncertain graph $\mathcal{G}' = (V, E, p)$

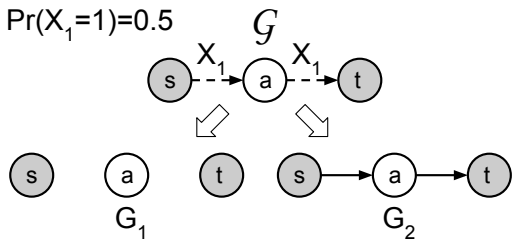

Figure 4.1: An extended uncertain graph and its only two possible worlds with nonzero probabilities, $w_{G_1,\mathcal{G}} = w_{G_2,\mathcal{G}} = 0.5$. This graph does not have an equivalent basic uncertain graph.

where $V = (s, a, t)$, $E = ((s, a), (a, s), (a, t), (t, a), (t, s), (s, t))$ and p some probability assignment vector. The definition of $\mathcal{G}'$ captures all possible basic uncertain graphs that can be constructed from three vertices s, a, t. The probabilities of G_1 and G_2 in $\mathcal{G}'$ are:

$$\begin{aligned} w_{G_1,\mathcal{G}'} &= (1-p_1)(1-p_2)(1-p_3)(1-p_4)(1-p_5)(1-p_6) \\ &= (1-p_1)(1-p_3)Q \\ w_{G_2,\mathcal{G}'} &= p_1(1-p_2)p_3(1-p_4)(1-p_5)(1-p_6) \\ &= p_1 p_3 Q \end{aligned}$$

where $Q = (1-p_2)(1-p_4)(1-p_5)(1-p_6)$ and $0 < Q \leq 1$. Assume by contradiction that $\mathcal{G}'$ produces the same stochastic mapping as $\mathcal{G}$, or equivalently $w_{G_1,\mathcal{G}'} = w_{G_2,\mathcal{G}'} = 0.5$, we have: $0 = w_{G_1,\mathcal{G}'} - w_{G_2,\mathcal{G}'} = (1-p_1)(1-p_3)Q - p_1 p_3 Q = (1-p_1-p_3)Q$. Since $Q > 0$, $1 - p_1 - p_3 = 0$ or $p_1 + p_3 = 1$. Moreover, $1 = w_{G_1,\mathcal{G}'} + w_{G_2,\mathcal{G}'} = (1-p_1)(1-p_3)Q + p_1 p_3 Q = (1 - p_1 - p_3 + 2p_1 p_3)Q = 2p_1 p_3 Q$. Since $Q \leq 1$, $2p_1 p_3 = 1/Q \geq 1$. Combining this with $p_1 + p_3 = 1$, we have $(p_1 + p_3)^2 - 2p_1 p_3 \leq 1^2 - 1 = 0$ or $p_1^2 + p_3^2 \leq 0$, and therefore $p_1 = p_3 = 0$. This solution does not satisfy $p_1 + p_3 = 1$, hence no basic uncertain graph equivalent to $\mathcal{G}$ exists. □

Although extended uncertain graphs strictly expand the expressiveness of basic uncertain graphs, there are cases in which the extended uncertain graph model of the studied system can be reduced to an equivalent basic uncertain graph using simple graph transformation tricks (Section 7.3).

4.5 Expressiveness of extended uncertain graphs

In this section, we show that our definition of extended uncertain graphs is maximally expressive, in the sense that for any stochastic mapping of Γ_V, we can construct an extended uncertain graph whose joint edge existence probability distribution is identically that of Γ_V's stochastic mapping.

Theorem 4.5.1. *Every stochastic mapping has an equivalent extended uncertain graph.*

Proof. Fix the set of vertices V. Let f be a stochastic mapping defined over $\Gamma_V = \{G_1, \ldots, G_N\}$. Define $f^{(i)}$ for $i = 1, \ldots, N$ the following mapping:

$$f^{(i)}(G_j) = \begin{cases} \frac{f(G_j)}{\sum_{k=1}^{i} f(G_k)} & \text{if } 1 \leq j \leq i \\ 0 & \text{if } i < j \leq N \end{cases}$$

Without loss of generality, assume $f(G_1) > 0$ so that every $f^{(i)}$ is well-defined, and moreover, it is a valid stochastic mapping since $\sum_{j=1}^{N} f^{(i)}(G_j) = 1$. Especially, $f^{(N)} \equiv f$. We will show how to iteratively construct an equivalent extended uncertain graph $\mathcal{G}^{(i)}$ of every $f^{(i)}$.

The first step is to show an equivalent extended uncertain graph $\mathcal{G}^{(1)}$ of $f^{(1)}$, a stochastic mapping that maps G_1 to 1 and the rest in Γ_V to 0. Define the extended uncertain graph $\mathcal{G}^{(1)} = (V, E, X^{(1)}, p^{(1)}, q^{(1)})$ as follows:

- V the set of vertices
- E the set of all $n(n-1)$ edges, i.e. $G = (V, E)$ is a complete directed graph
- $X^{(1)} = \{X_1\}$
- $p^{(1)} = (p_1)$ where $p_1 = Pr(X_1 = 1) = 1$
- $q^{(1)}$ works as follows: $\forall E_j \in E$, if $E_j \in E(G_1)$ then $q^{(1)}(E_j) = X_1$, else $q^{(1)}(E_j) = \neg X_1$

It can be easily seen that $\mathcal{G}^{(1)} \equiv f^{(1)}$.

Assume we have constructed $\mathcal{G}^{(i)} = (V, E, X^{(i)}, p^{(i)}, q^{(i)})$ where $X^{(i)} = \{X_1, \ldots, X_i\}$ and $p^{(i)} = (p_1, \ldots, p_i)$ such that $\mathcal{G}^{(i)} \equiv f^{(i)}$ for some $1 \leq i < N$.

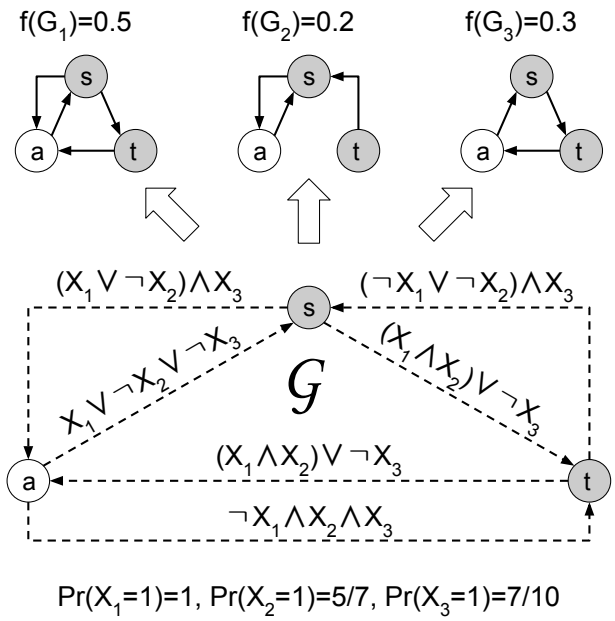

Figure 4.2: Top: a stochastic mapping f and three deterministic graphs with non-zero probabilities $f(G_1) = 0.5$, $f(G_2) = 0.2$, $f(G_3) = 0.3$. Bottom: an equivalent extended uncertain graph $\mathcal{G}$ of f generated using the construction described in Section 4.5.

If $f(G_{i+1}) = 0$ then $f^{(i+1)} \equiv f^{(i)}$. Hence $\mathcal{G}^{(i+1)} = \mathcal{G}^{(i)}$ is the equivalent extended uncertain graph of $f^{(i+1)}$. When $f(G_{i+1}) > 0$, the equivalent extended uncertain graph:

$$\mathcal{G}^{(i+1)} = (V, E, X^{(i+1)}, p^{(i+1)}, q^{(i+1)})$$

of $f^{(i+1)}$ can be constructed as follows:

- V the set of vertices
- E the set of all $n(n-1)$ edges
- $X^{(i+1)} = \{X_1, \ldots, X_i, X_{i+1}\}$ where X_{i+1} is the newly introduced random variable
- $p^{(i+1)} = (p_1, \ldots, p_i, p_{i+1})$ where $p_{i+1} = \frac{\sum_{j=1}^{i} f(G_j)}{\sum_{j=1}^{i+1} f(G_j)}$
- $q^{(i+1)}$ works as follows: $\forall E_j \in E$, if $E_j \in E(G_{i+1})$ then $q^{(i+1)}(E_j) = q^{(i)}(E_j) \vee \neg X_{i+1}$, else $q^{(i+1)}(E_j) = q^{(i)}(E_j) \wedge X_{i+1}$

The construction of $\mathcal{G}^{(i+1)}$ works by scaling down the edge existence probabilities in $\mathcal{G}^{(i)}$ by a factor of p_{i+1} before adding the new graph G_{i+1} with

probability $1-p_{i+1} = 1-\frac{\sum_{j=1}^{i} f(G_j)}{\sum_{j=1}^{i+1} f(G_j)} = \frac{f(G_{i+1})}{\sum_{j=1}^{i+1} f(G_j)} = f^{(i+1)}(G_{i+1})$. The last step of the construction achieves this by first performing a logic AND operation ($\wedge$) between the Boolean expression associated with every edge of $\mathcal{G}^{(i)}$ and the new random variable X_{i+1}, or formally $q^{(i+1)}(E_j) = q^{(i)}(E_j) \wedge X_{i+1}$. Then, for every edge of $\mathcal{G}^{(i)}$ that appears in G_{i+1}, we additionally perform a logic OR operation ($\vee$) between its associated Boolean expression and $\neg X_{i+1}$. The whole purpose of doing so is to force $\mathcal{G}^{(i+1)}$ to generate G_{i+1} with probability $1-p_{i+1}$. Combining these two operations, the Boolean expression associated with every edge of $\mathcal{G}^{(i+1)}$ that appears in G_{i+1} is:

$$\begin{aligned} q^{(i+1)}(E_j) &= (q^{(i)}(E_j) \wedge X_{i+1}) \vee \neg X_{i+1} \\ &= (q^{(i)}(E_j) \vee \neg X_{i+1}) \wedge (X_{i+1} \vee \neg X_{i+1}) \\ &= q^{(i)}(E_j) \vee \neg X_{i+1} \end{aligned}$$

This process allows us to construct an equivalent extended uncertain graph $\mathcal{G}^{(i)}$ of $f^{(i)}$ for $i = 1, \ldots, N$. As the result, $\mathcal{G} = \mathcal{G}^{(N)}$ will be the equivalent extended uncertain graph of f since $f \equiv f^{(N)} \equiv \mathcal{G}^{(N)}$. □

The construction outlined here introduces a new random variable for every deterministic graph that has a non-zero probability in f. Therefore, the total number of random variables used by the final extended uncertain graph is $r = |\{G_i | f(G_i) > 0 \text{ for } i = 1, \ldots, N\}|$. For example, the extended uncertain graph in Figure 4.2 only uses three random variables to model an equivalent stochastic mapping in which only three deterministic graphs have non-zero probabilities G_1, G_2, and G_3. After the first, second, and last iteration of the construction, the Boolean expressions associated with edge (s, a) in $\mathcal{G}^{(1)}$, $\mathcal{G}^{(2)}$, and $\mathcal{G}^{(3)}$ are X_1, $X_1 \vee \neg X_2$, and $X_1 \vee \neg X_2 \vee \neg X_3$, respectively. We notice that both edge (s, t) and (t, a) in $\mathcal{G}$ are associated with the same Boolean expression $(X_1 \wedge X_2) \vee \neg X_3$. This is because (s, t) and (t, a) coexist in all deterministic graphs that have a non-zero probability in f. In general, basic uncertain graphs are not capable of modeling such a behavior.

The main importance of this result is that our particular method for extending uncertain graphs, motivated by a particular need to describe correlation among edges in an attack graph, is capable of describing *any* joint distribution of edge existence probabilities. This is an important foundational result in the theory of uncertain graphs.

CHAPTER 5

UNCERTAINTY ANALYSIS

Uncertainty analysis plays an important role in understanding how uncertainty in model inputs affects its output. While a selection of the probability assignment vector p gives an expression of uncertainty, that expression itself is likely inexact. This is partly because in many cases, p cannot be directly computed or measured and hence some form of estimation is required. When estimation is used, the resulting estimate usually comes with the form of a mean, which is p, and an upper and lower bound. Analyses of uncertain graphs therefore must be applied to p as well as its credible neighborhood so that robust conclusions can be made [29]. Within the neighborhood of p, we are interested in two probability assignment vectors under which the model output, i.e. a property of the uncertain graph like reachability or expected shortest path length, acquires its maximum and minimum value. Those extrema tell us how much we should trust the value in the model output given the uncertainty presented in the model input.

In this thesis, we focus on the reachability property of uncertain graphs (first introduced in Section 3.2). Reachability has an intuitive interpretation in the context of security and forms the basis for answering numerous security-related questions (Section 6.2). From now on, when we talk about uncertainty analysis we will implicitly refer to uncertainty analysis with respect to the reachability of uncertain graphs. In the remainder of this chapter, we first formally define uncertainty analysis as the problem of finding the extrema of the model output (Section 5.1). Then, we show how to quickly identify the extrema using the monotonicity of reachability of the class of monotone uncertain graphs (Section 5.2). Uncertain analysis of uncertain graphs in general is NP-complete by reduction from a 3-SAT problem (Section 5.3).

Remark 1. It is important to note that for the supplied edge existence probability, we never truly know the underlying probability (if one exists)

and do not consider such a value in our model. Instead, the edge existence probability is the numerical representation of our belief (and the bounds our confidence in the number), given the information we have collected and subjected to the assumptions we have made.

5.1 Uncertainty model and monotonicity

Let $\mathcal{G} = (V, E, X, p, q)$ denote an extended uncertain graph and $\mathcal{R}_{s,t}(\mathcal{G})$ the probability that there exists a path from s to t in $\mathcal{G}$. Define $\epsilon = (\epsilon_1, \ldots, \epsilon_r) \in [0,1]^r$ the perturbation vector and $\mathcal{H}_{p,\epsilon}$ the hyperrectangle obtained by perturbing each entry p_i in p by an amount of at most ϵ_i, or formally:

$$\mathcal{H}_{p,\epsilon} = \{p' \in [0,1]^r \mid |p'_i - p_i| \leq \epsilon_i \; \forall i = 1, \ldots, r\} \tag{5.1}$$

The mean, upper and lower bound of estimates described earlier can be modeled using the probability assignment vector and perturbation vector. Uncertainty analysis of uncertain graphs aims to find the extrema of the reachability $\mathcal{R}_{s,t}(\mathcal{G})$ in the hyperrectangle $\mathcal{H}_{p,\epsilon}$ as well as two probability assignment vectors p^{min}, p^{max} in the hyperrectangle where the reachability reaches its extrema, i.e:

$$p^{min} = \underset{p' \in \mathcal{H}_{p,\epsilon}}{\operatorname{argmin}} \; \mathcal{R}_{s,t}(V, E, X, p', q) \tag{5.2}$$

$$p^{max} = \underset{p' \in \mathcal{H}_{p,\epsilon}}{\operatorname{argmax}} \; \mathcal{R}_{s,t}(V, E, X, p', q) \tag{5.3}$$

Here we use the notation $\mathcal{R}_{s,t}(V, E, X, p', q)$ to denote the reachability of the extended uncertain graph $\mathcal{G}' = (V, E, X, p', q)$.

Searching for p^{min} and p^{max} in $\mathcal{H}_{p,\epsilon}$ is nontrivial, in part due to the #P-completeness of computing reachability of uncertain graphs. Fortunately, the monotonicity property of reachability allows us to quickly find p^{min} and p^{max} without having to formulate and solve Equations 5.2 and 5.3 as optimization problems. The monotonicity of reachability in the context of deterministic graphs means (i) adding one or more edges to a deterministic graph does not change its reachability status from 1 to 0 and vice versa, (ii) removing one or more from the graph does not change its reachability status from 0 to 1. The next section extends this property to the class of monotone uncertain

graphs – uncertain graphs whose edges are associated with monotone Boolean expressions – and the implication regarding how to find p^{min} and p^{max}.

5.2 Uncertainty analysis of monotone uncertain graphs

A *monotone* uncertain graph is defined as an extended uncertain graph where the assignment functions are restricted to monotone Boolean expressions [13], i.e. those that only use the AND and OR logic operator and omit the NOT operator. We first start with some observations about extended uncertain graphs in general and monotone uncertain graphs in particular.

Corollary 5.2.0.1. *By restricting the assignment functions to only functions of the following form:*

$$\forall E_i \in E : q(E_i) \in X \text{ or } \neg q(E_i) \in X \tag{5.4}$$

we have a simpler yet completely equivalent definition of extended uncertain graphs.[1]

Proof. Given an extended uncertain graph $\mathcal{G} = (V, E, X, p, q)$, the Boolean expression $q(E_i)$ for any $E_i \in E$ can always be rewritten into an equivalent expression in conjunctive normal form (i.e., AND of ORs):

$$p(E_i) = \bigwedge_{i=1}^{k} \left(\bigvee_{j=1}^{n_i} Y_{i,j} \right) \tag{5.5}$$

where either $Y_{i,j} \in X$ or $\neg Y_{i,j} \in X$ for all i, j. In a similar fashion, edge E_i can be "expanded" into a subgraph in which each edge is associated with either a Bernoulli random variable in X or its negation. The expansion graph encodes the logical AND and OR operators of Equation 5.5 using series and parallel graph construction as shown in Figure 5.1. By replacing each and every edge in an extended uncertain graph with its expansion graph, we arrive at Condition 5.4.

□

[1] We require that X always contains the special random variable X^* where $Pr(X^* = 1) = 1$. This degenerate random variable is used for all certain edges introduced in the expansion graphs in Figure 5.1.

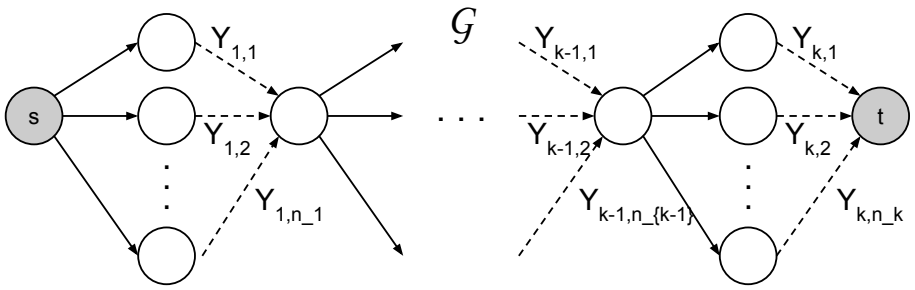

Figure 5.1: Expanding the Boolean expression in Equation 5.5 into an extended uncertain graph that satisfies Condition 5.4. For all i, j, either $Y_{i,j} \in X$ or $\neg Y_{i,j} \in X$.

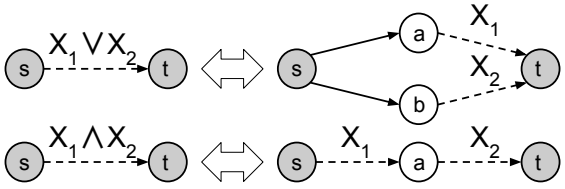

Figure 5.2: Two monotone uncertain graphs and their equivalent representations satisfying Condition 5.6.

Using Condition 5.4, what is possible now is that two distinct edges may be associated with the same Bernoulli random variable, or one edge to a random variable and the other to its negation. As a special case, a monotone uncertain graph can therefore be expanded into an uncertain graph in which the assignment functions have the specific form

$$\forall E_i \in E : q(E_i) \in X \tag{5.6}$$

For example, Figure 5.2 shows two 2-vertex monotone uncertain graphs and how they can be transformed into equivalent uncertain graphs that satisfy Condition 5.6. This observation allows us to prove the monotonicity of reachability of monotone uncertain graphs, which is stated below.

Theorem 5.2.1. *Let $\mathcal{G} = (V, E, X, p, q)$ and $\mathcal{G}' = (V, E, X, p', q)$ be two monotone uncertain graphs. Furthermore, let $p_i \geq p'_i$ for $i = 1, \ldots, r$. For all $s, t \in V$, the following inequality holds: $\mathcal{R}_{s,t}(\mathcal{G}) \geq \mathcal{R}_{s,t}(\mathcal{G}')$.*

Proof. Since $\mathcal{G}$ and $\mathcal{G}'$ are monotone uncertain graphs, without loss of generality, we can assume their assignment functions follow Condition 5.6. We first prove a special case of Theorem 5.2.1 in which $\mathcal{G}' = (V, E, X, p', q)$ where

$p' = (p'_1, p_2, \ldots, p_r)$. Define $E^1 \subseteq E$ the set of all edges associated with the random variable X_1 and assume $E^1 \neq \emptyset$ (otherwise, redefine $\mathcal{G}$ and $\mathcal{G}'$ without X_1). Furthermore, define two following uncertain graphs:

$$\mathcal{G}^0 = (V, E \backslash E^1, X, (p_2, p_3 \ldots, p_r), q)$$
$$\mathcal{G}^1 = (V, E, X, (1, p_2, \ldots, p_r), q)$$

Put simply, all possible worlds in $\mathcal{G}^1$ contain all edges in E^1 while none in $\mathcal{G}^0$ contains any. The reachability of $\mathcal{G}$ and $\mathcal{G}'$ with respect to any $s, t \in V$ can be computed by conditioning on the random variable X_1 as follows:

$$\mathcal{R}_{s,t}(\mathcal{G}) = p_1 \mathcal{R}_{s,t}(\mathcal{G}^1) + (1 - p_1)\mathcal{R}_{s,t}(\mathcal{G}^0)$$
$$\mathcal{R}_{s,t}(\mathcal{G}') = p'_1 \mathcal{R}_{s,t}(\mathcal{G}^1) + (1 - p'_1)\mathcal{R}_{s,t}(\mathcal{G}^0)$$

Hence:

$$\mathcal{R}_{s,t}(\mathcal{G}) - \mathcal{R}_{s,t}(\mathcal{G}') = (p_1 - p'_1)\left(\mathcal{R}_{s,t}(\mathcal{G}^1) - \mathcal{R}_{s,t}(\mathcal{G}^0)\right)$$

Since $p_1 \geq p'_1$, we only need to prove that $\mathcal{R}_{s,t}(\mathcal{G}^1) \geq \mathcal{R}_{s,t}(\mathcal{G}^0)$. For every possible world $G^1 \in \mathcal{G}^1$, the four following properties hold:

1. G^1 contains all edges in E^1,
2. G^0, as the result of removing all edges in E^1 from G^1, is a possible world in $\mathcal{G}^0$,
3. $w_{G^1,\mathcal{G}^1} = w_{G^0,\mathcal{G}^0}$, and lastly
4. $R_{s,t}(G^1) \geq R_{s,t}(G^0)$ according to the monotonicity of reachability of deterministic graph.

Consequently:

$$w_{G^1,\mathcal{G}^1} R_{s,t}(G^1) \geq w_{G^0,\mathcal{G}^0} R_{s,t}(G^0)$$
$$\sum_{G^1 \in \mathcal{G}^1} w_{G^1,\mathcal{G}^1} R_{s,t}(G^1) \geq \sum_{G^0 \in \mathcal{G}^0} w_{G^0,\mathcal{G}^0} R_{s,t}(G^0)$$
$$\mathcal{R}_{s,t}(\mathcal{G}^1) \geq \mathcal{R}_{s,t}(\mathcal{G}^0)$$

Therefore, $\mathcal{R}_{s,t}(\mathcal{G}) \geq \mathcal{R}_{s,t}(\mathcal{G}')$ for a specific case in which $\mathcal{G}' = (V, E, X, p', q)$ where $p' = (p'_1, p_2, \ldots, p_r)$.

Define $\mathcal{G}^{(i)} = (V, E, X, p^{(i)}, q)$ where $p^{(i)} = (p'_1, \ldots, p'_i, p_{i+1}, \ldots, p_r)$ for $i = 1, \ldots, r$. Note that $\mathcal{G}^{(0)} = \mathcal{G}$ and $\mathcal{G}^{(r)} = \mathcal{G}'$. By chaining the inequalities in the following fashion where each holds as a specific case, $\mathcal{R}_{s,t}(\mathcal{G}) = \mathcal{R}_{s,t}(\mathcal{G}^{(0)}) \geq \mathcal{R}_{s,t}(\mathcal{G}^{(1)}) \geq \ldots \geq \mathcal{R}_{s,t}(\mathcal{G}^{(r-1)}) \geq \mathcal{R}_{s,t}(\mathcal{G}^{(r)}) = \mathcal{R}_{s,t}(\mathcal{G}')$, the theorem is proven. □

The next result immediately follows Theorem 5.2.1:

Corollary 5.2.1.1. *Let $\mathcal{G} = (V, E, X, p, q)$ be a monotone uncertain graph, $\epsilon \in [0, 1]^r$ a perturbation vector such that $p_i - \epsilon_i \geq 0$ and $p_i + \epsilon_i \leq 1$ for $i = 1, \ldots, r$. We have: $p^{min} = p - \epsilon$ and $p^{max} = p + \epsilon$.*

As the main result of this section, Corollary 5.2.1.1 shows us how to perform uncertainty analysis of monotone uncertain graphs. The set of all monotone uncertain graphs contains all basic uncertain graphs but *strictly* subsumes the set of all extended uncertain graphs, as one might expect. If we take the extended uncertain graph in Figure 4.1 and change the Boolean expression associated with edge (a, t) from X_1 to $\neg X_1$, then we obtain a graph that does not have an equivalent monotone uncertain graph representation.

5.3 Uncertainty analysis of extended uncertain graphs

In the last section of this chapter, we show that uncertainty analysis of extended uncertain graphs in general is a hard problem by showing that any 3-SAT problem can be reduced to the problem of finding the maximum reachability of an extended uncertain graph where the probability assignment vector is defined within a certain hyperrectangle. Specifically, let $x = (x_1, x_2, \ldots, x_r)$ be the set of r Boolean variables and:

$$\mathcal{F} = \bigwedge_{i=1}^{n} (c_{i,1} \vee c_{i,2} \vee c_{i,3}) \tag{5.7}$$

is an n-clauses Boolean formula where $c_{i,j}$ are the literals, i.e. either $c_{i,j} \in x$ or $\neg c_{i,j} \in x$. Consider the following extended uncertain graph $\mathcal{G} = (V, E, X, p, q)$ constructed after $\mathcal{F}$:

- $V = \{V_1, V_2, \ldots, V_{n+1}\}$, $s \equiv V_1$ and $t \equiv V_{n+1}$,

- $E = \{E_1, E_2, \ldots, E_n\}$ where $E_i = (V_i, V_{i+1})$,
- $X = \{X_1, X_2, \ldots, X_r\}$,
- $p_i = Pr(X_i = 1) = 0.5$ for $i = 1, 2, \ldots, r$, and
- $q(E_i) = C_{i,1} \vee C_{i,2} \vee C_{i,3}$ which mimics the i-th clause of the Boolean formula $\mathcal{F}$ in the sense that if $c_{i,j} = x_k$ then $C_{i,j} = X_k$, otherwise if $c_{i,j} = \neg x_k$ then $C_{i,j} = \neg X_k$.

Finally, let us define the search space $\mathcal{H}_{p,\epsilon}$ where $\epsilon = 0.5$ – in other words, the search space is exactly the unit hypercube $\mathcal{H}_{p,\epsilon} \equiv [0,1]^r$. We then have the following result.

Corollary 5.3.0.1. *$\mathcal{F}$ is satisfiable if and only if* $\max_{p' \in [0,1]^r} \mathcal{R}_{s,t}(V, E, X, p', q) = 1$.

Proof. The "only if" part of the Corollary is straightforward given the construction. For the "if" part, note that if $\max_{p' \in [0,1]^r} \mathcal{R}_{s,t}(V, E, X, p', q) = 1$ then $p^* = \operatorname{argmax}_{p' \in [0,1]^r} \mathcal{R}_{s,t}(V, E, X, p', q)$ is a binary vector, i.e. either $p_i^* = 0$ or $p_i^* = 1$ for all i. This comes from the fact that $\mathcal{R}_{s,t}(V, E, X, p', q)$ is a linear function of p_i' for all i. Given the binary probability assignment vector p^*, we can easily find the corresponding truth assignment to x_i's that satisfies $\mathcal{F}$. □

We have just shown that a 3-SAT problem can be reduced to the uncertainty analysis problem of extended uncertain graphs. As a result, uncertainty analysis of extended uncertain graphs is NP-complete. Not surprisingly, this is usually the price we have to pay for extending the expressiveness of a modeling formalism. However, since the NOT logic operator is not required in the modeling examples in Chapters 6 and 7, uncertainty analysis can be performed efficiently in both cases.

Remark 2. Incorporating uncertainty into the model input is one step toward producing more trustworthy analyses. However, a large amount of uncertainty in the model input will likely produce a large amount of uncertainty in the model output. Although uncertainty analysis helps us quantify this relation, it does not tell exactly what part of the input's uncertainty contributes the most to the output's. This information is crucial to a modeler who desires to draw a more robust conclusion about the system and who

wants to know the best places to spend on reducing uncertainty (by collecting more information, adding more details into the model, etc.) When this is the case, a different but closely related form of analysis called sensitivity analysis should be considered.

CHAPTER 6

CASE STUDY: STUXNET PARTIAL ATTACK GRAPH

In the first modeling example, we show how to use an uncertain graph to model a partial attack graph of the Stuxnet worm (Figure 6.1), the infamous cyberweapon that sabotaged the Iranian nuclear program in 2009.

6.1 Modeling approach

On a high level, Stuxnet can be abstracted using logical steps that allow the worm to get access to the control system network of an air-gapped nuclear facility (Figure 6.1). Converting the Stuxnet partial attack graph (denoted as G_{Stux}) to an uncertain graph (denoted as $\mathcal{G}_{Stux}$) is relatively straightforward. $\mathcal{G}_{Stux}$ uses the same set of vertices as in G_{Stux}. Each random variable of $\mathcal{G}_{Stux}$ represents a unique edge label of G_{Stux}, so we will have a random variable for "Targeted Email With Dropper", another for "Infected USB Drive", and so on. Multiple edges of G_{Stux} that share the same starting and ending vertex are merged into a single edge of $\mathcal{G}_{Stux}$. Each edge of $\mathcal{G}_{Stux}$ is associated with a disjunction of random variables where each variable represents an edge label of G_{Stux}. For example if $E_i = (\text{con}, \text{lap})$ (Table 6.1) then $q(E_i) = X_{S7} \vee X_{USB}$ (Table 6.2 and Table 6.3). The complete description of $\mathcal{G}_{Stux}$ is given below:

- V = { web, splc, his, eng, tra, ecn, oss, wrm, vic, wrk, edr, cas, plc, emp, win, lap, con} (please refer to Table 6.1 for descriptions).
- E = {(web, eng), (web, oss), (web, win), (splc, vic), (his, oss), (his, cas), (eng, plc), (eng, splc), (tra, emp), (ecn, his), (ecn, cas), (oss, plc), (wrm, con), (wrm, emp), (wrm, tra), (wrk, cas), (wrk, eng), (wrk, ecn), (edr, ecn), (cas, web), (plc, vic), (emp, wrk), (emp, wrk), (win, oss), (win, eng), (lap, ecn), (con, emp), (con, lap), (con, cas), (con, eng), (con, edr)}.

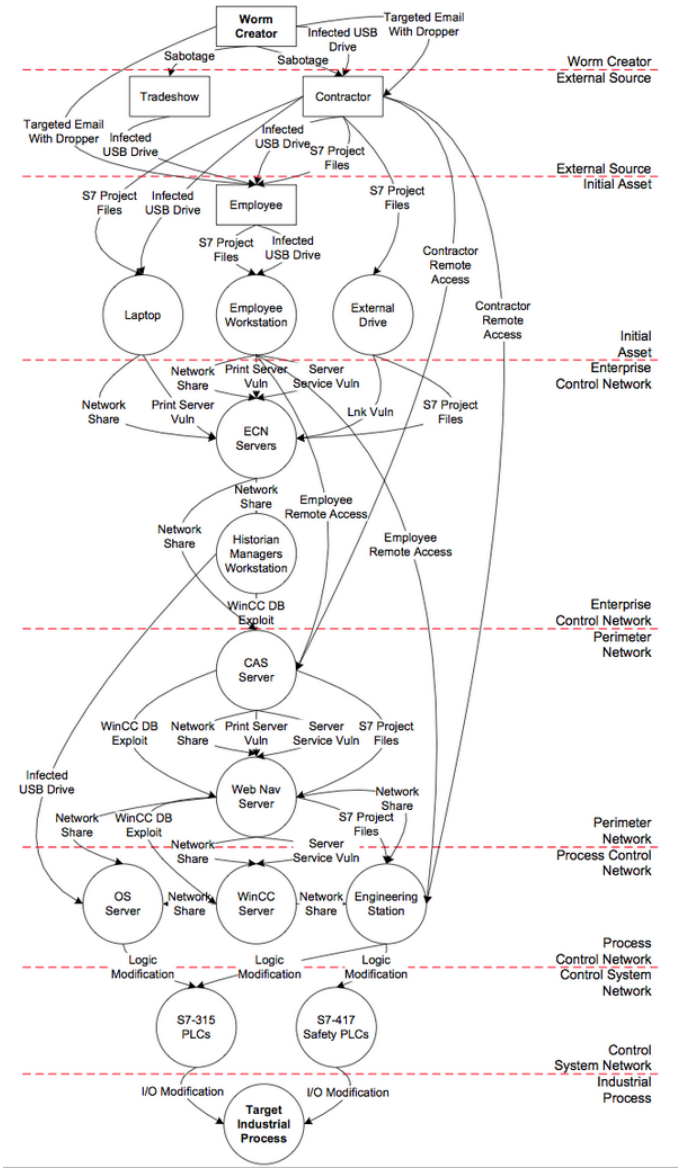

Figure 6.1: Stuxnet partial attack graph (figure reprinted from [30])

- $X = \{X_{USB}, X_{IOMOD}, X_{CRA}, X_{ERA}, X_{S7}, X_{SHR}, X_{EMAIL}, X_{WINCC}, X_{LMOD}, X_{PRINT}, X_{SERV}, X_{SAB}, X_{LNK}\}$ (please refer to Table 6.2 for descriptions).
- $p = (p_0, p_1, \ldots, p_{12})$ where $p_0 = Pr(X_{USB} = 1)$, $p_1 = Pr(X_{IOMOD} = 1)$, and so on.
- q is given in Table 6.3.

The remaining task is to come up with numerical values for the probability assignment vector of $\mathcal{G}_{Stux}$. Those numbers, which may include both the

means and their bounds, can be obtained after performing a full security auditing of the system.

Table 6.1: Vertices of $\mathcal{G}_{Stux}$.

Index	**Vertex**	**Description**
0	web	Web Nav Server
1	splc	S7-417 Safety PLCs
2	his	Historian Managers Workstation
3	eng	Engineering Station
4	tra	Tradeshow
5	ecn	ECN Servers
6	oss	OS Server
7	wrm	Worm Creator
8	vic	Target Industrial Process
9	wrk	Employee Workstation
10	edr	External Drive
11	cas	CAS Server
12	plc	S7-315 PLCs
13	emp	Employee
14	win	WinCC Server
15	lap	Laptop
16	con	Contractor

Table 6.2: Random variables used in $\mathcal{G}_{Stux}$.

Index	**Random variable**	**Description**
0	X_{USB}	Infected USB Drive
1	X_{IOMOD}	IO Modification
2	X_{CRA}	Contractor Remote Access
3	X_{ERA}	Employee Remote Access
4	X_{S7}	S7 Project Files
5	X_{SHR}	Network Share
6	X_{EMAIL}	Targeted Email With Dropper
7	X_{WINCC}	WinCC DB Exploit
8	X_{LMOD}	Logic Modification
9	X_{PRINT}	Print Server Vulnerability
10	X_{SERV}	Server Service Vulnerability
11	X_{SAB}	Sabotage
12	X_{LNK}	Lnk Vulnerability

Define s as the wrm vertex and t the vic vertex, using Algorithm 1 for symbolic computation, the reachability of the monotone uncertain graph $\mathcal{G}_{Stux}$

Table 6.3: Assignment functions used in $\mathcal{G}_{Stux}$.

Index	Edge	Assignment function
0	(web, eng)	$X_{S7}\ X_{SHR}$
1	(web, oss)	X_{SHR}
2	(web, win)	$X_{SHR} \vee X_{WINCC} \vee X_{SERV}$
3	(splc, vic)	X_{IOMOD}
4	(his, oss)	X_{USB}
5	(his, cas)	X_{WINCC}
6	(eng, plc)	X_{LMOD}
7	(eng, splc)	X_{LMOD}
8	(tra, emp)	X_{USB}
9	(ecn, his)	X_{SHR}
10	(ecn, cas)	X_{SHR}
11	(oss, plc)	X_{LMOD}
12	(wrm, con)	$X_{USB} \vee X_{EMAIL} \vee X_{SAB}$
13	(wrm, emp)	X_{EMAIL}
14	(wrm, tra)	X_{SAB}
15	(wrk, cas)	X_{ERA}
16	(wrk, eng)	X_{ERA}
17	(wrk, ecn)	$X_{SHR} \vee X_{PRINT} \vee X_{SERV}$
18	(edr, ecn)	$X_{S7} \vee X_{LNK}$
19	(cas, web)	$X_{S7} v X_{SHR} \vee X_{WINCC} \vee X_{PRINT} \vee X_{SERV}$
20	(plc, vic)	X_{IOMOD}
21	(emp, wrk)	X_{USB}
22	(emp, wrk)	X_{S7}
23	(win, oss)	X_{SHR}
24	(win, eng)	X_{SHR}
25	(lap, ecn)	$X_{SHR} \vee X_{PRINT}$
26	(con, emp)	$X_{USB} \vee X_{S7}$
27	(con, lap)	$X_{USB} \vee X_{S7}$
28	(con, cas)	X_{CRA}
29	(con, eng)	X_{CRA}
30	(con, edr)	X_{S7}

can be computed as (after simplification) $\mathcal{R}_{s,t}(\mathcal{G}_{Stux}) = p_1p_8(p_0p_{11}p_2p_3p_4p_5p_6 - p_0p_{11}p_2p_3p_4p_5 - p_0p_{11}p_2p_3p_4p_6 + p_0p_{11}p_2p_3p_4 - p_0p_{11}p_2p_4p_5p_6 + p_0p_{11}p_2p_4p_5 + p_0p_{11}p_2p_6 - p_0p_{11}p_2 - p_0p_{11}p_3p_4p_5p_6 + p_0p_{11}p_3p_4p_5 + p_0p_{11}p_3p_4p_6 - p_0p_{11}p_3p_4 + p_0p_{11}p_4p_5p_6 - p_0p_{11}p_4p_5 - p_0p_2p_3p_4p_5p_6 + p_0p_2p_3p_4p_6 + p_0p_2p_3p_5 - p_0p_2p_3 + p_0p_2p_4p_5p_6 - p_0p_2p_5 - p_0p_2p_6 + p_0p_2 + p_0p_3p_4p_5p_6 - p_0p_3p_4p_6 - p_0p_3p_5 + p_0p_3 - p_0p_4p_5p_6 + p_0p_5 - p_{11}p_2p_3p_4p_5p_6 + p_{11}p_2p_3p_4p_5 + p_{11}p_2p_3p_4p_6 - p_{11}p_2p_3p_4 +$

$p_{11}p_2p_4p_5p_6 - p_{11}p_2p_4p_5 - p_{11}p_2p_6 + p_{11}p_2 + p_{11}p_3p_4p_5p_6 - p_{11}p_3p_4p_5 - p_{11}p_3p_4p_6 + p_{11}p_3p_4 - p_{11}p_4p_5p_6 + p_{11}p_4p_5 + p_2p_3p_4p_5p_6 - p_2p_3p_4p_6 - p_2p_4p_5p_6 + p_2p_6 - p_3p_4p_5p_6 + p_3p_4p_6 + p_4p_5p_6$). Even without knowing the values of the probability assignment vector, there are two immediate observations we can make here.

First, in the formula of $\mathcal{R}_{s,t}(\mathcal{G}_{Stux})$ there are four missing parameters. They are p_7, p_9, p_{10}, and p_{12}, which correspond to the random variables X_{WINCC}, X_{PRINT}, X_{SERV}, and X_{LNK}, respectively (Table 6.2). The reason why those four parameters are eliminated from the formula can be attributed to the specific structure and the assignment functions of the Stuxnet graph. To understand this, consider a much simpler example in Figure 6.2. In this uncertain graph $\mathcal{G}$, s reaches t via a when $X_1 = X_2 = 1$, but if $X_1 = 1$ then s can also get to t via b. Hence X_1 is the only deciding factor that dictates whether s can reach t or not; in other words, $\mathcal{R}_{s,t}(\mathcal{G}) = Pr(X_1 = 1)$.

Second, the reachability formula of $\mathcal{G}_{Stux}$ can be factored into a product of three terms. The first two are p_1 and p_8, which correspond to the random variables X_{IOMOD} and X_{LMOD}, respectively. This factorization can be understood by visually inspecting the Stuxnet graph. By setting either X_{LMOD} or X_{IOMOD} to 0, the attack graph is partitioned at the control system network layer. Therefore, all paths from the wrm vertex to the vic vertex are disabled. Hence, these two are among the deciding factors regarding whether wrm can reach vic or not. The third term of the product is a rather lengthy formula that cannot be factored any further – this means there is no other single factor that can singlehandedly disable all paths from wrm to vic like X_{LMOD} or X_{IOMOD}.

6.2 Security analysis

The resulting uncertain graph $\mathcal{G}_{Stux}$, its reachability formula $\mathcal{R}_{s,t}(\mathcal{G}_{Stux})$, and the uncertainty analysis in the previous chapter allow an analyst to answer the following security-related questions:

1. What is the probability that there exists a path from outside of the system to a targeted industrial process?

2. To what extent should I trust the computed reachability probability –

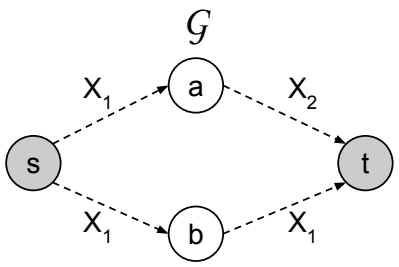

Figure 6.2: An extended uncertain graph with a redundant edge (a, t).

in other words, how sensitive is it to perturbation of model input?

3. If some form of network hardening is applied to the system and the probability assignment vector re-estimated, will reachability probability be reduced, and if so, by how much?

4. Instead of performing network hardening, I want to deploy an intrusion detection system (IDS) to detect ongoing attacks. Assume I choose to monitor a specific set of hosts, what is the probability that I miss an attack?

5. What should I do if the outcome of the analysis is not precise enough to draw any conclusion?

Questions 1 and 3 ask about the reachability of the uncertain graph which is estimated by means of sampling as shown in Section 3.3. Since the size of $\mathcal{R}_{s,t}(\mathcal{G}_{Stux})$ is relatively small, reachability can be directly computed using Equation 3.2 and Algorithm 1. Uncertainty analysis in Chapter 5 answers Question 2 since $\mathcal{G}_{Stux}$ is a monotone uncertain graph. Question 4 can be rephrased into the problem of estimating reachability of uncertain graphs; i.e., if I remove the set of vertices that correspond to the set of monitored hosts (together with all edges that connect to and from those vertices), what is the probability that t remains reachable from s? Question 5 is likely to arise in practice and usually indicates that the given amount of information is not sufficient to reason about the security posture of the system (refer to Remark 2 at the end of Chapter 5).

To complete the chapter, we include some numerical result in Figure 6.3. When p_i is set to 0.2 for all i, the reachability of the Stuxnet uncertain graph is $\mathcal{R}_{s,t}(\mathcal{G}_{Stux}) = 0.00687$ as shown in the solid horizontal line. To study the effect of p_i's on the overall reachability, we pick one parameter at a time,

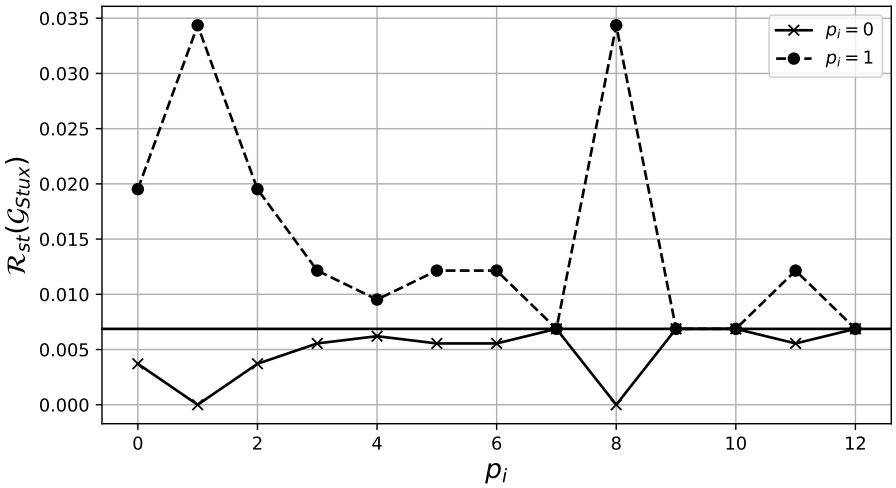

Figure 6.3: Reachability of the Stuxnet uncertain graph under different probability assignment vectors.

set it to 0 (dashed line marked with black circle) or to 1 (solid line marked with x) and recompute the reachability. It is obvious that p_1 (X_{IOMOD}) and p_8 (X_{LMOD}) contribute the most to the change in reachability, which was observed from the formula earlier on. The figure also shows us some other information that was not obvious from before, e.g. after p_1 and p_8, p_0 (X_{USB}) and p_2 (X_{CRA}) are the next important parameters to consider.

CHAPTER 7

CASE STUDY: NETWORK SECURITY WITH SERVICE UNCERTAINTY

In the second modeling example, we show how to use uncertain graphs to model a computer network with incomplete information about the network services, or service uncertainty. We first introduce the studied network and some basic networking concepts (Section 7.1). Then we define the threat model (Section 7.2) and propose an approach to model service uncertainty using uncertain graph (Section 7.3). We conclude the section with a note on how the probability assignment vector can be estimated using available information obtained from the common vulnerability databases.

7.1 Network model

Figure 7.1 shows a simple enterprise network consisting of 3 firewalls and 8 hosts. The firewall rules regulate the communication traffic in the network and define which hosts can directly talk to the other. For example, the 5-tuple rule `<6,0,1,*,80>` of firewall 1 allows all TCP traffic (protocol type 6) from any port on host 0 to port 80 on host 1. The deny-by-default policy is applied to all firewalls. As a result, firewall 1 blocks all TCP traffic from any port on host 0 to port 25 on host 1. The given enterprise network and the firewall rulesets effectively define a flow graph of logically connected hosts (Figure 7.2). The flow graph is a directed graph where each vertex represents a host in the enterprise network and each directed edge a flow, i.e. a logical connection. For example, the directed edge from vertex 0 to vertex 1 with the label `<80>` in Figure 7.2 represents a 3-tuple flow `<6:0-65535:80-80>` (i.e. the protocol, the source and destination port). There can be more than one flow from one host to another and in that case, the flow graph is a directed multigraph.

The flow graph is a general description of the types of traffic allowed be-

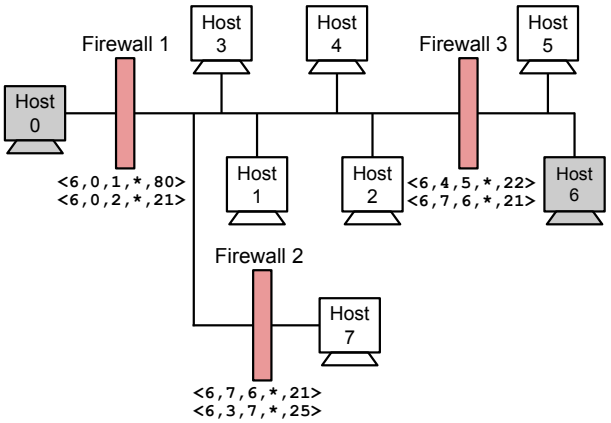

Figure 7.1: An enterprise network with 3 firewalls and 8 hosts (example adopted from [31], slightly modified for illustration purposes).

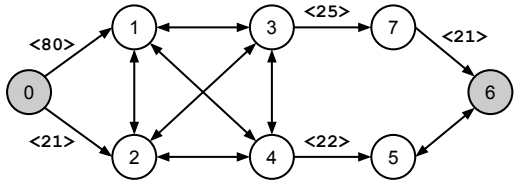

Figure 7.2: Flow graph representation of the enterprise network in Figure 7.1. Label `<80>` on flow from vertex 0 to vertex 1 is short for `<6:0-65535:80-80>`. Flows without label allow any traffic.

tween hosts in the network. Knowing that flow `<6:0-65535:80-80>` from host 0 to host 1 exists, we can make an educated guess that host 1 runs some form of an http service. For the purpose of security modeling and analysis, we are also interested in knowing the version and configuration details of the service. Without such information, the existence of a flow does not necessarily imply that an attacker can utilize it as a part of his lateral movement. In fact, the flow might exist while its corresponding service is not running at all. Security modeling and analysis under unquantified input uncertainty will not produce any significant result since any outcome is equally likely. However, if we are allowed to make further assumptions, which are reasonable ones, then the service uncertainty in flow graphs can be greatly reduced and reasonably estimated using augmented information from the public domain.

7.2 Threat model

We assume the attacker has already gained access to host 0. His ultimate goal is to gain access to host 6, which we know as a critical asset in the system. To simplify the discussion, we make some further assumptions:

- The attacker only exploits vulnerability of network services running on the receiving host of flows. As a result, if no flow from host 0 to host 1 is allowed or host 1 does not run any vulnerable service, then the attacker cannot launch a direct attack from host 0 to host 1.
- The flow graph remains unchanged throughout the attack period, meaning the attacker does not attempt to attack the firewalls and modify the rulesets to enable new flows.
- Local attacks are not modeled; we assume the attacker is capable of performing privilege escalation to acquire the highest access level after getting access to a machine.

7.3 Modeling approach

Define $X_{1,80}$ and $X_{1,!80}$ as the random variables that denote whether host 1 runs a vulnerable service on port 80 and on some other port. The flow graph in Figure 7.2 indicates the correlation between exploitability of flows in the following sense. If host 1 runs a vulnerable http service on port 80, or $X_{1,80} = 1$, then an attacker on either host 0 or 2 can use the existing flows to attack host 1. In contrast, if host 1 does not run any vulnerable http service on port 80, or $X_{1,80} = 0$, the attacker cannot attack host 1 from host 0. However, he might be able to do so from host 2, given that host 1 runs a vulnerable service on some other port, i.e. $X_{1,!80} = 1$. If we convert the flow graph to an extended uncertain graph with the same set of vertices and edges, then such a property can be modeled by associating edge $(0, 1)$ with $X_{1,80}$ and edge $(2, 1)$ with $X_{1,80} \vee X_{1,!80}$. Repeating this process to other edges and vertices, we can build an extended uncertain graph that models the service uncertainty and the correlation between edge existence of the flow graph in Figure 7.2. In this modeling example, by using simple graph transformation tricks we can further reduce such an extended uncertain

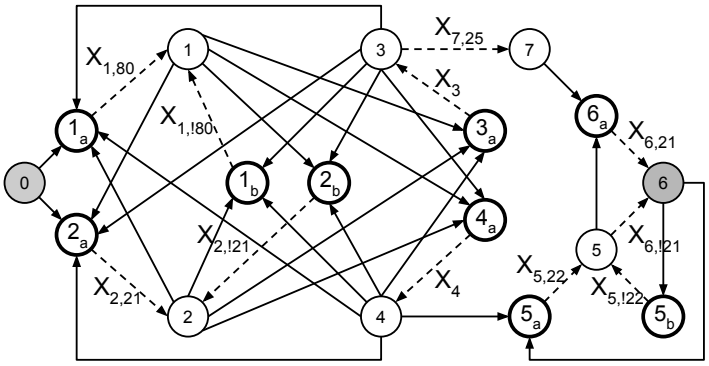

Figure 7.3: Basic uncertain graph representation of the flow graph in Figure 7.2.

graph to an equivalent basic uncertain graph $\mathcal{G}_{Serv}$ as shown in Figure 7.3 and tabulated in Table 7.1. The transformation introduces artifacts like certain edges, i.e. edges that exist with probability one (solid arrows), and extra vertices (bold circles). Defining s as vertex 0 and t vertex 6, and using Algorithm 1 for symbolic computation, the reachability of the basic uncertain graph $\mathcal{G}_{Serv}$ can be computed as (after simplification):

$$\begin{aligned}\mathcal{R}_{s,t}(\mathcal{G}_{Serv}) = (p_2 + p_4 - p_2p_4)(p_0p_6p_{10} + p_1p_7p_8 \\ + p_7p_8p_{10} - p_1p_7p_8p_{10} - p_0p_6p_7p_8p_{10})\end{aligned} \tag{7.1}$$

As before, we notice that the reachability formula can be nicely factored into a product form. To double check the result, we can examine the first term of the formula $(p_2 + p_4 - p_2p_4)$, which is equal to 0 if both $p_2 = 0$ and $p_4 = 4$. This translates to removing both uncertain edges $(1_a, 1)$ and $(2_a, 2)$ from $\mathcal{G}_{Serv}$, which in turn disables every path from vertex 0 to vertex 6 as seen from Figure 7.3, causing $\mathcal{R}_{s,t}(\mathcal{G}_{Serv}) = 0$. We also notice a few parameters missing from the formula. They are p_3, p_5, and p_9, which correspond to random variables $X_{1,!80}$, $X_{2,!21}$, and $X_{5,!22}$, respectively.

In the last part the section, we briefly discuss how to estimate the probability assignment vector for the constructed uncertain graph. The security analyst may assume (or he may learn from the system administrator) that without exception, all network services run on standard network ports, i.e. http services on port 80, ftp services on port 21, smtp services on port 25, and so on. The problem of service uncertainty still persists but the uncertainty

Table 7.1: Uncertain edges and associated random variables in $\mathcal{G}_{Serv}$.

Index	Edge	Random variable	Probability
0	(3, 7)	$X_{7,25}$	p_0
1	(5, 6)	$X_{6,!21}$	p_1
2	(1_a, 1)	$X_{1,80}$	p_2
3	(1_b, 1)	$X_{1,!80}$	p_3
4	(2_a, 2)	$X_{2,21}$	p_4
5	(2_b, 2)	$X_{2,!21}$	p_5
6	(3_a, 3)	X_3	p_6
7	(4_a, 4)	X_7	p_7
8	(5_a, 5)	$X_{5,22}$	p_8
9	(5_b, 5)	$X_{5,!22}$	p_9
10	(6_a, 6)	$X_{6,21}$	p_{10}

is now greatly reduced, since the analyst can infer that in order to make a direct lateral movement from host 0 to host 1, host 1 must run a vulnerable http service. The probability that host 1 runs a vulnerable http service, i.e. $Pr(X_{1,80} = 1)$, can be estimated as follows. For each http implementation h, the analyst searches in the common vulnerability databases (e.g. the National Vulnerability Database[1]) to see if there exists any vulnerability of h that can be exploited to compromise the hosting machine. Denote $v(h) = 1$ if the analyst finds at least one vulnerability and $v(h) = 0$ otherwise. The probability assigned to $X_{1,80}$ can be estimated as:

$$Pr(X_{1,80} = 1) = \frac{\sum_h w_h v(h)}{\sum_h w_h} \tag{7.2}$$

where w_h is the relative weight assigned to implementation h. If no further information is given, all implementations carry the same weight. This approach allows her to compute the most part of the probability assignment vector used in Equation 7.1 except for $p_1 = Pr(X_{6,!21} = 1)$. To estimate p_1, the analyst needs to find out all other services running on host 6 and use Equation 7.2 to compute the aggregated probability. If no further information is given, the analyst may assume host 6 has some default probability p^{def} of running a vulnerable network service. The probability assignment

[1]https://nvd.nist.gov/

$p_{10} = Pr(X_{6,21} = 1)$, $p_1 = Pr(X_{6,!21} = 1)$ and p^{def} are related according to:

$$p^{def} = p_1 + p_{10} - p_1 p_{10} \tag{7.3}$$

Therefore, p_1 can be computed as:

$$p_1 = \frac{p^{def} - p_{10}}{1 - p_{10}} \tag{7.4}$$

Numerical results of the analyses are not reported in this thesis and will be a significant topic in follow-up work, in which we study larger and more realistic systems.

CHAPTER 8

CONCLUSION

In this thesis, we show how to use uncertain graphs for the security modeling and analysis of computer systems with uncertainty. In doing so, we have extended the traditional uncertain graph formalism to model the correlation between edge existence and prove theoretical results about the expressiveness of basic and extended uncertain graphs. We also show how to perform uncertainty analysis of monotone uncertain graphs. Modeling-wise, the developed examples serve as a starting point for taking on larger and more complex systems. In such systems, uncertainty arises from modeling at different layers of abstraction and from the presence of humans-in-the-loop. Regarding humans, uncertain graphs can use existing human-related models to plug holes in the overall attack graph and model the probability that a phishing campaign succeeds or the probability that a power grid operator plugs in the USB stick he received at the conference. Analysis-wise, we are also interested in formulating and solving optimization problems to find the best defense actions, which minimizes the probability of a successful attack, given a limited budget. These aspects will be explored in subsequent studies.

REFERENCES

[1] J. P. McDermott, "Attack net penetration testing," in *Proceedings of the 2000 Workshop on New Security Paradigms*. ACM, 2001, pp. 15–21.

[2] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *Journal of Information Security and Applications*, 2016.

[3] R. P. Lippmann and K. W. Ingols, "An annotated review of past papers on attack graphs," DTIC Document, Tech. Rep., 2005.

[4] X. Ou and A. Singhal, *Quantitative Security Risk Assessment of Enterprise Networks*, ser. SpringerBriefs in Computer Science. Springer-Verlag New York, 2012.

[5] S. Asthana, O. D. King, F. D. Gibbons, and F. P. Roth, "Predicting protein complex membership using probabilistic network reliability," *Genome Res.*, 2004.

[6] N. Vesdapunt, K. Bellare, and N. Dalvi, "Crowdsourcing Algorithms for Entity Resolution," *Proc. VLDB Endow.*, vol. 7, no. 12, pp. 1071–1082, Aug. 2014. [Online]. Available: http://dx.doi.org/10.14778/2732977.2732982

[7] J. Ghosh, H. Q. Ngo, S. Yoon, and C. Qiao, "On a Routing Problem Within Probabilistic Graphs and its Application to Intermittently Connected Networks," in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, May 2007, pp. 1721–1729.

[8] M. Hua and J. Pei, "Probabilistic Path Queries in Road Networks: Traffic Uncertainty Aware Path Selection," in *Proceedings of the 13th International Conference on Extending Database Technology*, ser. EDBT '10. New York, NY, USA: ACM, 2010. [Online]. Available: http://doi.acm.org/10.1145/1739041.1739084 pp. 347–358.

[9] C. J. Colbourn, *The Combinatorics of Network Reliability*. New York, NY, USA: Oxford University Press, Inc., 1987.

[10] R. Jin, L. Liu, B. Ding, and H. Wang, "Distance-constraint Reachability Computation in Uncertain Graphs," *Proc. VLDB Endow.*, vol. 4, no. 9, pp. 551–562, June 2011. [Online]. Available: http://dx.doi.org/10.14778/2002938.2002941

[11] A. Khan, F. Bonchi, A. Gionis, and F. Gullo, "Fast Reliability Search in Uncertain Graphs," in *Proceedings of the 17th International Conference on Extending Database Technology, EDBT 2014, Athens, Greece, March 24-28, 2014.*, 2014, pp. 535–546.

[12] R.-H. Li, J. X. Yu, R. Mao, and T. Jin, "Recursive Stratified Sampling: A New Framework for Query Evaluation on Uncertain Graphs," *IEEE Trans. on Knowl. and Data Eng.*, vol. 28, no. 2, pp. 468–482, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2015.2485212

[13] A. Blum, C. Burcht, and J. Langford, "On learning monotone Boolean functions," in *Proceedings 39th Annual Symposium on Foundations of Computer Science*, Nov 1998, pp. 408–415.

[14] F. Moskowitz, "The analysis of redundancy networks," *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, vol. 77, no. 5, pp. 627–632, Nov 1958.

[15] L. G. Valiant, "The Complexity of Enumeration and Reliability Problems," *SIAM Journal on Computing*, vol. 8, no. 3, pp. 410–421, 1979.

[16] M. Potamias, F. Bonchi, A. Gionis, and G. Kollios, "K-nearest Neighbors in Uncertain Graphs," *Proc. VLDB Endow.*, vol. 3, no. 1-2, Sep. 2010.

[17] O. M. Sheyner, "Scenario graphs and attack graphs," Ph.D. dissertation, Carnegie-Mellon University, 2004.

[18] B. Schneier, "Attack trees," *Dr. Dobb's Journal*, vol. 24, no. 12, pp. 21–29, 1999.

[19] R. W. Ritchey and P. Ammann, "Using Model Checking to Analyze Network Vulnerabilities," in *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, ser. SP '00. IEEE Computer Society, 2000.

[20] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing, "Automated generation and analysis of attack graphs," in *Security and privacy, 2002. Proceedings. 2002 IEEE Symposium on*, 2002.

[21] X. Ou, W. F. Boyer, and M. A. McQueen, "A scalable approach to attack graph generation," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*. ACM, 2006, pp. 336–345.

[22] P. Ammann, D. Wijesekera, and S. Kaushik, "Scalable, graph-based network vulnerability analysis," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM, 2002, pp. 217–224.

[23] L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia, "An Attack Graph-Based Probabilistic Security Metric," in *Proceeedings of the 22nd Annual IFIP WG 11.3 Working Conference on Data and Applications Security*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 283–296.

[24] P. Xie, J. H. Li, X. Ou, P. Liu, and R. Levy, "Using Bayesian networks for cyber security analysis," in *2010 IEEE/IFIP International Conference on Dependable Systems Networks (DSN)*, June 2010, pp. 211–220.

[25] G. F. Lyon, *Nmap network scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Insecure, 2009.

[26] OpenVAS Developers, "The Open Vulnerability Assessment System (OpenVAS)," 2012.

[27] C. Baier and J.-P. Katoen, *Principles of Model Checking (Representation and Mind Series)*. The MIT Press, 2008.

[28] G. S. Fishman, "A Comparison of Four Monte Carlo Methods for Estimating the Probability of s-t Connectedness," *IEEE Transactions on Reliability*, vol. 35, no. 2, pp. 145–155, June 1986.

[29] A. Saltelli, M. Ratto, S. Tarantola, and F. Campolongo, "Sensitivity analysis practices: Strategies for model-based inference," *Reliability Engineering & System Safety*, vol. 91, no. 10-11, pp. 1109–1125, 2006.

[30] E. Byres, "Stuxnet Report V: Security Culture Needs Work," http://www.isssource.com/stuxnet-report-v-security-culture-needs-work/, 2011.

[31] L. Wang, S. Jajodia, A. Singhal, P. Cheng, and S. Noel, "k-Zero Day Safety: A Network Security Metric for Measuring the Risk of Unknown Vulnerabilities," *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 1, pp. 30–44, Jan 2014.