Determining the Effectiveness of Nuclear Security Through Computer Simulation

by

Nicholas Jordan Chornoboy

A thesis submitted in partial fulfillment
of the requirements for the degree of

Masters of Applied Science

in

Nuclear Engineering

University of Ontario Institute of Technology

Supervisor: Dr. Ed Waller

Sept 2015

Abstract

There is a growing concern from both national regulators and the International Atomic Energy Agency (IAEA) about the threat posed by attacks against iconic targets such as nuclear power plants. This has led to an increased desire to be able to objectively measure the effectiveness of the physical security of these sites to prevent theft or sabotage of the nuclear and radiological material. Currently verification of physical protection systems is done using subjective expert opinion as well as time consuming and expensive live exercises. A method that allows experts to design and test a facility in the absence of live action exercises using larger sample sizes would be highly desirable. To fill the niche a synthetic environment model was designed around the force on force simulation program STAGE to allow the full 3-D simulation of a nuclear facility. This allows for simple user modifications to the model, allowing many scenarios to be tested. Many detectors were added to more accurately reflect the types of sensors present at a nuclear facility. Having modeled the facility and the probabilities associated with various events, Monte-Carlo methods were applied to obtain statistics on how effective the guard force was at stopping the adversarial force. This technique can be used to give experts more robust, simple to use tools for the design and verification of physical protection systems.

Keywords: Modeling, Simulation, Nuclear Security, STAGE, Monte-Carlo

Acknowledgements

First I would like to thank the University Networks of Excellence in Nuclear Engineering (UNENE) and the Natural Sciences and Engineering Research Council (NSERC) for their funding and support.

I would also like to thank the Presagis team not only for generously providing their software under an NSERC Engage grant but also for their continued technical support without which the modifications to the STAGE engine would likely would have taken significantly longer.

I'd like to thank my friends, family and colleagues who were my sounding boards, tech support, editors and commiserates all throughout my studies.

Thank you to Dr. Waller for giving me this opportunity and shepherding me along the way.

And finally thank you for taking the time to read this document, I hope it contains what you are looking for.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The validation of physical protection systems at nuclear facilities is a topic of increasing interest among both national and international regulatory bodies due to the growing threat posed by asymmetric attacks [1, 2]. These are attacks perpetrated by small groups of determined individuals with the goal of theft or sabotage of the radiological or special nuclear material present at these facilities for use in radioactive dispersal devices, improvised nuclear devices, or other terrorist goals. With the global increase in terrorist activities regulators and operators must validate that both current designs and new build physical protection systems can withstand the anticipated threat [3]. This anticipated threat is referred to as the design basis threat and is a theoretical attack based on the most conservative estimates of adversary strength and ability [4].

Nuclear security is similar to the defense of any other hard target such as a military installation or warehouse. These targets are secured using physical protection systems such as walls, a variety of deployable barriers and sensors in addition to the guard

force and their response procedures [3, 5]. Nuclear security differs from other hard target security due to the presence of nuclear material and the added regulations surrounding its security as well as the safety concerns it entails. Security systems are designed with the goal of having physical protection systems that are varied enough to detect and slow the intruders long enough for the response force to intercept and defeat them [5, 6]. If the defense force is not able to respond before the adversary completes their goals the physical protection systems are considered ineffective [5]. An example of a portion of the physical protection system at a nuclear power plant is shown in Figure 1.1.

Current methodologies for validation and verification of physical protection involve live force on force exercises, used in conjunction with expert opinion, to determine weaknesses and areas for improvement [3, 7]. These exercises are expensive and time consuming, meaning they are done infrequently [4, 8]. In the absence of live action exercises, experts have less information which leads to more uncertainty in determining the systems effectiveness [7]. It would be useful to have some way to test the effectiveness of the security of a given facility that can be done on an ongoing basis to supplement live action exercises. By providing a larger sample size, better information on the effectiveness of the physical protection systems can be obtained.

1.2 Motivation for Thesis

Current methods of designing and verifying nuclear security work well. However, being able to quantify how effective a design is and to be able to rapidly prototype and modify new designs would be an asset to designers. This is difficult with current methods as expert opinion and the simple models used do not have as much rigor as is desirable and, as mentioned in the previous section, live action exercises are

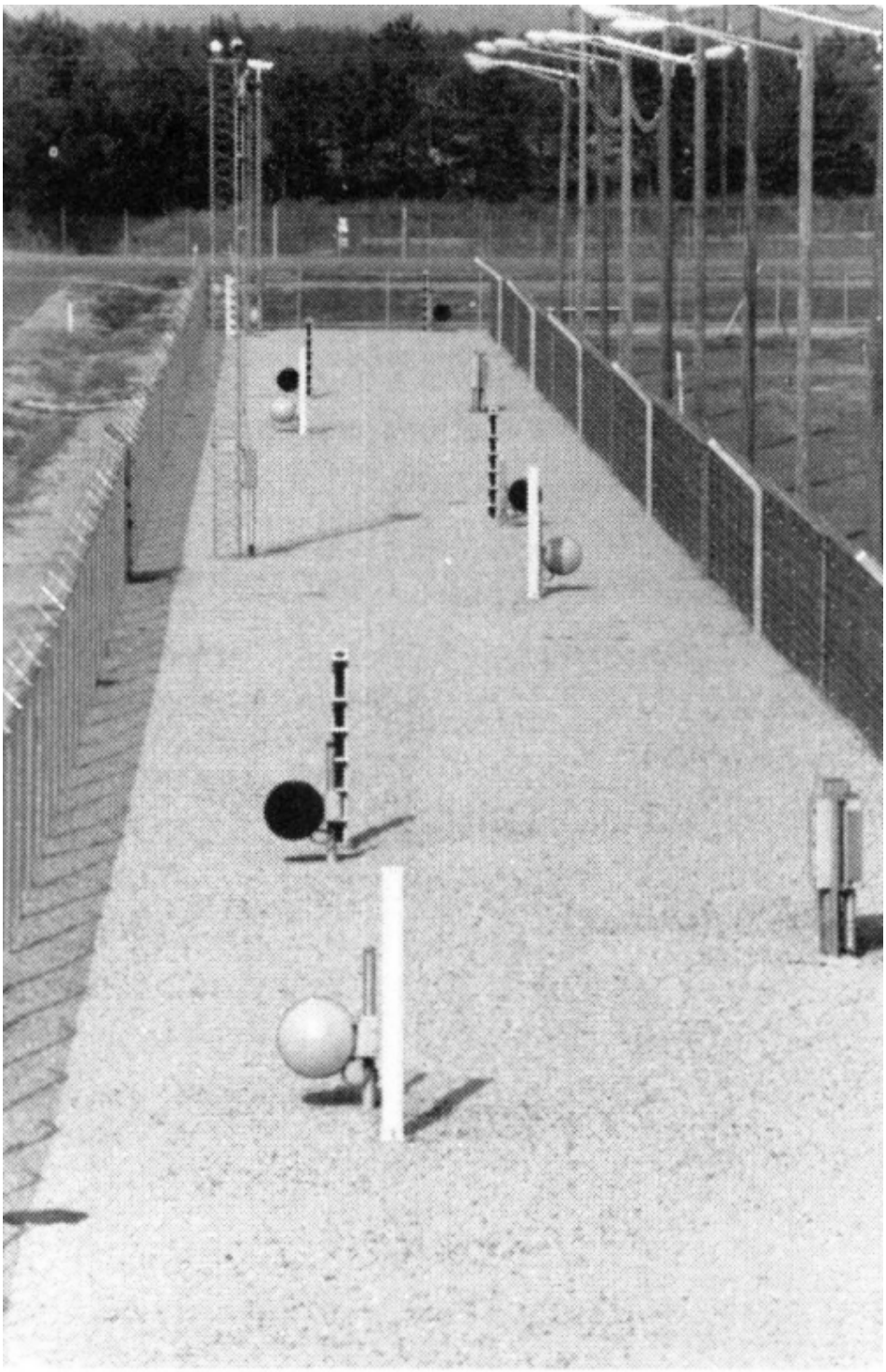

Figure 1.1: Double fence enclosure around a nuclear power plant with a variety of sensors [5].

expensive and infrequent [7]. The models used are abstract, making them difficult to apply for complex facilities and scenarios [5,7]. Additionally live action exercises are only possible on already constructed facilities. Therefore, it is desirable to have some sort of tool that allows many scenarios to be run in order to obtain concrete data on how effective the physical protection systems of the facility are before committing to live action exercises.

1.3 Objective of Thesis

The objective of this thesis is to develop a tool that assists industry experts in the design and testing of the security of nuclear facilities in a more rigorous way. This is achieved using computer simulation and synthetic environment modeling of the facility in order to run Monte-Carlo simulations of force on force encounters at the facility. To effectively do so the model must simulate the detection, delay, interception, and engagement of an adversary force. It must also be simple enough to use so that constructing new simulations within the model or making slight changes to the model requires minimal training and are quick to implement.

1.4 Organization of Thesis

This thesis begins in Chapter 1 with an introduction to nuclear security and some of the short comings of current techniques. This is followed in Chapter 2 with the theoretical background of nuclear security and how its effectiveness is measured. This is followed in Chapter 3 by a description of typical physical protection system features and how they function. These concepts provide the base line information required to have a good understand of the material being presented. Chapter 4 is an analysis of

the current security analysis tools available and how synthetic environment simulation can be used to improve the industry. Next in Chapter 5, an overview of the modeling engine used as a foundation for the work done will be covered. In Chapter 6 the modifications made to this engine and the creation process of scenarios will also be discussed. The results of the scenarios created will be presented in Chapter 7 followed by discussion on the impact and validity of the results and the model in Chapter 8. Finally in Chapter 9 some conclusions will be drawn and future work discussed.

Chapter 2

Literature Review - Theoretical Background

2.1 Design Basis threat

When designing a facility, the physical protection system is evaluated against the design basis threat to determine if it is sufficient to prevent successful attack against the facility with a high degree of certainty [4, 5]. The design basis threat is created using an in depth threat assessment based on the State's evaluation of the threat of a theorized attack on the facility using the most pessimistic assumption of adversary abilities and its consequences. This includes, but is not limited to the adversary's [4];

- Motivations
- Goals (theft, sabotage, embarrassment, etc.)
- Numbers
- Tactics

- Weapons
- Training
- Tools
- Transportation
- Explosives
- Level of access
- Insider knowledge

Also included in the design basis threat is the possibility of an insider threat varying from coerced assistance, to active participation. This can include ignoring alarms, opening doors and active participation in combat [4]. These are conditions that the physical protection system will be held accountable to reasonably defeat [4]. At most nuclear facilities the consequence of a successful malicious act is likely to be unacceptable and therefore all efforts must be taken to prevent this occurrence [2].

Not all threats are included in the design basis threat. Threats with low consequence may be discarded as well as those without a credible motive or intent [4, 9]. Steps must still be taken to prevent these, however they are not the focus of the physical protection system. Some threats such as those posed by state actors are also excluded from the design basis threat and remain the responsibility of the state to protect against [4]. Protection is planned beyond the design basis threat. There is some inherent protection, however after a point, protection against unacceptable consequences is no longer assured. [4]. More than one design basis threat may be developed for different facilities, different materials and different adversary objectives [2]. The relationship between the design basis threat and the threats laid out in the threat assessment can be seen in Figure 2.1.

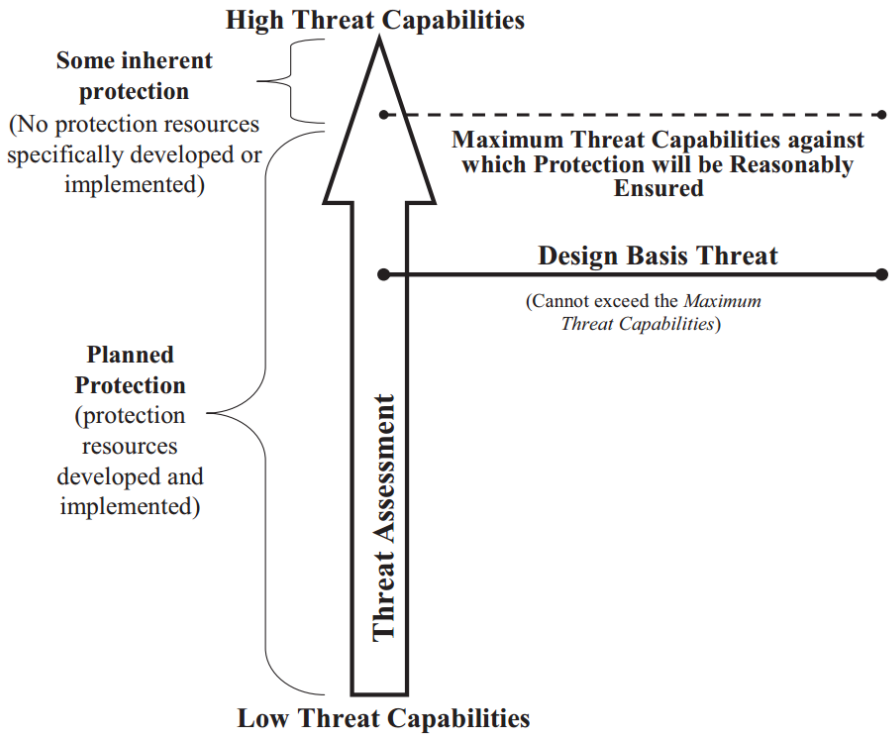

Figure 2.1: DBT relationship to various levels of threat [4].

It is important to note that the design basis threat is not a specific threat or named adversary but is the maximum threat against which protection against unacceptable consequence must be reasonably assured [2,4]. This may include capabilities derived from likely adversaries however, should be kept as general as possible to account for other possibilities. The design basis threat exists to provide clear conditions under which a physical protection system must be effective in a concise manner. Threats are inherently dynamic and as such the design basis threat represents a generic level of protection a facility must provide [9].

2.2 Regulatory Requirements

The security at nuclear sites both in Canada and other countries around the world are governed by various regulations put forth by their respective regulatory agencies.

In the case of Canada this is the CNSC (Canadian Nuclear Safety Commission) and for the United States this is the NRC (Nuclear Regulatory Commission). These regulations determine what is deemed sufficient for a physical protection system as well as some requirements for how this standard must be met [10].

Canadian regulations are governed by the Nuclear Safety and Control Act specifically Nuclear Security Regulations SOR /2000-209 [10]. This is further supported by a variety of guidance documentation, specifically for nuclear power facilities Security Programs for Category I or II Nuclear Material or Certain Nuclear Facilities G-274 [11]. Much of this documentation is available on a need to know basis, however those that are available give basic requirements of a physical protection system such as minimum protected area fence height [11]. No specific requirements are given for the effectiveness of the facility. This is because, as will be discussed latter, security is very difficult to quantify objectively. Assessment of the effectiveness of a facilities physical protection systems is done on a case by case basis with some assistance from simple tools and exercises informing the decision [10].

Where Canadian regulations are descriptive the United States regulations are more prescriptive. This means that much of the capabilities of the physical protection system a facility must have are laid out in the regulations. These are governed by USNRC regulatory documents 5.1 to 5.84. of particularly interest to physical protection systems are 5.52, 5.59, 5.76 [12–14]. These outline the specific requirements for a physical protection system. Similar to the Canadian regulations these are available on a need to know basis. These regulations outline of the desired effectiveness must be met rather than giving guidelines on what must be achieved. The effectiveness of a physical protection system is determined using a NRC monitored live action exercises every five years as outlined in USNRC RG 5.75 [15].

Because of how difficult quantifying of effectiveness the security of a nuclear facility

objectively can be a variety of methods exist that aid in doing so. Many of these methods exist to aid the designer in designing or improving the physical protection system with a small subset being used by the regulator to measure the effectiveness of the facility. It is not the goal of all measurement tools to meet the guidelines, many exist to aid in the design separate from the regulatory requirements.

2.3 Quantifying Security

2.3.1 Components

There are two main components to a physical protection system, detection mechanisms and delay mechanisms [5, 16]. A successful defense also requires a response force in some capacity to respond to the intrusion [3]. Detection mechanisms are often sensors such as cameras, infrared beams and fence vibration detectors however they can also include stationary and patrolling guards [5]. These serve to alert the operators that an adversary is present and trying to breach the facility. Delay mechanisms are usually physical barriers such as walls and doors but can also include large distances and more advanced systems such as immobilizing foam [5]. These delay mechanisms serve to slow the adversary down, giving the defense force enough time to react and deploy to and prevent the adversary from completing their malicious actions. Finally, the defense force consists of individuals tasked with responding to alarms and prevent, adversary task completion. The defense force can vary substantially from an on-site garrison with a large number of defenders to a handful of guards with batons to an off-site force that must first reach the facility before responding [8]. These work together to prevent unacceptable consequences to the facility.

2.3.2 Adversary Task time

The various components of the physical protection system must work together. If an adversary can enter the facility undetected, the delay mechanisms before they are detected do not contribute to the defense of the facility. [5, 8]. Physical protection systems are designed to delay the adversary long enough for the defense force to respond. If the adversary has not been detected the defense force can not begin to respond and those portions of the physical protection system cannot be included in the delay time [3, 17]. In Figure 2.2 a graphical representation of this scenario is shown. Should the overall physical protection system time exceed the adversary time after the point of detection, the adversary will complete their task before they can be intercepted [3]. Adversary tasks include events such as breaching a door or crossing an area. Each of these events have a detection probability and task time associated with them [5].

It is crucial that detection probabilities be as high as reasonably achievable towards the beginning of the security event to ensure adequate time for the defense force to respond [8]. This is important for the same reason that delay times of the physical protection system be higher closer to the adversary's target. The target being considered and the adversary's plan of action as well as how well the physical protection system defends against it is determined using the design basis threat and expert opinion supported by live action exercises and adversary sequence diagrams [8]. These adversary sequence diagrams are then also used to find the effectiveness of the physical protection system against this attack.

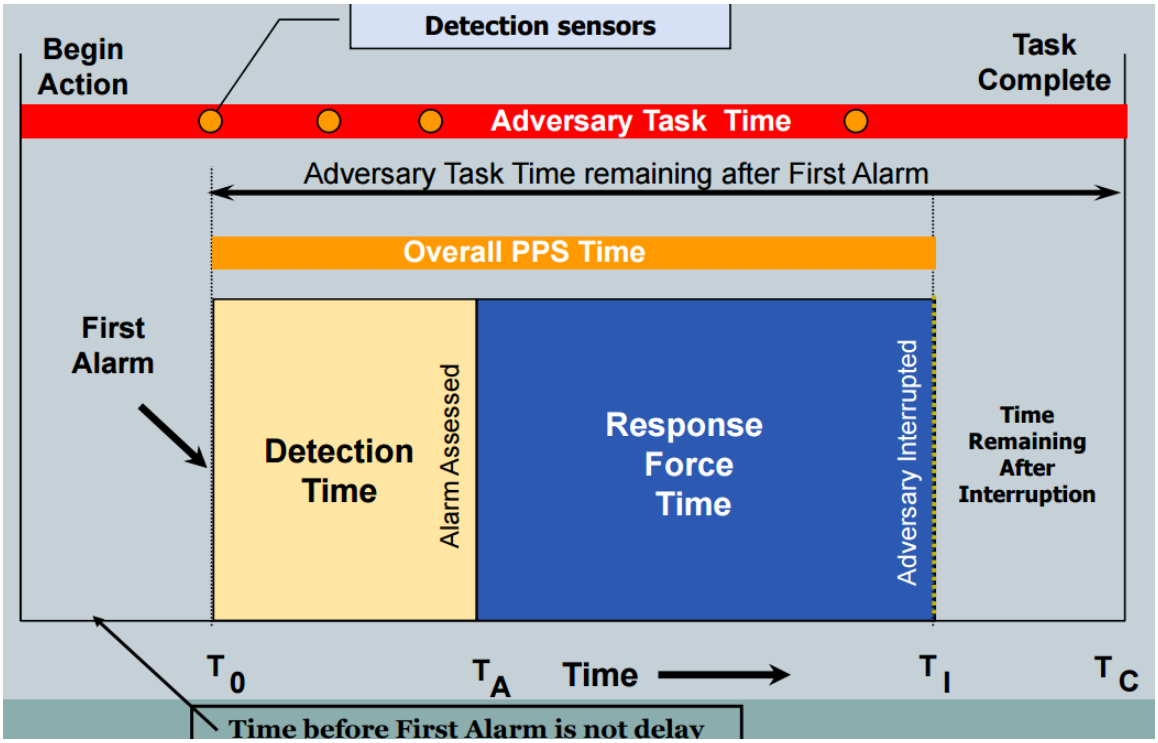

Figure 2.2: Relationship between advesary task time, response force time and detection [18].

2.3.3 Measuring Effectiveness

It is important to be able to quantify how effective a facility's physical protection system is. However, measuring the effectiveness of a physical protection system does not lend itself well to numerical representation. This is mainly due to the difficulty of performing measurements. Effectiveness is often given as the probability that the defense force successfully prevents the adversary from completing their intended malicious action for a theorized event [5,8]. Thankfully attacks on facilities are rare therefore this number must be estimated in some way [1]. Probability of effectiveness is often broken up into two parts to make it easier to estimate; these are probability of interruption and probability of neutralization [5,8]

Probability of interruption represents the likelihood that the adversary is detected and intercepted by the guard force [5,8]. This focuses on tools such as adversary sequence diagrams were experts use properties such as detection probability and delay time of the various components of the physical protection system to estimate how likely and after how long the adversary will be detected [5]. The equation for the probability of interruption is given as,

$$P_I = \left(1 - \prod_{i=1}^{n} (1 - P_{D_i})\right) * P_C, \tag{2.1}$$

where n is the number of possible detection events, P_{D_i} is the probability of detection for each individual event, and P_C is the probability of communication of a detection to the response force [5]. Information about the guard force is then used to determine if they will arrive before the adversary completes their intended task. For this reason detection events that occur when the adversary task time is shorter than the response force time are not counted. The probability of interruption does not have any information on whether or not the response force defeats the adversary [16,19].

Probability of neutralization represents how likely the response force is to defeat the adversary in combat [5, 8]. This is generally estimated using data from live action exercises [8]. These exercises involve teams of trained forces undergoing mock engagements with each side having differing numbers, equipment and tactics [19]. The results of these in the form

$$P_N = \frac{N_{wins}}{N_{engagements}}, \tag{2.2}$$

are then compiled into charts for use in [8]. An example comparing number of participants on each side can be seen in Table 2.1. These charts can then be used to estimate the probability of neutralization using

$$P_N = 1 - \left(1 - P'_{N_i,j}\right)^k, \tag{2.3}$$

where $P'_{N_i,j}$ is the value found on the previously mentioned charts for i response force members and j adversary members. k is the force multiplication coefficient which is used to approximate outcomes for engagements between unlike forces [8]. This will be discussed in the next section.

These two values can then be multiplied together to find the overall probability of effectiveness of the scenario. These results are found using point estimates that may not be completely representative of the scenario of interest that ignore the interactions between the various elements of the physical protection system. An example of this methods usage will be shown in conjunction with adversary sequence diagrams later on in this chapter.

Table 2.1: Probability of neutralization for differing number of participants [8].

		Number of Adversaries				
		1	2	3	4	5
Number	1	0.5	0.165	0.042	0.007	0.001
of	2	0.835	0.5	0.225	0.079	0.024
Defence	3	0.958	0.775	0.5	0.26	0.112
Force	4	0.993	0.921	0.74	0.5	0.285
	5	0.999	0.976	0.888	0.715	0.5

2.3.4 Approximations

The rule of two is a force multiplication coefficient used as an approximation in simple security calculations to determine the outcome of combat between forces with dissimilar armaments [8]. For example, this factor would be applied to probability of neutralization charts to adjust for one side being armed with assault rifles and the other with pistols. The rule states that every level of armament difference, such as pistols to assault rifles is worth twice as many participants as the previous level [8]. This gives k a value of two when the opponent is armed with the lesser equipment and 0.5 when the response force is. Similar approximations are used to account for other differences such as training, in order to modify the data gained from generic live action exercises that are often done using the same equipment and training [8]. An example of this relationship can be seen in Figure 2.3.

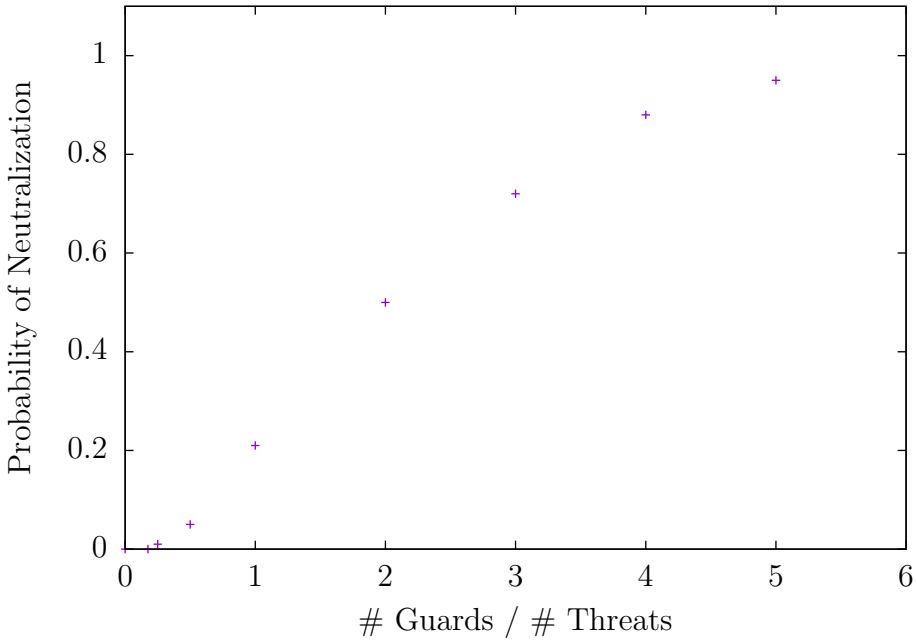

Figure 2.3: Probability of neutralization for hand guns vs assault rifles [8].

2.4 Effectiveness Measurement tools

2.4.1 Overview

The most accurate information about the effectiveness of a physical protection system of a facility comes from an attack on the facility however as mentioned these are thankfully rare. For this reason attacks on a facility must be simulated in some manor in order to estimate the effectiveness. A simulation is an imitation of a real world process using a model that is intended to replicate key behaviors and functions of the system it represents [8, 19]. The model used can vary from simple analytical equations to complex interacting compartmental models to real life exercises. These have varying accuracy and all have merit to there approaches but are all tied together by their attempt to approximate the real world system. Below various simulations used to estimate the effectiveness of a physical protection system are discussed.

2.4.2 Adversary Sequence Diagrams

One of the primary tools used by experts to determine how effective a facility is at defending against attack are adversary sequence diagrams [5,8,19,20]. An adversary sequence diagram is a collection of adversary activities and the associated delay time and detection probability of each. To properly construct an adversary sequence diagram an adversary path analysis must be performed [5]. To do this, each element of the physical protection system is given a value for delay and a probability of detection. These components are then used to predict all credible adversary paths into the facility to the target [5] [20]. This must be done for each threat and target laid out by the design basis threat as the adversary capabilities determine the delay and detection values of the physical protection system components [4]. For example, a concrete wall has a lower delay time but higher detection probability if the adversary uses explosives than if they use power tools [3]. Once all of the credible pathways have been determined, adversary sequence diagrams can be constructed, an example of one can be seen in figure 2.4. The larger boxes represent physical areas within the facility that the adversary must pass through. The smaller boxes between are the delay features that the adversary must defeat. These are used in adversary path analysis to attempt to determine the shortest path through the facility.

2.4.3 Interruption Analysis Charts

An adversary sequence diagram is used in conjunction with an adversary path analysis to identify the key physical protection system components of interest during an attack and use them to estimate the probability of interruption [20]. An example of an interruption analysis using the EASI (Estimate of Adversary Sequence Interruption), a code which solves the probability of interruption equation 2.1, can be

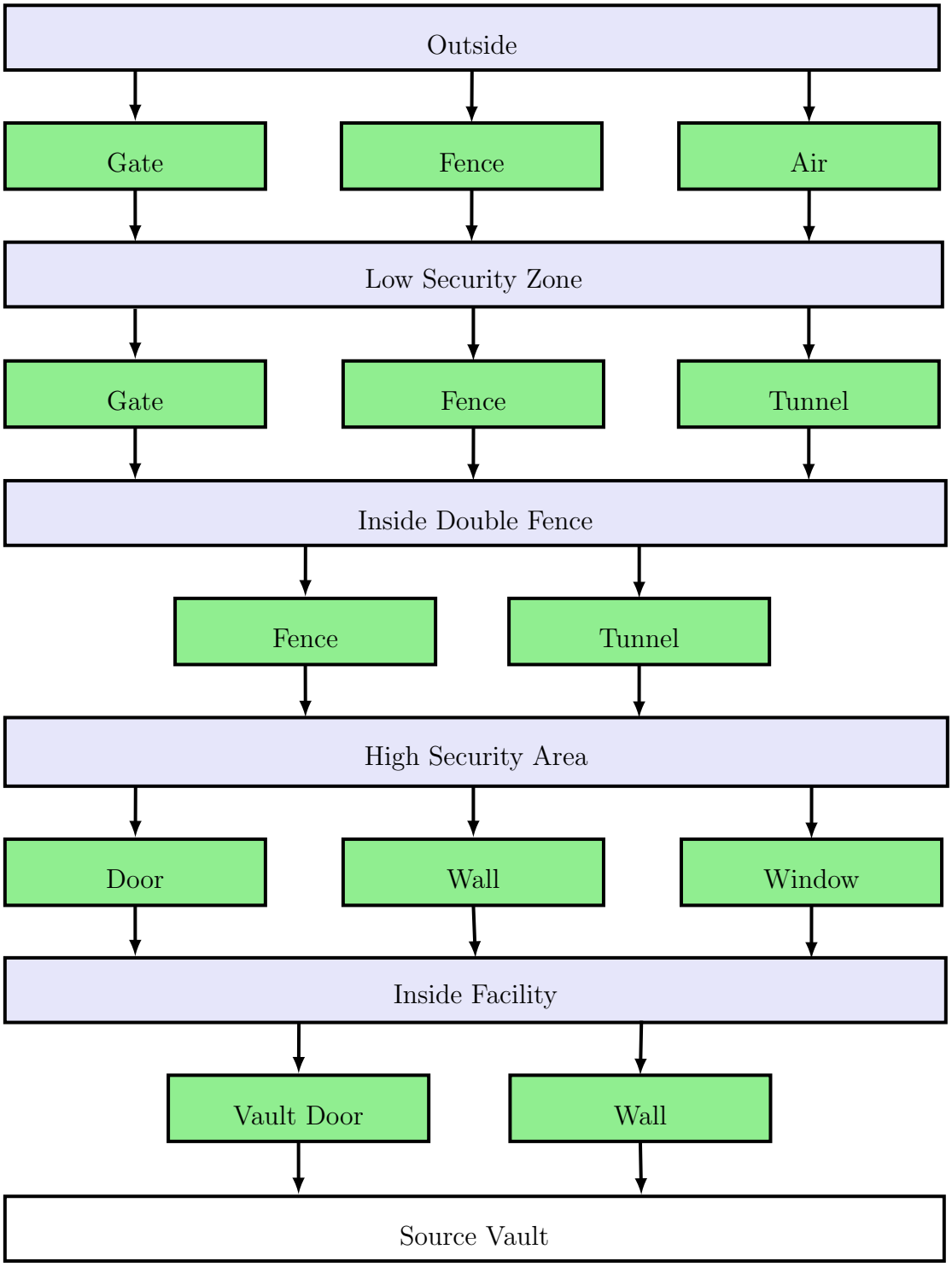

Figure 2.4: Adversary sequence diagram for example facility.

seen in Table 2.2. This example was generated using the adversary sequence diagram constructed in Figure 2.4. The properties of the guard force are shown in the top right. This includes the average response time with a standard deviation associated with it as well as the probability that the alarm gets communicated to them. This communication probability is important as no system is perfect and radio messages can get missed or communication technology malfunction [20]. Along the left hand side are adversary tasks. These are the components of the physical protection system that the adversaries must overcome to reach their target. Each of these components has an associated probability of detection and delay time that is the same as the ones found in the pathway analysis [8]. EASI also allows locations associated with each of these tasks to be specified, indicating when during the task the adversary would be detected if a detection occurs: at the beginning, middle or end [20]. These locations, along with being able to specify standard deviations allows EASI to give a more complete analysis.

Table 2.2: Adversary sequence diagram example using EASI [20].

	Estimate of Adversary Sequence Interruption	Probability of Guard Communication		Response Force Time(s)	
				Mean	STD
		0.95		120	36
				Delays (s)	
Task	Description	P(Detection)	Location	Mean	STD
1	Approach site	0.01	M	360	108
2	breach outer fence	0.15	B	30	9
3	approach inner fence	0.2	M	36	10.8
4	penetrate inner fence	0.8	B	90	27
5	move toward facility	0.3	M	180	54
6	enter facility	0.85	B	10	3
7	locate fuel	0.3	M	60	18
8	Enter Fuel Bay	0.75	B	10	3
9	sabatoge	1	M	0	0
	P (Interruption)	0.867			

To determine the probability of interruption all detection probabilities on tasks occurring within the response force time of the final event are ignored [20]. In the example in Table 2.2 this means every task after five is ignored for the purposes of detection. This is because even if the adversary is detected they could not be interrupted in time to prevent them from completing their malicious action. The time at which this boundary is crossed is refereed to as the critical detection point [3]. For actions before this threshold the probability of none detection for each is multiplied together. Probability of none detection is simply one minus the probability of detection given [20]. This number is then multiplied by the probability of guard communication giving the final probability of detection, in this case being 86%. This is an estimation of the probability that an attacker taking this route through the facility would be detected [20].

As was previously mentioned EASI uses equation 2.1 to calculate the probability of interruption, this can also done by hand and will be presented here. To determinate which tasks detection probabilities are relevant the delay times are subtracted from the response force time starting with task nine until zero is reached. This occurs during task five, therefore only the first five tasks are considered. The probability of detection for theses tasks are then used as the P_{D_i} terms in equation 2.1. P_C is also taken for the chart. This gives a final probability of interruption of 0.860. This differs slightly from the result given by EASI. This is mainly because EASI accounts somewhat for time of detection as well as standard deviation. If all standard deviations are set to zero and locations set to beginning the same result is achieved.

Included in the probability of detections of both methods presented for each task are multiple assumptions. Each includes the basic efficiency of each detector present [8, 20]. This is a measurement of is how often the detector indicates a detection when an event occurs within its detection parameters. A similar procedure involving the

non-detection probabilities is then carried out for each detector. Also included is the probability that the adversary fools the sensor in some manner [5]. This is a product of the adversary's equipment and capabilities laid out by the design basis threat. Due to the large number of possibilities, adversary path analysis can have a great degree of uncertainty for more complicated systems [5,8,20]. Finally this number must also include the probability of undetected malfunction causing the sensor to not detect the adversary.

The probability of interruption found using an adversary sequence diagram and adversary interruption analysis is an estimate of how likely an adversary following the given path through the facility will be detected. Depending on the adversary path analysis and the design basis threat, adversary sequence diagrams for many different scenarios may have to be constructed [4]. Depending on the facility these can get quite complex and have a high degree of uncertainty.

To obtain the final probability of effectiveness of a facility the probability of interruption found must be multiplied with the probability of neutralization. As mentioned previously this is often taken from a neutralization chart, however it can also be obtained using Neutralization, a code designed to work with EASI. Using both these methods the number of guard force members and adversaries must be specified and their armament know. With this information a probability neutralization can be found on a chart or generated by Neutralization which simply uses an internal chart and the rule of two to generate and value. For example assume the attack discussed previously consisted of five adversaries armed with pistols and three defenders armed with rifles. Using equation 2.3 in conjunction with a $P_{N_{i,j}}$ from Table 2.1 given as 0.112 and a k of 2 due to the rule of two this gives a probability of neutralization of 21%. Neutralization gives the same value. This means the overall probability of effectiveness for this facility against this attack is found by multiplying the probabilities

of neutralization and interruption giving 18%. This probability can now be used to make a judgment call on whether or not security is sufficient. In this case it is likely not and could vastly be improved by increasing the number of guards.

2.4.4 Live action exercises

Another method used to gauge the effectiveness of physical protection systems as well as to test effectiveness of certain systems and scenarios are live action exercises, also referred to as force on force exercises, or penetration testing [7, 8]. This involves physically acting out an attack on the facility and recording the results [5, 6, 19, 21]. This can be done in a variety of ways. One method is for the response force to split into two groups, sometimes referred to as red team and blue team, with one half playing the response force (blue team) and the other half playing the attackers (red team) [21]. Red team can also be played by some external group, often selected by the regulator [21]. Red team attempts to simulate an attack based on the information in the design basis threat and blue team attempts to stop them, this is sometimes called penetration testing [21]. Red team may also be played by an outside independent group. These are very valuable as they can illustrate ways of fooling the sensors or blind spots that may have been over looked.

For full scale facility tests, live action exercises can't be done frequently due to time and cost constraints [19]. It is for this reason that many large power facilities only run a few each year [5, 21]. At smaller facilities this can be even less [5, 21]. Smaller exercises or drills can be run more frequently, however these often only test smaller components of the physical protection system or a few aspects of response procedure. These can give valuable information, however they do not provide comprehensive results as they do not incorporate a full scenario [5]. Smaller exercises can still be useful for obtaining data for models. This includes obtaining detection probabilities

for small scenarios under certain circumstances as well as probability of neutralization tests as discussed earlier [5].

2.4.5 Table-top Simulation

It is not always practical to hold live action exercises within the facility due to cost and time constraints, however the utility of the exercises for training and analysis are still desired. For this reason simulations of live action exercises are often done on a miniaturized mock up of the facility, these are often referred to as Table top simulations [21]. This generally involves a human player, or teams of human players, controlling both the adversaries and the response force by moving representations of these forces around the mock up. Stochastic methods are generally used to determine the outcome of events such as detection and combat [21]. These allows designers to better visualize the attack as well as train security forces on defense strategy without disrupting the facility. While these simulations do not need to be real time they still are limited in how many can be run due to human involvement. Given these are not done within the facility itself the simulations require simple approximations of sensor behavior such as those used in interruption analysis mentioned earlier [21]. This can impact the simulation's accuracy. Table top simulations also suffer from the same difficulty of obtaining statistics as live action exercises due limited sample size. These however still make useful tools for designers as the visual and tactile element can be significantly clearer than adversary sequence diagrams.

2.4.6 Computer Simulation

The previous two methods are both valid under certain circumstances. However it is desirable to have a middle ground incorporating some of the realism of a live action

exercise with the more quantitative abilities of adversary sequence diagrams. This is where synthetic environment modeling is used. It involves simulating the facility in software with as much detail as possible and running scenarios similar to live action exercises [7, 22]. This includes features such as 3-D modeled buildings. Synthetic environment modeling allows for live action exercises to be played out with virtual agents, with the red and blue teams' actions controlled by a set of rules in the software simulating appropriate reactions [7]. This software also controls detectors and other facility features. There is a continuum of this kind of simulation depending on the level of detail required as can be seen in Table 2.3. Level one contains models such as adversary sequence diagrams while level six has simulations of the facility as close to reality as possible, using humans to control both red team and blue team such as the table top simulations mentioned earlier [7]. The levels in between have various degrees of detail in their models and varying capability for automation. Synthetic environment modeling falls into the area covered by levels three and four.

The highest level of detail and accuracy is not always desirable. Levels five and six require humans to interact with the simulation, and this means that it must be run in real time or close to real time [7]. These are generally war gaming simulations were one person controls the actions of many agents attempting to counter a threat with the information available to them through simulated sensors. The requirement of real time reduces the number of simulations that can be run in a period of time. With levels three and four many simulations can be run and averaged to estimate the probability of effectiveness of the facility as well as the probabilities of interruption and neutralization [22, 23]. Simulations can be averaged for the human controlled models but fewer simulations can be run and biases from the human players may be introduced [7].

Table 2.3: Different modeling approaches for security simulations [7].

Level	Type of Model	Level of Detail of Detection and Delay Models	How Guard-Adversary Combat is Modeled
1	Analytical (point estimates)	Parameters set using point estimates and/or aggregated values	Point estimates of Response Force Time
2	Analytical (stochastic)	Parameters set using distributions based on tests and/or uncertainty	Distributions for Response Force Time
3	Stochastic simulations with simple models	Detailed performance models including interaction between security features and time-varying performance	Node Adjacency models - if guards, at node i, see/are seen by adversaries at node j, what is the probability guards win the ensuing confrontation?
4	Stochastic simulations using agents	Detailed performance models including interaction between security features and time-varying performance	Computerized agents represent the behavior of security and/or adversary personnel.
5	Stochastic simulations using human commanders	Detailed performance models including interaction between security features and time-varying performance	Humans play the role of security or adversary commanders in the simulation.
6	Stochastic simulations using human participants	Detailed performance models including interaction between security features and time-varying performance	Humans play the role of specific security or adversary personnel in the simulation.

2.4.7 Existing Codes

Currently a wide variety of codes are used for analysis of nuclear security. EASI and Neutralization are two codes that have already been mentioned [3,20,24]. They were developed by Sandia National Labs for the NRC (Nuclear Regulatory Commission) as an easy to use implementation of the basic effectiveness equations [20,24]. These codes are very popular due to them being easy to use and widely distributed. These fall in level one in Table 2.3, analytical point estimates. These are useful for getting quick estimates however they do not take into account a large number of factors such as individual sensor functionality and the importance of events interacting [24]. These codes also break up interruption and neutralization.

Many codes follow EASI and Neutralization's approach of dividing interruption and neutralization. BATLE (Brief Adversary Threat Loss Estimator) is an analytical model used to simulate small engagements [24]. This model finds the probability of neutralization similarly to Neutralization however takes into account more factors that effect the outcome of an engagement. BATLE uses a wide range of military engagement data to account for attrition rates and time dependent factors that are not accounted for in simpler models [24]. As most of these factors are handled by BATLE without user input the code is easy to use with minimal input form the user. This code still takes an analytical approach falling into level one in Table 2.3. The minimal input makes BATLE easier to use however it limits the scenarios that it can simulate [24].

SAFE (Safeguards Automated Facility Evaluation Methodology) is an automated analysis tool that uses EASI and BATLE along with a facility layout to find the effectiveness of a physical protection system [24]. SAFE works by first having the user implement a layout of the facility into the computer. This includes components such as walls, fences, and doors as well as the detection and delay probabilities associated

with them. Using criteria specified by the user SAFE then attempts to find a minimum path through the facility [24]. Criteria that can be specified are adversary task time and adversary detection probability [24]. This path is the implemented by SAFE into EASI and and a probability of interruption is found. Using specified adversary and response force characteristics BATLE is also used to determine the probability of neutralization. SAFE then uses these to find the probability of effectiveness. This approach automates much of the analysis process, however it still suffers from the limitations of EASI and BATLE. SAFE is a useful tool for early analysis before using what was learned implement a limited number of scenarios into a more complex tool [24].

A variety of stochastic simulations also exist. This includes FESEM (Forcible Entry Safeguards Effectiveness Model), a Monte-Carlo model that utilizes compartmental modeling to estimate the effectiveness of a facility [24]. This model falls under level three in Table 2.3. Barriers and detection probabilities are input similarly to an adversary interruption analysis chart however a finer division of barriers and detection is allowed. The key difference is that the mean and standard deviation are used to sample from a normal distribution [24]. This allows for the outcome of latter events to be dependent on earlier outcomes giving a more realistic response. The combat model is a coupled set of differential equations, one for the response force and the other for the adversaries. These allow for a time dependent simulation of combat outcomes which is desirable as this more accurately reflects reality. The simulation then uses Monte-Carlo methods by running the scenario many times to estimate the probability of effectiveness [24]. As a compartmental model FESEM can be used to simulate simple scenarios to more complex ones. It is still however limited in the amount of interactions it can have due to the increasing complexity of set up the more interactions there are. FESEM is also limited in the amount of human factors

it can simulate. FESEM is also more difficult to access than the previous codes [3].

Similar to FESEM is a model called SNAP (Safeguards Network Analysis Procedure). SNAP is also a stochastic compartmental model that uses Monte-Carlo methods to estimate the effectiveness of a physical protection system [3]. SNAP differs in its implementation of this methodology. SNAP allows the construction of scenarios my linking together modules, in a symbolic manor functioning similar to a programing language. These modules are grouped into three sub models: The facility, the guard force, and the adversary force [24]. Due to the large number of modals available and the complexity of linking them while quite complete SNAP can be difficult to use. It can also be difficult to visualize the scenario [24].

The solutions presented here are not an all encompassing list of models available however they are a representative sample of those commonly used in industry. Each code has its own strengths and weaknesses, however many of those that can do more complex analysis suffer from being more difficult to use. There is also a gap in models that fit into level four in Table 2.3. A model that fills these two criteria could be a valuable tool.

2.5 Monte-Carlo Methods

Monte-Carlo methods are a class of techniques for simulating physical models using random sampling to obtain numerical results [25, 26]. In the case of physical protection systems, this involves using random numbers to sample detection probabilities, combat outcomes, and to some extent human behavior. Through the law of large numbers, by running a large number of simulations the probability of a particular occurrence can be reasonably approximated [25]. This means that if enough trials are run fluctuations in the mean of the results will stabilize around the expected value,

which in the case of Monte-Carlo methods is the value of interest. This is only valid, as the name suggests, when a large number of trials are run. In the case of nuclear security, the main point of interest to investigate is the probability of effectiveness of the physical protection system and defense force. Monte-Carlo methods are useful as they can be used to estimate complex systems where the outcome is not always easy to predict, such as those present in defense scenarios [25, 26].

Monte-Carlo methods can be used to investigate physical protection systems in a number of ways depending on the method used to construct the simulation model as seen in Table 2.3. For the first level, Monte-Carlo methods can not be used as only average and aggregate numbers representing over all process are input to these models [7]. For levels two to four some level of Monte-Carlo analysis can be used [7]. All of these models in some form have distributions representing the probability that certain events occur that can be randomly sampled to obtain a weighted random response to what occurs. This can be done for every event, including events such as passing through a sensor's detection envelop for detection, time to complete a task, a choice between multiple tasks, combat models etc.. These probabilities can be sampled sequentially forming a chain of events leading to a final probability of the defense force winning. By following this procedure many times the the overall probability of effectiveness of the physical protection system can be found using the outcome of the individual trials [25, 26]. These models do not require Monte-Carlo methods to give results as averages can be used. However Monte-Carlo methods can give more nuanced results with more interactions taken into account [7].

2.6 Summary

Presented here are the techniques used to quantify elements of nuclear security as well as current methods used to estimate a physical protection systems effectiveness. These can be used to both design new facilities and to ensure current ones can withstand new attacks. These rely on knowledge of the various detection, delay, and response elements of the physical protection system which will be discussed in the next chapter.

Chapter 3

Literature Review - Practical Application

3.1 Design

Physical protection system design centers around preventing unacceptable consequences to the facility when faced with the design basis threat [4]. The design process is iterative. As new components are proposed the design is analyzed and evaluated using the techniques and tools described earlier as well as the one developed for this thesis [5, 21]. If the design fails to meet the objectives it must be modified to correct the vulnerability found. This process is repeated until the facility can be reasonably assured to defend against the design basis threat [5, 19]. The probability of effectiveness that qualifies as reasonably assured is difficult to define, the exact value will depend on various factors such as the the facility in question, the threat to the facility, economic concerns, and so on. [4, 9].

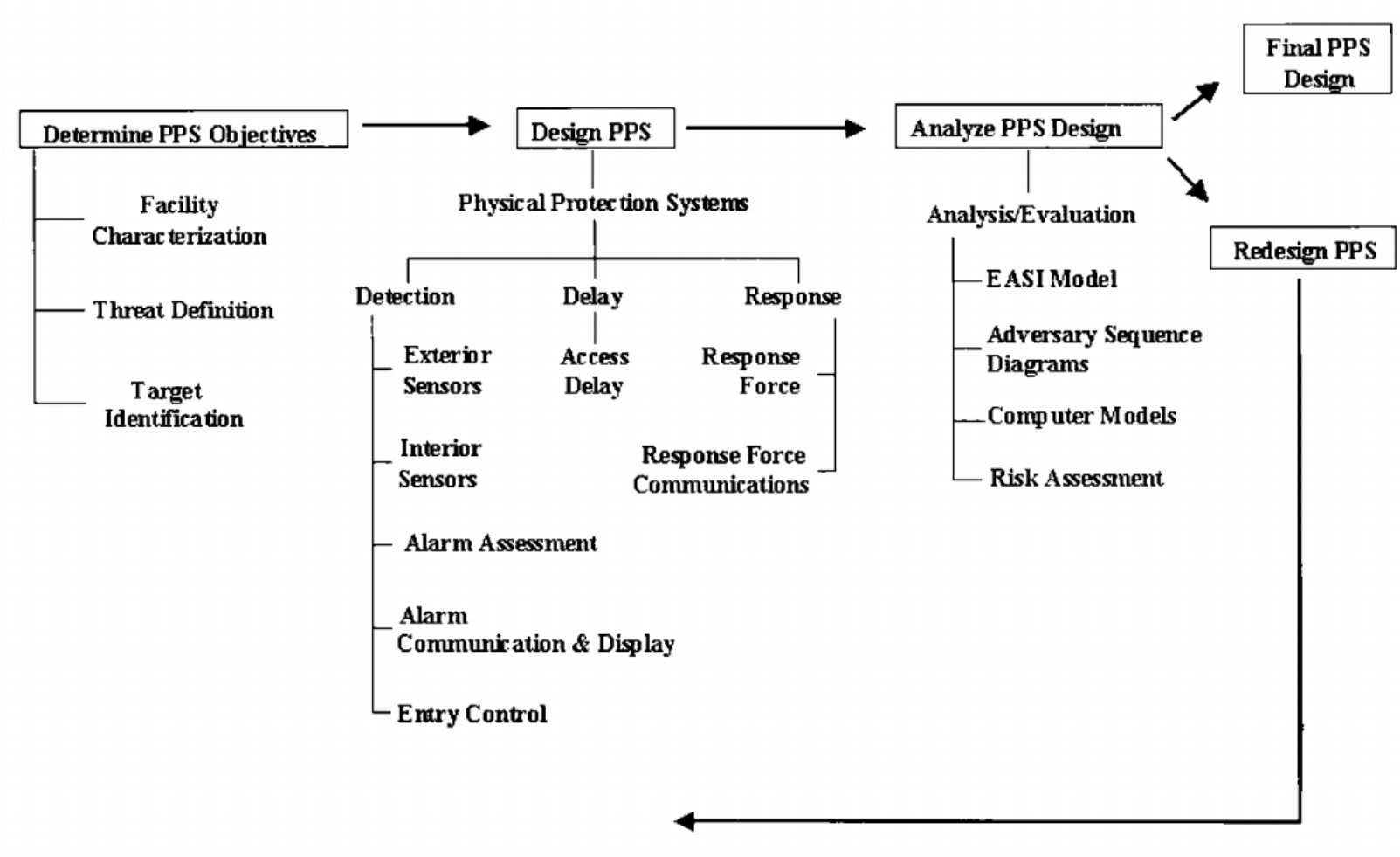

Figure 3.1: Flow diagram of the design and evaluation process of a physical protection system [5].

A graphical representation of this development procedure broken down into components can be seen in Figure 3.1. The objectives of the physical protection system are informed by components of the design basis threat [4]. The design of the physical protection system involves the three elements mentioned in Chapter 2: detection, delay and response. The physical protection system is constructed using a combination of these elements. They follow the idea of defense in depth. Designers try to put as many layers of security between the adversary and target as possible, thereby minimizing the consequence of any one failing [27]. A facility's security system often has many components of each element making optimizing them a non-trivial task [5]. The final design stage is review where it is determined if the elements used are sufficient. If not, the procedure is repeated until they meet the required effectiveness. It is in this last step where tools must be used to verify the physical protection system. Easier to use and more accurate tools can increase designers' confidence that the proposed design can reliably defeat the threat [19, 21]. A similar procedure is used to evaluate existing physical protection systems.

3.2 Operation

To function correctly the three elements of a physical protection system must work together. An alarm is useless if not monitored and even if detected an adversary can not be stopped if there is no response force [3]. Additionally no delay mechanism is perfect, necessitating both detectors and a response force. To maintain the level of protection of the original design, elements of the physical protection system must be verified on an ongoing basis [16]. This includes testing sensors to ensure they still function correctly, verifying that physical barriers remain structurally sound with no means of avoiding them such as overhanging trees, and maintaining a training regime

for the defense force to ensure they are prepared.

Sensors are only useful when they are trusted; not only must a sensor indicate when an adversary is present it must have a minimal amount of nuisance alarms. These are alarms caused by something other than an adversary, such as wildlife or weather triggering an alarm or some kind of malfunction [3,17]. These types of errors reduce operators' trust in the sensor possibly leading to slower response times or no response at all. If a sensor indicates a nuisance alarm a couple of times a week, an operator is not likely to respond the same way as if the sensor only triggers when needed [17].

It is also necessary to perform analysis on the effectiveness of the physical protection system on an ongoing basis. This is done using the same tools used during design. In addition, this is when live action exercises are conducted. As mentioned in Chapter 2 the threat to a facility is not static, therefore ongoing analysis is important to ensure the facility can meet an updated threat [9, 20].

3.3 Detection

3.3.1 Overview

Detection is one of the three key elements of a physical protection system. It informs the defenders of an attack on the facility and allows them time to interrupt the adversary. If an adversary enters the facility undetected, delay elements prior to detection do not add time for the response force to react [3,17]. Detectors take many forms with each functioning on different principles, which lead to each having different strengths and weaknesses [3, 17, 28]. Visual sensors, for example, function poorly in low light conditions while an infrared sensor continues to function. It is crucial that detectors functioning on different properties be paired together at important points

in the physical protection system. This allows the sensors to compensate for each other in the areas that they are deficient [3, 29]. The following discussion identifies the major kinds of sensors used at a nuclear facility and how they function.

Most sensors can be fit into multiple broad categories [3, 28]. The first of these are whether the sensor is passive or active. Passive sensors detect some type of energy being emitted by the adversary such as the sound made by the adversary walking. Active sensors emit some form of energy themselves and detect a change in the received energy. This is how an infrared sensor detects something passing through the beam. Passive sensors have an advantage in that it is more difficult for the adversary to know the limits of the sensing area as there are no emissions for them to detect. The advantage of active sensors is it is often harder to trick the sensor into a false negative [3, 5].

Another property a sensor can have is whether it is covert or visible. A covert sensor is in someway hidden from the adversary such as being buried. A visible sensor is, as the name suggests, visible to the intruder such as a camera mounted on a support structure. Covert senors are more difficult for an intruder to detect but visible ones may deter an intruder. Visible sensors are also often easier to maintain as they are more accessible [3, 5].

Next there are line of sight vs terrain following sensors. Line of sight sensors include detectors such as cameras that are blocked by intervening objects or terrain. Terrain following sensor's detection is dependent on the feature it is imbedded in and does not require line of sight. This includes detectors such as fence vibration sensors that follow the fence they are mounted on. This allows line of sight sensors to operate in a variety of environments where terrain following sensors would function poorly and vice versa [3, 5].

Finally there are volumetric and line sensors. Volumetric sensors detect an adver-

sary entering a volume, found in detectors such as cameras, while line sensors detect occurrences along a line such as fence vibration sensors. Volume detectors by their nature are harder for adversaries to determine the exact extent of the detection area. Line sensors can function in areas were many volume sensors would be blocked by intervening objects [3, 5]. The following sections discuss specific sensor types. Although a non-exhaustive overview is provided, these sensors are the most commonly found at nuclear facilities.

3.3.2 Microwave

Characteristics: Active, Visible, Line-of-Sight, Volume

There are two main types of microwave sensors: bistatic and monostatic [3,29,30]. Bistatic microwave sensors consist of two parts, a transmitter and a receiver. The transmitter outputs a continuous microwave signal while the receiver looks for changes in the beam which indicate motion within the detection area [30]. Due to reflection of microwaves back towards the receiver from objects moving near the beam, the cross section of the detection volume perpendicular to the antenna widens until its maximum diameter, this is a function of emitter strength as well as emitter and reviver separation. This occurs half way between the transmitter and receiver. The detection cross section then begins to narrow again until it reaches the receiver. The volume of detection lies between these with detectors on average having a practical range of 100 meters before detection efficacy begins to drop [3, 29]. A zone of no detection exists for the first few meters in front of either antenna as the beam has not yet reached its full width in the vertical direction, creating a risk for crawling adversaries which must be accounted for [3]. Microwave sensors are vulnerable to uneven ground causing shadows in their detection area so the location at which they are installed must be carefully chosen [5]. Monostatic microwave detectors function

the same manner as bistatic ones however the transmitter and receiver are co-located. These sensors must be pulsed and then look for a change in the reflected energy and have a shorter range to reflect this [3]. A sample of both kinds of sensors can be seen in Figure 3.2.

Figure 3.2: microwave detectors, left is bistatic, right a monostatic [31].

3.3.3 Infrared

There are two general categories of infrared detectors: active and passive.

Active Infrared

Characteristics: Active, Visible, Line-of-Sight, Line

Active infrared sensors work by emitting parallel beams of infrared light towards a receiver which can then detect any interruption in this beam indicating something passed through the beam [5, 29]. By using light in the infrared region these beams can not be seen by the human eye making them harder to avoid, however the emitters

and receivers can still be seen. Generally multiple beams are used to create a larger coverage area in order to make it more difficult to avoid [29]. Detection range varies based on atmospheric conditions and is limited by the distance at which a beam can remain coherent and be aimed correctly. Atmospheric conditions such as fog will also affect infrared sensors reducing their effectiveness [5]. Due to the narrow detection plane infrared sensors can be avoided by either going under or over them [5]. An example of an active infrared sensor can be seen in Figure 3.3 on the left side.

Passive Infrared

Characteristics: Passive, Visible, Line-of-Sight, Volume

Passive infrared sensors still work on the principle of detecting infrared emissions however they do not emit any nor do they require a beam. Humans emit infrared radiation in the form of thermal energy. This can be picked up by passive sensors as a means of intruder detection [5, 32]. The sensor is broken up into many small regions in a checker board pattern. They function by detecting a difference in thermal energy between adjacent regions [5, 32]. This pattern allows the sensor to avoid nuisance alarms due to natural temperature fluctuations as it detects differences between two squares rather than an overall change [5]. Because they function by detecting a difference in thermal energy they work best when the background is at a much different temperature than the intruder. Due to this these sensors work best on cold days when they can be functional to around one hundred meters. This range is reduced the hotter the outside temperature gets [5]. Passive infrared sensors are vulnerable to nuisance alarms due to many prevailing conditions such as animals, blowing debris, and adverse weather such as snow. For this reason they are often used indoors although they can still function outside [32]. These sensors can be defeated by any method that shields the intruders thermal signature including something as basic as blocking the sensor's

field of view. An example of a passive infrared sensor can be seen in Figure 3.3 on the right side.

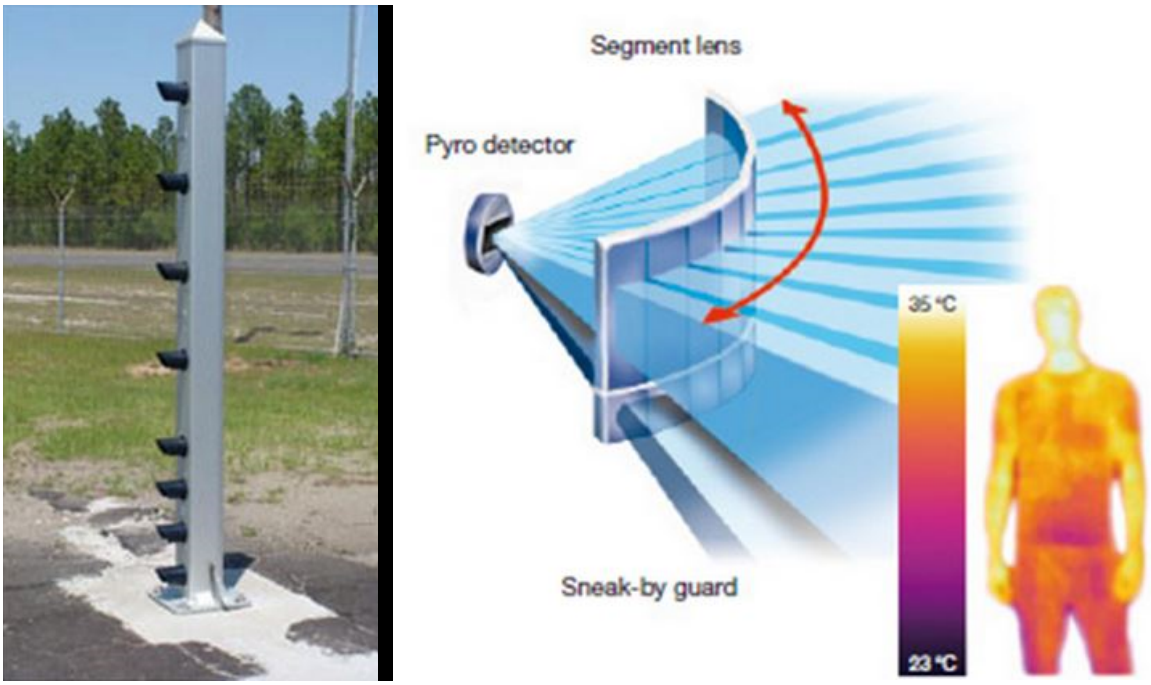

Figure 3.3: Infrared detectors, left is active [33], right a passive [34].

3.3.4 Visual

Characteristics: Passive, Visible, Line-of-Sight, Volume

Visual sensors come in a wide variety of forms but the most common at a nuclear site are closed-circuit television camera [5]. While cameras have been used in security situations for a long time, technology has changed how they can be used. Through use of computer processing, motion can be detected against a static background allowing a detection to be registered without human involvement [5,17]. This has the advantage of removing the human element, however increases the number of nuisance alarms due to wild life and weather. To use cameras at night lighting must be provided and the range at which they work is highly dependent on atmospheric

conditions and intervening terrain. Cameras can be defeated by taking advantage of poor visibility conditions and camouflage. These weaknesses can be partially mitigated using methods such as night vision or thermal vision which use parts of the light spectrum normally beyond human perception to allow visual detection under these circumstances. Cameras are also used to verify detections to avoid dispatching the response force for nuisance alarms.

3.3.5 Fence Associated

Disturbance Sensor

There are two main types of fence associated sensors: disturbance and capacitance.

Characteristics: Passive, Visible, Terrain following, Line

Fence disturbance sensors can work in a variety of ways and have various different implementations however they all aim to detect when a intruder is attempting to climb or cut a fence [5, 35]. This is done with with transducers of various types such as fiber-optic cable, piezoelectric crystals, strain sensitive cables, etc.. These all function by converting vibration in the fence into an electrical signal that can be detected [5, 35]. It is also possible to make the entire fence out of wire connected to transducers that function in a similar manner. This type has lower instances of nuisance alarms [5]. When the fence moves due to an adversary climbing or cutting the sensors will alert the defense force. The fence must be sturdily installed as shaking due to wind can cause nuisance alarms. Tunneling or any other method of getting around the fence without touching it can defeat this kind of sensor [5]. An example of a fence disturbance sensor can be seen in Figure 3.4 on the left side.

Capacitance Sensor

Characteristics: Active, Visible, Terrain following, Volume

Capacitance, or electric field, fence sensors are a volume sensor unlike their fence disturbance counter part. These sensors work by sensing the change in capacitance between two electrically isolated wires when an adversary approaches the fence [5, 28, 29]. Since the air acts as a dielectric medium between the two wires, when an adversary approaches the capacitance of the air is changed, producing a measurable result thereby indicating a detection [3, 5]. These changes can be detected up to one meter away from the fence depending on how the sensor is configured. The higher the sensitivity however the more potential there is for nuisance alarms. This method also requires all metal objects within range to be grounded to prevent nuisance alarms including the fence itself [5]. The range beyond the fence that these sensors have make them harder to defeat than disturbance sensors. An example of a capacitance fence sensor can be seen in Figure 3.4 on the right.

Figure 3.4: Fence associated detectors, left is disturbance [31], right capacitance [36].

3.3.6 Buried Sensors

There are a variety of buried sensors that rely on various phenomena to detect what is occurring on the ground above them. The main varieties are: pressure, magnetic feild, ported cable, and fiber optic

Pressure Sensors

Characteristics: Passive, Covert, Terrain following, Line

One type are pressure sensors. These are generally pressurized liquid filled tubes connected to a transducer. When the ground above them is disturbed in some way, for example walking or digging, the forces acting on the tube change alerting the operators [3,17,29]. These types of sensors are generally sensitive to about 1.5 meters away, although this varies depending on the soil and burial depth. The sensitivity of this kind of sensor is heavily impacted by frozen soil [3]. While it is difficult to locate the sensor, if its location is known simply bridging the location of the gap will allow the sensor to be avoided as movement above the ground is not detected.

Magnetic Field Sensors

Characteristics: Passive, Covert, Terrain following, Volume

Another type of buried line sensors are magnetic field detectors. These are a buried coil of wire that have an induced current when something metallic passes near it, changing the local magnetic field [3,29]. Magnetic field detectors are mainly used to detect vehicles. This type of sensor can be defeated if the adversary is not carrying any metal or only has small amounts of metal. Local electromagnetic disturbances such as lightning can also cause nuisance alarms in this kind of detector [29].

Ported Cable Sensors

Characteristics: Active, Covert, Terrain following, Volume

Ported coaxial cable is another type of buried cable detector. It works by having two coaxial cables buried parallel to one another. Both cables having a series of regularly spaced holes in the cable's shielding. A signal is run through one of these cables and due to the holes leaks out into the surrounding medium in the form of a electromagnetic field [3, 29]. This field makes it to the second cable interacting with it creating a coupling. When an intruder passes above these cables they disturb the established field which can be detected in the coupled cable [5]. The volume in which the intruder can be detected extends about a meter above the ground and about two meters surrounding the cables [5]. These types of sensors are not disturbed by frozen ground however moving water will induce a nuisance alarm. Standing water as well as large stationary metal objects can also distort the signal creating an area of no detection.

Fiber Optic Sensors

Characteristics: Passive, Covert, Terrain following, Line

The final kind of buried line sensor is fiber optic cable. Fiber optic cables are very fine, transparent cables that light can be transmitted through. The light diffraction pattern at the other end of the fiber is highly sensitive to the shape of the fiber meaning that even the smallest distortion can be detected [5,17]. These are sensitive enough to detect the slight distortions caused by an adversary standing on the ground above the cable. Fiber optic cable does not have a large range of detection so is therefore often woven into a mesh before it is buried to give it a larger area of coverage [5]. The mesh can be very sensitive to vibrations in the soil and therefore create a high number of nuisance alarms when located in packed ground nearby machinery.

3.4 Delay

3.4.1 Overview

Delay is a crucial part of any physical protection system; the defense force can not be everywhere at once. They need time to respond and delay elements allow this. Delay elements can take many forms but all, in some way, slow the adversary down either by forcing them to choose a different, longer route or to take time to disable the barrier [3, 5, 17]. As has been stated previously, a barrier is only useful if the adversary has been detected, this means that delay and detection elements must work in tandem to provide adequate protection.

Nuclear facilities are often large and complex, so it is difficult for a single delay element to effectively cover the entire facility [3]. Some concessions must also be made to allow access or maintenance, as well as those necessitated by function. This means that a physical protection system is only as strong as its weakest link; an adversary is not going to attempt to penetrate the thick concrete wall when they can simply breach the simple wooden door that goes through it [3, 17]. When selecting delay elements for the physical protection system it is important to keep this in mind.

Not all delay elements function in the same way, for the purpose of discussion here they can be categorized as passive or dispensable [5]. Passive delay elements including barriers such as walls and fences are the most common and the most recognizable. These can also include natural barriers such as distance and overfill [5]. Dispensable barriers are those that are only in place during an attack. These can include materials such as adhesive foams or other entanglement devices or techniques such as sonic irritants [5].

3.4.2 Passive Barriers

Passive barriers are those that do not require any active input from the guard force to delay the adversary. These come in a variety of forms including: fences, vehicle barriers, structural components, penetration coverings, and natural features.

Fences

Fences are one of the most common passive barriers at a nuclear facility as they are relatively compact and cheap to build as well as having a wide variety of sensors available. The primary problem is that common link fences are insufficient as a delay barrier since they can be penetrated quickly with simple hand tools [5]. To overcome this, most fences used at nuclear facility are much sturdier than the standard chain link fence and have additional features such as barbed wire, thicker links, and stronger supports. As many fence associated sensors require the fence to not move outside of an intrusion scenario, fences used must be reinforced to not move in the wind or due to small animal interactions. A major feature used to increase detection is the double fence, an example was seen earlier in Figure 1.1 [5,29]. This is generally done only for more sensitive areas such as those around the reactor building, or protected area, as it increases the size and cost of the system. This technique allows for a wide variety of sensors to be placed between the two fences to compliment each other, making it very difficult to avoid detection [5]. To increase penetration time for a simple fence, techniques such as the triple fence seen in Figure 3.5 can be used. These increase delay time in multiple ways. Firstly the roll of barbed wire on top vastly increase the difficulty of climbing or bridging the fence. Secondly the rolls of barbed wire between the fences increases the difficulty of cutting through the fence as significantly more difficult material must be passed through [5]. To undetermined adversaries, a sufficient fence can act as a deterrent.

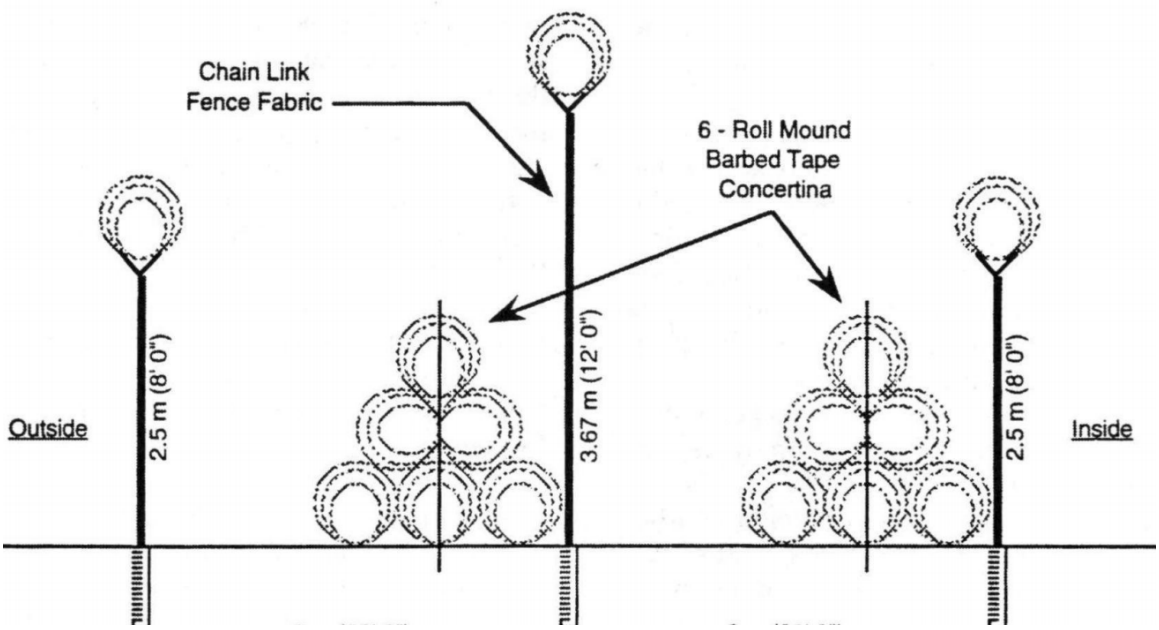

Figure 3.5: Triple fence delay setup [5].

Vehicle Barriers

Fences are relatively easily defeated by motor vehicles and as such different methods must be taken to prevent breach in this manner. Due to the advantage that a motor vehicle gives an adversary, from carrying heavier tools such as large gas powered saws to speed in reaching the facility quicker and removing more material, it is important to ensure that at the very least they do not get into the facility undetected [17]. For this reason areas that are hard to observe must have more robust vehicle barriers [5]. These barriers range from simple concrete cylinders sunk deep into the ground, to more complex arrestor systems [5].

Concrete cylinders or other firmly anchored solid barriers form a ridged object that can stop a vehicle in a very short distance with little deformation [5,29]. These kinds of barriers are difficult to defeat in a rapid manner assuming they are tall enough that they cannot be bridged due to their solid construction. Depending on the size of vehicle anticipated these can also be surrounded by a crash cushion, empty steel

barrels for example, that will deform absorbing some of the force [5]. The downsides are that they cannot be placed where authorized vehicles will need access and for a large facility they can be expensive. Temporary barriers that function in the same way can be made from water filled plastic containers. These should only be used in monitored areas as they are easier to defeat and cannot stop as very large vehicles [5].

The arrestor system method of stopping vehicles aims to stop them over a longer distance. An example of one of these systems is a cable barrier. Although a vehicle can crash through it, the cables become attached to the vehicle by the force of the crash. These systems utilize flexible cables and fixed poles, utilizing the elasticity of the cables to provide the force to gradually slow a vehicle [3, 5]. These systems are also often designed to attempt to mechanically interfere with the vehicle such as tangling up the wheels. They can be cheaper to install but must be set up in a monitored area as they can be defeated by cutting the cables before driving the vehicle through [5]. This can be done with relative ease if the adversary has not been detected. he arrestor method method also requires significantly more space as the vehicle is not stopped immediately. An example of this kind of barrier can be seen in Figure 3.6.

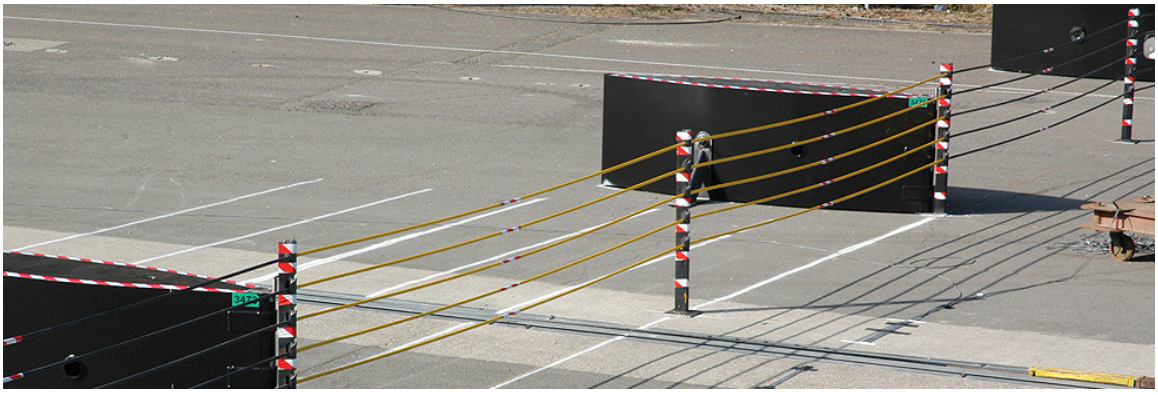

Figure 3.6: An example of arrestor wire [37].

Structural Barriers

Structural barriers include elements such as the walls, ceilings, and floors of the facility. The primary goal of these components are structural in nature, supporting the facility and allowing access to vital components. This does not mean that they cannot be vital to security; the opposite is true. While they must be present, if they are not hardened to attack they will be a liability in the physical protection system [3,5]. It is often the case that the doors, windows, and other such penetrations in the concrete structure are the weakest point in a given physical protection layer and are thus the most vulnerable to attack. This means that after a certain point reinforcing the concrete structure against attack is not a viable method of increasing penetration time [5].

Thick concrete walls are a common feature in many structures at a nuclear power plant. These walls are often steel rebar reinforced, adding both structural stability and increasing the time it takes to penetrate [5]. In fact the largest contributor to the time it takes to penetrate a reinforced concrete wall with explosives is cutting through the rebar that is left after the explosive charge removes much of the concrete [5]. This can take many minutes, therefore it is often more practical for an adversary to attempt to breach some form of penetration such as a door or window. It is important to design physical protection systems such that there are minimum penetrations in the reactor building or other equivalent targets [5]. Materials such as wood can be used for construction however these offer substantially less protection. It is often practical to upgrade a structure to be made of reinforced concrete if it is security critical.

It is still possible to breach the concrete structure and this may be the most desirable course of action if some of the walls of the facility are comparatively less guarded and if this offers a shorter path to the target [3, 5]. In these cases it is best

to have more walls between the target than a single wall of twice the thickness. This is due to the steel reinforcements providing the increase in penetration time and the initial blast causing the highest likelihood of detection [5]. Increasing the number of walls that must be passed through then increases the number steel reinforcements that must be penetrated and requires more blasts to get through increasing protection over the alternative of a single wall.

Penetrations

As was mentioned previously, often the weakest aspect of a physical protection system are the doors and other penetrations in the concrete structure of the building surrounding the target [5, 17]. Doors come in many varieties, from standard personnel doors to thick steel vault doors. Standard wooden doors such as those used on a house offer little resistance to a determined adversary with access to the correct tools [19]. Properly mounted vault doors on the other hand can be very difficult to penetrate, substantially increasing adversary delay time. In any case during working hours many doors must be unlocked representing a further security liability that must normally be accounted for with a higher guard presence while the doors are open [17]. This can be mitigated by using various forms of access control.

The principle of balanced design states that to increase the effectiveness of the physical protection system the weakest aspect, often being the penetrations, must be improved. One primary way to do this is to limit the number of windows and doors and reinforce those that are left. Methods of penetrating a door vary as do the methods of reinforcing them. Doors with physical key locks are vulnerable to lock picking techniques so it is best if these can be avoided [5, 17]. Doors with exposed locking mechanisms are also vulnerable to direct attack of the locking mechanism to defeat the lock [5, 17]. If the hinge pins are exposed this is an avenue for attack

against the door even if they are unremovable, as a focused attack on the pins is simpler than defeating the entire door [5, 17]. Hollow steel doors present at most industrial sites are vulnerable as they are relatively easy to create crawl through holes although simple interior reinforcements make this more difficult. Finally the door frame is also an avenue of attack as an improperly mounted door can be ripped out frame and all [5, 17]. Most pedestrian doors are still vulnerable to blasts after they have been reinforced. Blasts however are very easy to detect, increasing the likelihood that the attack will be noticed.

Natural Barriers

Designers often take advantage of the protection that natural barriers can provide. If a facility is difficult to penetrate by vehicle forcing the adversary to go by foot space can add a significant amount of delay [3]. Forcing the adversary to cross long distances can also increase the chance that they are detected as well as make it difficult to carry many large, heavy tools. Another way natural barriers are used is by making a portion of the facility underground. If the adversary's target is buried with limited access points, it is easier to defend the small entrance, increasing the likelihood they are detected and stopped [17]. Finally natural features such as bodies of water or cliffs can make natural barriers that are difficult to traverse undetected making them an unattractive option for adversaries with little to no cost [17].

3.4.3 Dispensable Barriers

Dispensable barriers vary from traditional passive barriers as they are only active during an adversary attack. These kinds of barriers require input from either a member of the guard force or sensor but some activate based on inherent properties of some other barrier [3, 5]. An example of these inherent properties would be a hollow

wall filled with an irritating, expanding substance. When the wall is penetrated the substance escapes and impairs the adversary. These types of delay elements are generally expensive and require more maintenance than passive barriers making them economical only very close to the asset [5]. Many dispensable barriers are being developed and can function in vastly different ways. A small sample of those that exist are presented here to give an idea of how this kind of barrier can function.

One type of dispensable barrier that has been developed is adhesive foam [5]. As the adversary attempts to remove the assets a large volume of incredibly adhesive foam is released, adhering to the adversary and to the target. This foam hinders movement and makes it very difficult for the adversary to effectively escape [5]. There is also aqueous foam or smoke. These function by filling an area making it difficult for an adversary to see and as such difficult for them to complete their malicious actions. These can also contain eye and skin irritants making it even harder for the adversary to continue to act [5]. Finally there are entanglement devices that in some way make it difficult for an adversary to move through an area such as a net placed above an area that drops based on a sensor, making it difficult for the adversary to move [5]. Some of these can be seen in Figure 3.7

Figure 3.7: Example dispensable barriers, L: aqueous foam, R: adhesive foam [5].

3.5 Response

3.5.1 Overview

The response force, sometimes called the defense force, is the final element in a physical protection system. They are relied upon to interrupt the adversary and neutralize them before they can complete malicious action against the facility [3, 17, 19]. Without the response force any delay mechanism would eventually be breached, leading to a failure of the physical protection system [3, 5]. They are generally an armed force that can mobilize to anywhere on the facility quickly in numbers large enough to defeat the adversary. The exact requirements for the defense force's capabilities are based on the threat assessment and design basis threat for the facility [4]. Some facilities might have on-site defense force while others rely on police response or some combination of the two. The response force may be armed with assault rifles and specialist equipment or pistols. Tactics will also vary depending on facility and threat [3, 19]. The guard force is a separate group responsible for the day to day security of a facility as well as the verification of alarms.

3.5.2 Organization

Guards

The guard force is a key component of any physical protection systems. This is the group responsible for the day to day security such as monitoring the alarm systems and performing access control throughout the plant [3, 5]. Depending on the facility this force may or may not be armed however they are not the force that responds to interrupt an adversary [5]. The guard force communicates the alarms to the response force as well as performs verification of the alarms. Nuisance alarms are

a normal occurrence for any physical protection system and are not desirable for the response force to respond to. The guard force attempts to verify alarms with either cameras or in person [5]. The guard force may also perform patrols of the facility as both a deterrence and detection measure. Depending on the site the guard force and response force may be one integrated unit however individuals within this organization will fulfill different roles of either the response force or guard force [17].

Responders

There are two kinds of response force, on-site and off-site. An on-site response force is one that is stationed at the facility. This has the advantage of fast response times and usually better integration with the guard forces for the purposes of communication and so on [3, 5]. This can be impractical for a small facility, since a response force large enough to stop an adversary attack would be very expensive to maintain. For this reason some facilities rely on an off-site response force. An off-site response force can be a separate group employed by the facility or it can be the local police force or military [3, 5]. This can be cheaper however the response time is likely to be much longer meaning that the delay elements will have to be functional for much longer. Depending on the threat to the facility this may however be sufficient. A combination of both elements is common, a small response force on site to control and contain the situation while an off-site response force is dispatched [3, 5]. This will be discussed latter.

3.5.3 Equipment

Communication

Communication equipment is an important part of the response force functioning at peak efficiency. Without the ability to effectively communicate alarms, the response force will not be unable to respond to them [5, 17]. Communication at a nuclear facility is often done with the use of radios due to its ease of use and reliability [3, 5]. There are some problems with radio however, mainly the unsecured nature of it makes eavesdropping by the adversary a possibility, as well as signal strength issues at large sites [5]. The signal issues can be addressed with more powerful transceivers however eavesdropping is a continuing problem. This can be solved in two ways. The first is to use frequency hopping radios. These systems are preset to change the frequency used at certain short intervals to make it difficult for the adversary to stay on the correct channel [5]. This is done through a pre-programmed frequency pattern that is input into all plant systems. Another method is coded transmission, by overlaying electrical noise on top of the transmission before outputting it. The receiving end will be preset with the same noise and remove it [5]. Both of these are still vulnerable to enemy capture of equipment. It is important to have an alternate method of communication as well, should radio be unavailable. This can be achieved through plant wide public announcement systems or through hard wired intercoms [5].

3.5.4 Strategy

Containment

The strategy that the response force takes is highly dependent on the facility layout, the threat to the facility, and the type of response force being used. If the threat to the facility is theft of material, a containment strategy may be initially sufficient

[5]. In this type of strategy the guard force does not attempt to prevent the initial theft of material but instead places themselves such that the adversary can not escape the facility [3, 5]. This gives the response force more time to react as they do not have to interrupt the adversary before they get to the target. This also allows for a larger off-site response force more time to arrive while a smaller force contains the adversary [5]. The initial force may also be off-site as well given the longer time to initially respond. Eventually the adversary will have to be defeated either through force or surrender so containment is only an intermediate step.

Denial

The denial strategy involves preventing the adversary from reaching the asset in the first place. If the adversary's goal is sabotage, the containment strategy will be insufficient as simply reaching the target allows them to complete their goals. In these cases the denial strategy must be used [3, 5, 17]. This means there is less time for the guard force to intercept, making it more suited for an on-site response force. If an off-site response is to be used the physical protection systems must have sufficient delay for the response force to have time to arrive and intercept [5]. This strategy also requires the response force to have enough members and equipment to be reasonably assured to defeat the adversary without outside assistance. This often means outnumbering the adversary and having similar or better equipment. The choice of strategy is often governed by the design basis threat, as if the threat is sabotage it is the only choice [5].

3.5.5 Summary

This chapter outlined how the various elements of a physical protection system and showed how they work together to detect, delay, and respond to an adversary threat.

Modeling how these elements work is a key component of effectively estimating the effectiveness of the system. Using both the theory presented earlier and the more practical aspects presented here the problem of developing a model for estimating the effectiveness of a physical protection system will be analyzed. This analysis will be used to develop a model to allow effectiveness to be better and more easily estimated.

Chapter 4

Problem Statement

4.1 Current Limitations

Current methodologies of physical protection systems analysis such as adversary sequence diagrams and live action exercises are useful but lack some capabilities. Adversary sequence diagrams are rapid to set up, however they are less rigorous as they rely on data with a large degree of uncertainty due to using simple point estimates and aggregate parameters [7,8]. They are also difficult to construct for more complex systems such as when many routes are possible with many different physical protection systems interacting [20]. For these cases a large number of different diagrams have to be produced for very little output that is difficult to verify [3, 4]. Live action exercises closely approximate an attack on the facility, however they are expensive and disruptive, and consequently cannot be run frequently. This low frequency means conclusions are often drawn from small sample sizes. They also involve human factors that can be a source of bias in the results [21]. For this reason it would be desirable to have a method that has some of the advantages of both systems and mitigates some of their weaknesses.

Synthetic environment simulation attempts to emulate live action exercises in varying levels of detail using stochastic models [7]. These models can vary depending on what is required from simple distributions to detailed performance and behavior models. Synthetic environment simulation sees wide use in industry for training and simulation purposes, mostly focusing on those with human actors controlling red and blue teams [7]. These allow for training of personnel in a synthetic environment and allow for an approximation of live action exercises without many of the limitations. These kinds of simulation still require human players however so still suffer from human bias as well as the constraint of having to be run in real time [7]. This limits the sample size that can be taken from these types of simulations.

Currently, for nuclear security applications very little is being done with synthetic environment simulations with computerized actors. These types of simulations are valuable because they try to simulate reality as closely as possible allowing higher accuracy and clarity. Some simulations exist, however are generally inaccessible and require expert modification to change the scenario [16, 38, 39]. This is due to the highly sensitive nature of these simulations in applications such as military simulation [39]. These black box solutions are used in industry, however modification is generally done on a contract basis, making rapid modification difficult [39]. The advantage of synthetic environment simulations with computerized actors is the ability to use it to preform Monte Carlo analysis. By running the model many times the effectiveness of the facilities physical protection systems can be found based on the win rate of the defense force [7, 23]. This allows more scenarios to be tested than other methods relying on human interaction. These scenarios also have a larger sample size as each scenario is run many more times than a live action exercise possibly could. These simulations still use approximations for sensor behavior depending on how they are modeled. They are best used to try a large number of scenarios to

select interesting ones for further testing using other methods [7, 23]. A synthetic environment simulation is the type of model that will be investigated in this thesis.

4.2 Proposed Model

4.2.1 Goals

Designing a synthetic environment simulation of the physical protection systems at a nuclear facility that is capable of being easily modified for rapid scenario prototyping would be a valuable tool for analyzing physical protection systems [16,39]. This would fill the gap between simple analytical models using point estimates and averaged properties, and full stochastic simulations both at the facility itself and in a synthetic environment. It would also be an asset to the iterative design process laid out in Chapter 3. The goal is to create an easy to use synthetic environment for designing and testing a wide variety of scenarios quickly and easily before committing to more expensive actions. Rapid scenario design and prototyping will also allow iterative improvement of live action exercises as exercise can be used to improve the model which then can be used to improve the exercises etc. [7]. Some modeling solutions in this niche exist, however access is limited even to those designing the facility and often require changes to the model to be implemented by the contracting company [38, 39]. For this reason it would be desirable to have a model that is easy to use and available for designers to modify quickly as well as for researchers studying the field.

4.2.2 Model Requirements

To fulfill the niche caused by the lack of synthetic environment security models the model designed must;

- Be able to emulate the physical layout of the facility so that it is a reasonable approximation of the physical protection systems present such that the simulation is easily comparable to its live action exercise counterpart.
- Include sensors that perform in a similar manner to those present within the physical protection system. The sensors should have detection probabilities that can be verified by real world exercise.
- Be able to find an approximation of the probability of effectiveness, portability of interruption, and probability of neutralization of the physical protection system with a reasonable degree of certainty.
- Be simple enough to make small changes to the scenario to allow rapid prototyping of new scenarios.
- Have enough flexibility to be used for a wide variety of scenarios.

By fulfilling these requirements it is intended that the tool will be useful for security experts to narrow down the scenarios necessary to be tested with live action exercises. It is also hoped that this tool will put more control in the designer's hands when it comes to sample size for validating data. The intent is to use the model in conjunction with live action exercises. The synthetic environment can simulate more exercises than could be done in real time and the live action exercises for the most critical scenarios can verify the results.

4.2.3 Approach

Software Selection

The decision was made to modify a commercial software application to meet the requirements in order to complete the project within a reasonable period of time.

This imposed an additional set of requirements for the selection of software;

- Have as many of the required simulation components as possible already present.
- Be simple enough to modify so that it can be learned in a reasonably short time frame to add missing components.
- Have a graphical interface that can be used to create scenarios for better user friendliness.
- Have available and accessible support and training to get up to speed on how to use it as fast as possible for both initial tool designer and end user.
- Be available within a limited budget.

To meet these requirements software to be considered was limited to force on force simulators as they were the most likely to have features that cross over with those needed. The available software mainly consists of war gaming simulators. Many of these such as SAS (Strategic Analysis Simulation) used by the U.S.A. military and the CAX System (Computer Assisted Exercise System) used by NATO are intended to have humans controlling the bulk of the units making them less ideal [40, 41]. Many of these tools such as those previously mentioned are very difficult to access by those outside government and military facilities. Ease of use is intended to not only makes it easier to modify the tool but also simpler for the end user to create and modify scenarios.

To meet these requirements the force on force modeling software STAGE was selected [7,42]. STAGE was primarily chosen due to the ease of access to the software as it is produced by a Canadian company and is commercially available. Other commonly used synthetic environment solutions for force on force simulations such as MANA (Map Aware Non-Uniform Automate), Pythagoras, JCATS (Joint Conflict

and Tactical Simulation), and JANUS were significantly less accessible so were not considered [39]. The next step was to modify STAGE to add the capabilities not already present.

Modification

In order to use the STAGE software many modifications had to be made. As a force on force simulator mainly intended for combat simulation, there are limited sensor capabilities. These had to be added to allow for realistic adversary detection [42]. It was desirable to implement them as detailed performance models for individual sensors to allow for as close a comparison to physical reality as possible. STAGE also has limited statistical capabilities so many had to be developed and implemented [42]. This includes the capability to utilize randomness in its scripting engine as well as more random target selection and firing order for the controlled agents. In order to obtain statistics such as the probability of effectiveness for the physical protection system, Monte-Carlo methods were implemented. This involved external code communicating with the STAGE simulation engine and running a scenario many times.

The STAGE simulation engine has many features to assist in these modifications. Many solutions involved making use of STAGE's scripting engine to modify the behavior of agents and sensors when necessary [42]. STAGE also has the ability to incorporate user written plug-ins to change the behavior of the simulation at a more intricate level when necessary. Finally it is possible to pipe data to and from the STAGE simulation engine using a command line interface allowing for external code to interact when necessary [42].

Usage

STAGE helps to meet the model's goal of user friendlessness through its graphical interface and mission scripting language [42]. Initially users must implement a 3-D model of the facility into the STAGE environment to provide the basis of the simulation. The sensors developed previously are then placed around the facility and the properties of the sensors can be set based on those present in the actual facility. This only has to be done once unless testing is being done on the placement of sensors. Next, response force and guard behavior can be set using the mission editor based on facility procedure. Finally the attack is implemented using a similar method. All of these properties are reasonably easy to change allowing many scenarios to be developed and tested quickly. Obtaining the effectiveness of the physical protection system for the scenario is then done by inputting the file name into the Monte-Carlo code and prompting it to run.

4.3 Summary

In this chapter the limitations of current methods of estimating the effectiveness of physical protection systems were discussed. Using the material previously covered a solution using synthetic environment modeling and Monte-Carlo methods was proposed to fill this gap in analysis tool capabilities. The requirements for this tool as well as the approach towards the creation of the model was also laid out. In the following chapter the force on force modeling software STAGE that forms the basis of the tool will be discussed as well as the methodology used to modifying it in order to create the final effectiveness analysis tool.

Chapter 5

Methodology

5.1 Overview

STAGE is a synthetic environment development engine produced by Presagis, a modeling and simulation company based in Montreal, Quebec, Canada and a subsidiary of CAE [42]. It offers an integrated development environment for all aspects of synthetic environment simulation from unit definition and scenario creation to agent behavior. It's primary function is to provide military and first responder training exercises in a synthetic environment. Development in STAGE is done through a graphical user interface that allows for modification of scenarios and units in an easy to use visual manner. This graphical approach extends to scenario creation that mirror their real life counter part with increased realism. A primary feature of STAGE is its mission scripting language; it allows for robust character modeling and realistic simulation of complex mission behaviors. Within this scripting language are a variety of dynamic functions for commonly desired behavior such as path-finding and combat mechanics to assist in behavior development. Finally there is robust support for the creation of plug-ins that modify STAGE's behavior to better suit the users

needs. These features provide a strong base for modifying STAGE for use in physical protection system analysis. Presented below is a discussion of the various interfaces of the STAGE environment and the limitations encountered during the creation of the analysis tool [42].

5.2 Development Environment

5.2.1 Unit Library

The STAGE unit library editor allows for the definition of the various agents or units that will be used in the simulation. An example of this editor is seen in Figure 5.1. The set of characteristics for each agent is gathered together under the platform heading seen in Figure 5.1 where each unit is defined, in this example a solider. Platforms have a sub type such as human, but can also be fixed wing aircraft, land vehicle etc.. The sub-type changes what characteristics are available for modification. For example the sub type fixed wing aircraft can have maximum and minimum climb and dive rate defined. Once the sub type is chosen the unit's characteristics must be defined, these include how large it is, how visible to various detection methods, how quickly it can move, what occurs when it is damaged, as well as the various equipment it might have. The example in Figure 5.1 shows the soldier's visibility in generic units to the various types of native STAGE sensors [42].

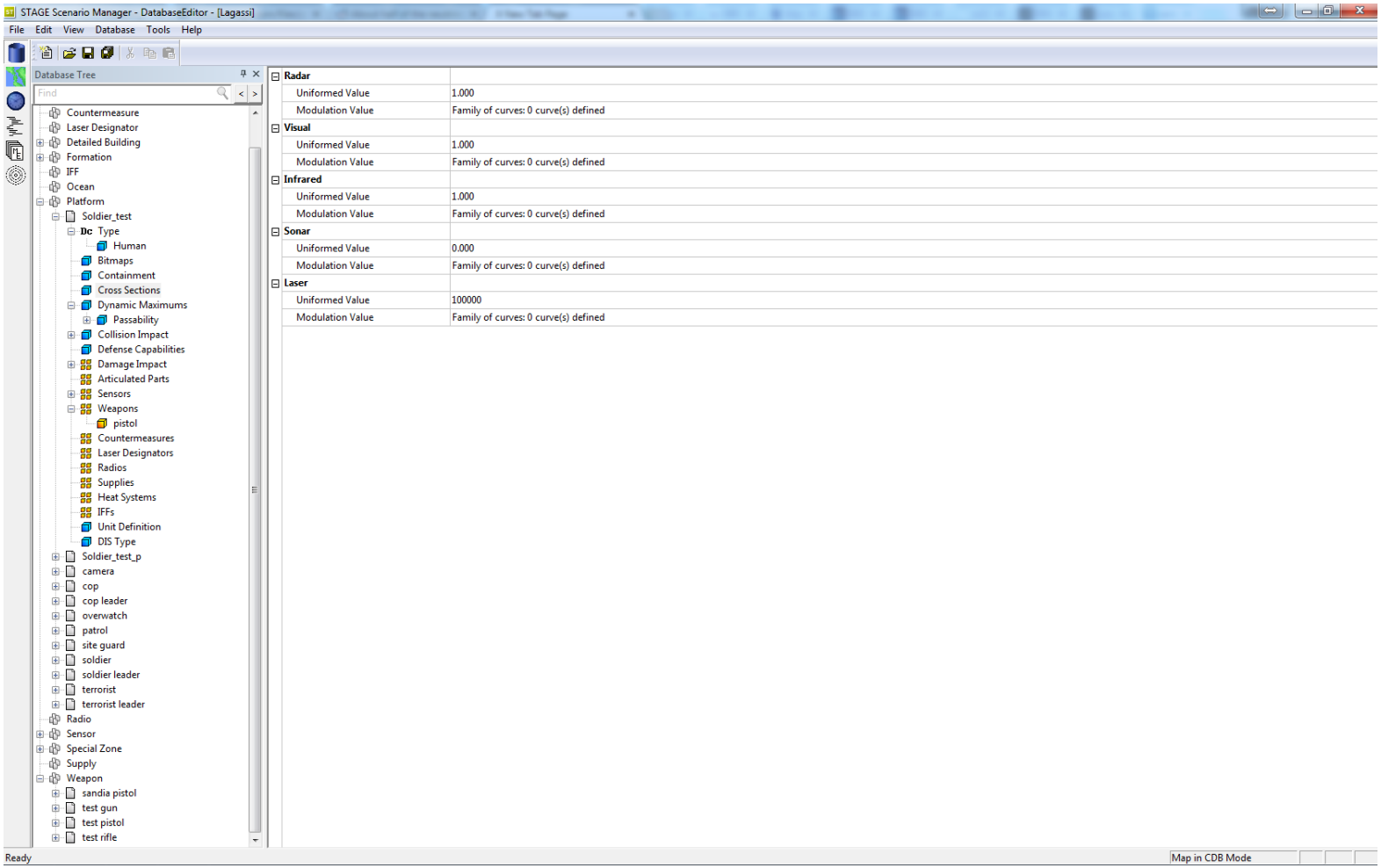

Figure 5.1: STAGE unit libarary [42].

The equipment associated with a platform includes weapons, sensors, radios, and other generic supplies which can be defined as necessary. Each piece of equipment is also defined with various characteristics within the unit library. Weapons are modeled using a method called probability of kill. For each weapon, a curve is generated for the probability that a round fired kills the target versus the distance between weapon and target. An example curve can be seen in Figure 5.2. This number is target independent. A separate curve must be defined, if a different target has a different probability of kill with the curve to be used being selected by the mission scripting language discussed later. Also defined are the weapon's rate of fire and round velocity. Weapons such as missiles can also be defined with appropriate characteristics [42].

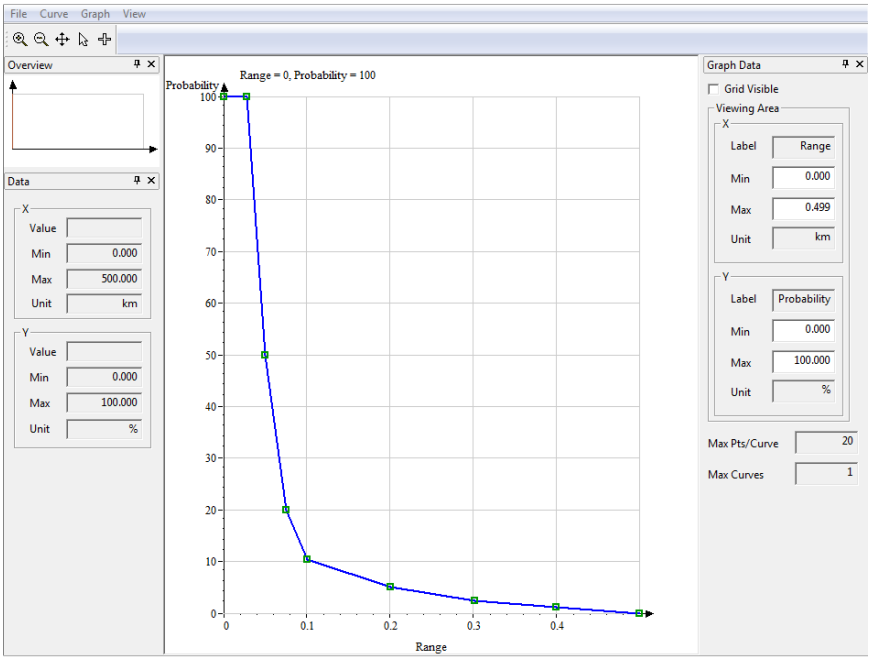

Figure 5.2: STAGE weapon model [42].

Finally the properties of the sensors on each platform are defined. For a human these are generally visual sensors representing the eyes. These are the same sensors that will be used in detectors placed around the facility. The sensors available in STAGE are radar, sonar, passive infrared and visual. Sensors are defined using the

solid angles from the unit at which they can detect, as well as factors unique to each detection method such as a radar frequency. Also defined are the detection curves of the sensor. These are similar to the probability of kill curves mentioned earlier for weapons, however sensors have the probability of detection as a function of distance rather than probability of kill. This causes limitations which will be discussed later. Where sensor definition differs from weapons is that multiple curves can be defined for various 'Z' values. These Z values correspond to to the platform's visibility to various detection methods shown in Figure 5.1. When testing for detection STAGE will use the curve for the appropriate Z value of the unit of interest. If a curve is not given for the Z value an interpolation of the closest two curves is used [42].

5.2.2 Scenario Editor

The scenario editor is used to create the synthetic environment used, as well as to place all the features and agents to be present in the simulation. An example scenario editor screen is shown in Figure 5.3. To assist with this, the scenario manager allows importing various terrain databases to create the basis for the scenario. STAGE supports multiple varieties of terrain databases, CDB and OpenFlight, these are commonly used in the industry [7,42]. These contain information such has elevation, slope, and ground conditions for many points in a given area as well as imagery for STAGE's graphical components. Depending on the accuracy desired, the resolution of these databases can vary from one point every couple of meters down to every couple of centimeters. STAGE defaults to the CDB world wide database which gives ground maps for most of the earth with a resolution of about two meters. These can also contain information about buildings and roads pre-defined for use in STAGE. Buildings can also be placed on top of existing terrain to function much like real buildings blocking movement and sensors [42].

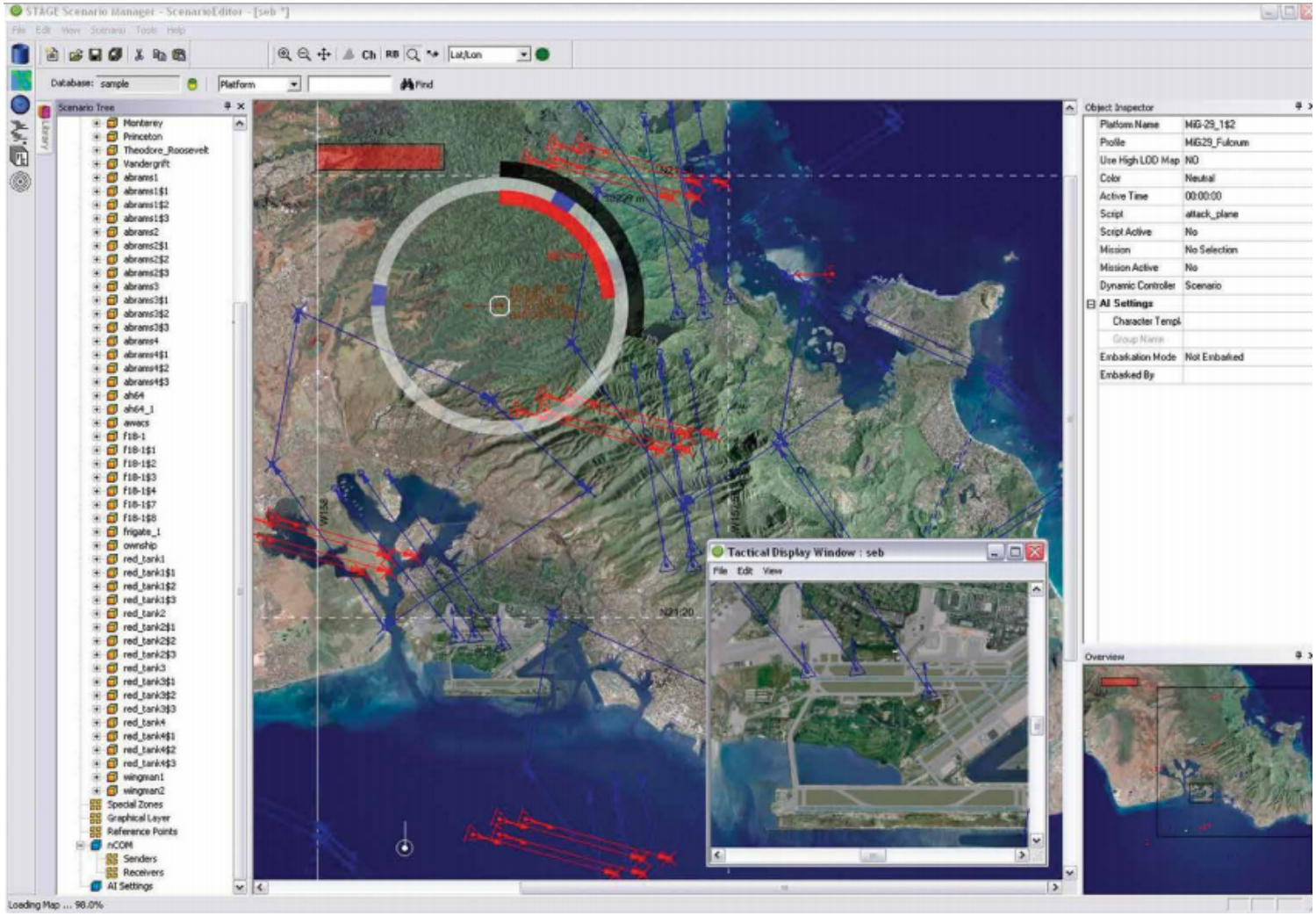

Figure 5.3: STAGE Secenario Manager [42].

Using the scenario tree seen in Figure 5.3, units can be imported from the unit library as created earlier. These units will be given behaviors using the mission scripting editor and can be given their initial conditions such as location, heading, and initial supplies. It is also at this point that the unit's team and mission is set. The opposing teams are red team and blue team with a gray team representing civilians. The unit's mission is the initial behavior it will have, the specifics of which will be discussed in the next section. Besides units, key reference points and special zones features can be created. These are used to indicate a point or area of interest to units for them to interact with [42].

5.2.3 Mission Editor

The mission editor is used to determine the agent's behavior within the scenario through use of the mission scripting language. An example of this environment can be seen in Figure 5.4. The mission scripting language is a simplified programming language consisting of event triggers and if statements organized into task groups. Each task group has an initial condition such as a time, event, action, or fulfillment a certain condition that activates it. These conditions can be based on various properties of the entity or an opponent such as damage level, navigational cues, etc. If all of the conditions for that task are true then the task activates, changing some property of the entity that the mission controls. This could tell the entity to change its heading and begin motion, to fire its weapon for x seconds at y opponent, communicate z to ally, or many other things. Through the use of sub missions a unit can be doing multiple actions at once. For example, a unit could be moving towards a target while looking for an enemy, see the enemy, stop, and fire their weapon. Using these components, human behavior during any kind of event can be approximated. The mission scripting language is intentionally simplistic, as this increases ease of use for the end

user in designing scenarios. Actions and conditions are selected from a list of allowed ones as seen in Figure 5.5. This makes more complicated behaviors difficult, however the mission scripting language can be extended using STAGE's plug in support [42].

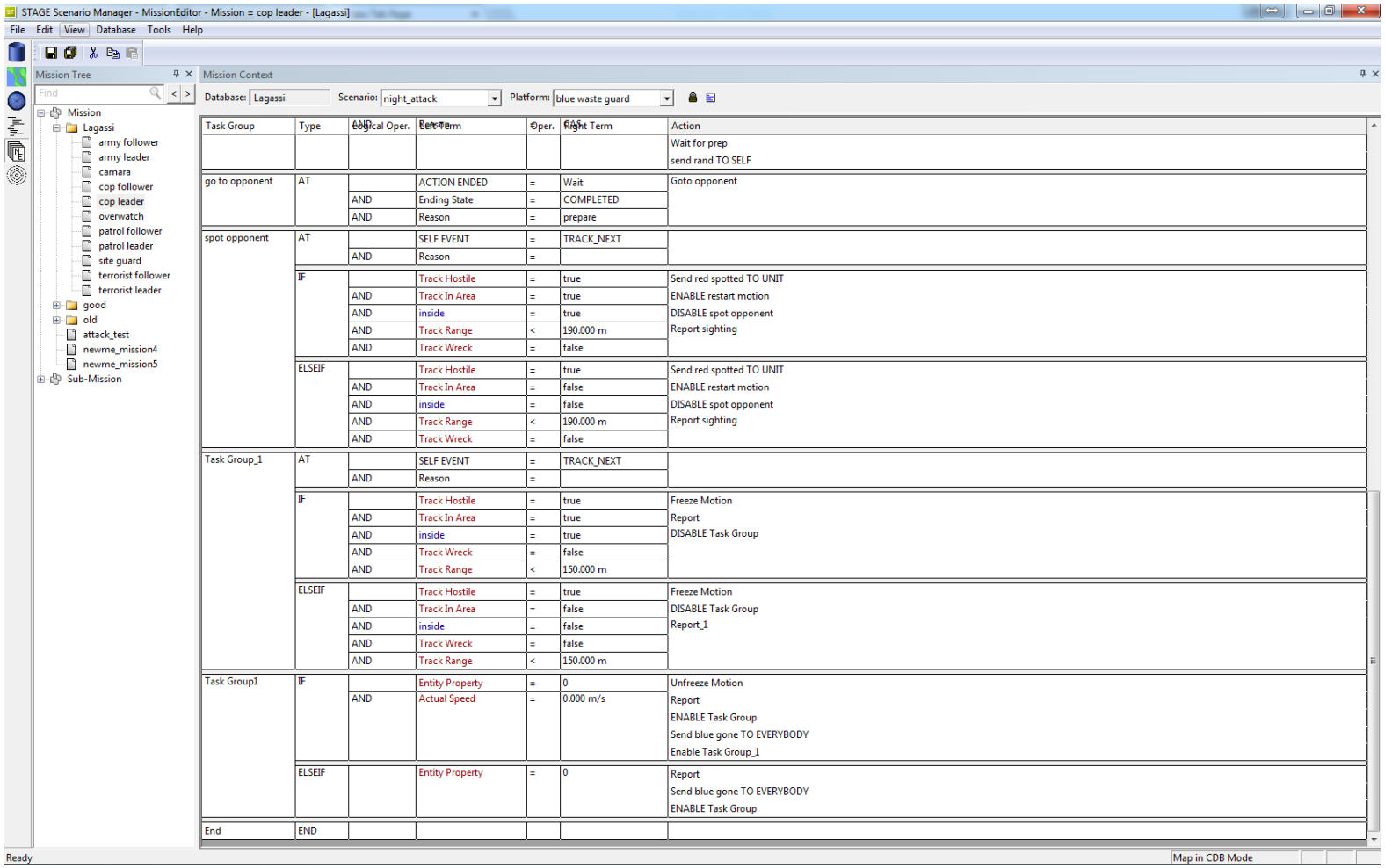

Figure 5.4: STAGE mission editor [42].

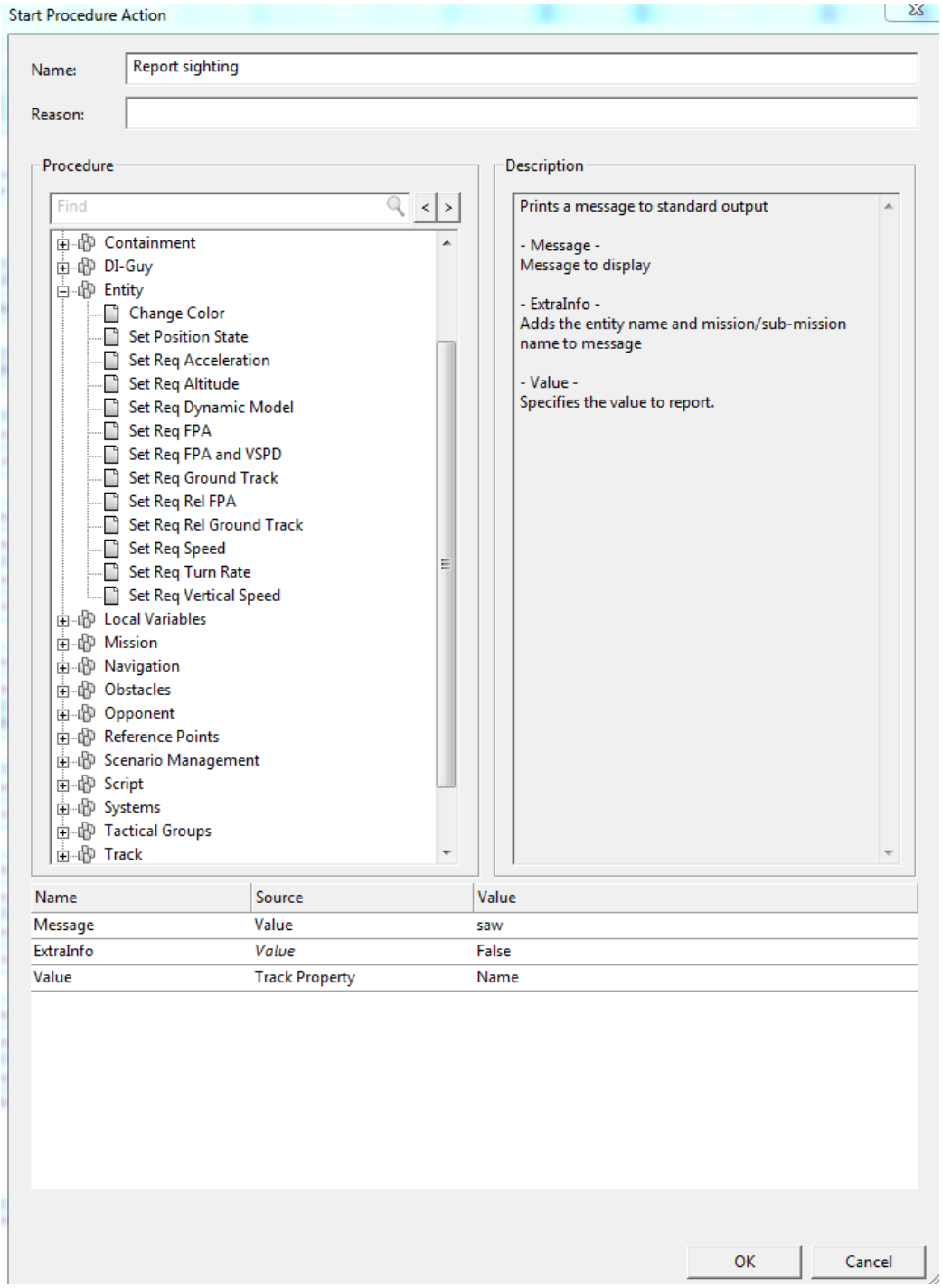

Figure 5.5: STAGE mission editor actions [42].

5.2.4 Run-time Environment

The final component of the STAGE development package is the run time environment. This is the portion of the interface that uses the STAGE simulation manager to run the scenarios created. The view is the same as that in the scenario editor however with additional controls. The play button that starts the simulation will cause the computerized agents to begin following the behavior laid out in their assigned missions. Depending on the mission, a guard may walk a patrol between reference points using his visual sensors to watch for an adversary. Once an adversary enters visual range and is detected the guard will sound the alarm and retreat waiting for backup. If desired and additional step of identifying friend from foe can be implemented. The simplest form of this is to assume every entity detected and not identified as friendly is a foe. Meanwhile the adversary may attempt to close the distance and fire their weapon. Using the run time tools, unit's missions can be overridden and the unit will follow new orders given by the operator. There is a 3-D visualizer that can be used to view the simulation from a fixed point of view or from the point of view of one of the units. The simulation manager also provides a debug menu that outputs messages associated with any report command in each entity's mission. an example of this output can be seen in appendix B. The scenario will run until paused [42].

5.3 Limitations

STAGE has many features that make it useful for simulating force on force engagements in a synthetic environment, however this is only one component of the analysis of a physical protection system. Detection must also be simulated in addition to the penetration time of the various delay mechanisms present. This meant adding a variety of sensors to the STAGE environment as well as creating a method for

penetrating barriers. A way also had to be found to use the simulation to find the probability of effectiveness of the physical protection system. STAGE does not have a native scoring system or any post run statistical abilities. To fulfill the goal of creating a physical protection system analysis tool these features had to be added to the STAGE environment. These modifications were not trivial however they were significantly simpler than creating the entire tool from scratch.

It was decided that using Monte Carlo methods for analysis of the scenarios would be the most effective method for obtaining information from the simulation. To aid this, additional modifications had to be made where STAGE's behavior was not conducive to statistical analysis by Monte Carlo methods. One change that had to be made was to heavily modify the combat mechanics for better target selection and firing order; these were not as random in native STAGE as was required for using this kind of analysis. The system's treatment of random numbers was also insufficient for Monte Carlo and additional capabilities had to be created such as the ability to sample from a distribution. Finally the STAGE engine is very graphical, making it difficult to obtain information from a simulation in an automated method; external code was introduced to gain information from the simulation for use in finding the effectiveness.

5.4 Summary

This chapter outlined the functionality of the STAGE engine. Also shown was how it can be used to create a synthetic environment model that can be modified to be used as the desired effectiveness estimation tool. Chapter 6 will discuss the details of all of the modifications done and the reasoning behind them.

Chapter 6

Model Development

6.1 Overview

In order to fulfill the requirements laid out for the model in Chapter 4 the chosen STAGE software had to be modified to add the required capabilities. The modifications were primarily made using the STAGE mission editor to control the behavior of units however some changes had to be implemented using plug-in support. On top of this some external code was utilized for information gathering and analysis using Monte-Carlo methods. The intent of these modifications was to create a method of modeling an attack on a physical protection system to in order to determine its effectiveness in a user friendly manner.

The first modifications were made to the STAGE unit library which lacked the ability to create many of the sensors used in nuclear facilities physical protection system. STAGE's primary focus is force on force engagements, not detection, so the lack of appropriate sensors was expected. The next modifications involved agent behavior, as some capabilities required, such as proper target selection, were not present and others were insufficient for statistical analysis. A large portion of the work revolved

around designing a method of obtaining statistical results from the simulation. It was decided early on that, for a synthetic environment simulation using computerized agents, Monte-Carlo methods would yield the best results while preserving the models user friendlessness and resemblance to reality. These capabilities are not native in STAGE so it was decided to use an external coding approach. This utilized the STAGE simulation managers ability to run asynchronously without the graphical interface. Finally, for testing purposes various scenarios had to be developed to confirm the model was running as expected. All of the changes are implemented as modules that can be simply integrated into any scenario. The intent behind the modular approach outlined is to make the construction of these scenarios as simple as possible for the end user. These modifications are outlined in the following sections.

6.2 Sensor Implementation

6.2.1 Microwave

Microwave detectors function on line of sight. For this reason it was decided to implement them by modifying the already present visual sensors. A microwave sensor's detection volume is an ellipsoid with relatively equal detection probability throughout the volume, although slightly weighted towards the emitter [30]. STAGE's visual detector is a cone originating at the sensor with a detection probability defined based on distance. The detection probability curve of the visual sensor can be used to implement the microwave sensor's detection characteristics, however the area of detections shape must be modified. This is most easily done using the area of interest tool and the mission scripting language. Oval areas of interest are not available in STAGE, so two circles were used to approximate an oval. More areas can be used if a closer approximation is desired. Mission scripts can only be applied to units. By using a

mission the unit mounted with the sensor can be told to ignore any detection that does not fall within all the areas of interest. The area of interest is two dimensional, meaning that in the vertical direction the detection volumes shape is governed by the visual sensors cone. This causes the side of the volume with the detector to have the expected gap in between the detection volume and the ground for the first few meters. This gap however is not present on the far side of the detection volume. This can be accounted for by placing a second visual sensor on the opposite side with the same field of view. As it is not desirable to double the detection probability this second sensor will have a one hundred percent detection probability but it can only communicate to the other entity involved in the microwave sensor. The alarm is then communicated only if both sensors detect the adversary and they are within the volume of interest. These sensors must be mounted on units representing the sensor poles. The final results as implemented in the scenario editor can be seen in Figure 6.1.

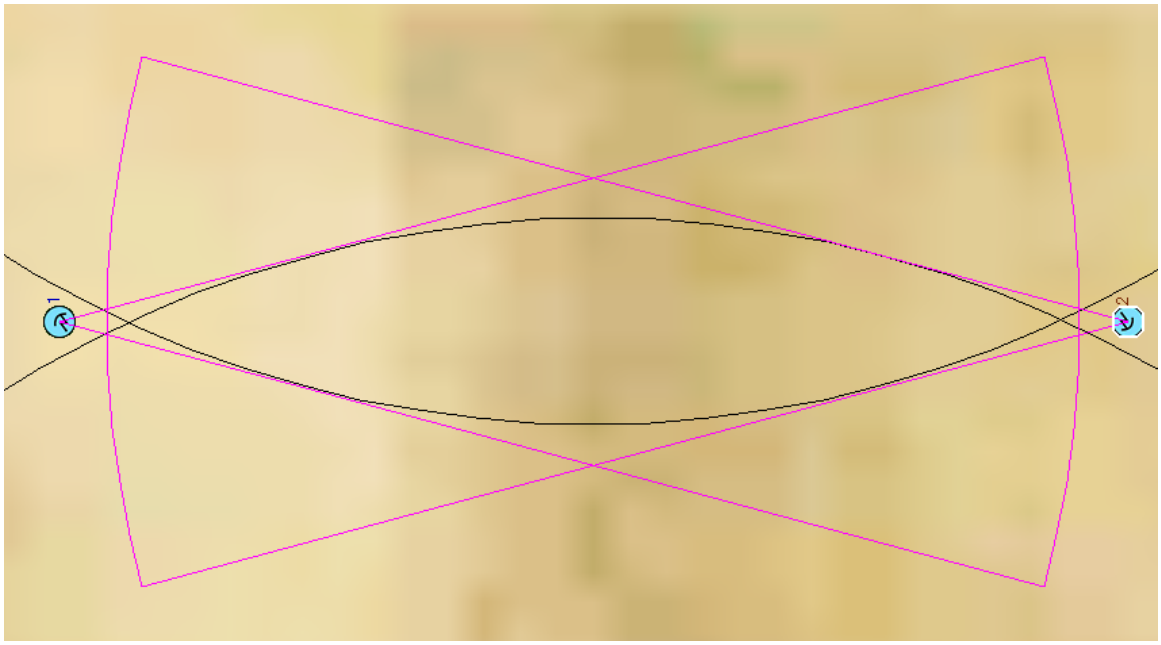

Figure 6.1: Implementation of a microwave detector in STAGE. Black lines are two large circular area of interest, pink lines are detection area.

6.2.2 Active Infrared

Active infrared sensors are both line of sight, and line sensors. No line sensors exist in STAGE and as such their behavior had to be approximated [42]. This is implemented similarly to the microwaves sensors through the use of areas of interest to limit the area of detection of a visual sensor and then to determine detection through mission scripting. The probability of detection curve for the visual sensor is first set to be the same at all distances as it does not matter how far from the emitter the beam is interrupted. The angle above and below the visual sensors emitter that the sensor can see is defined using the unit library tools. This can be used to imitate a beam. However, if both the angle above and below the sensor are set to zero, as it would be for a beam, the sensor will not function. This can be mitigated in a similar manner used for microwave detectors by using two detectors, one with the upper and the other the lower angle being zero. These sensors then use mission scripting to only alarm if both see the adversary. By placing these sensors at different heights a plane of detection can be created simulating the most common fence configuration of active infrared sensor. A single beam can be approximated by placing the upper and lower sensors very close together.

Next, an area of interest is then used to define the width of the infrared sensors beam. There are limits to how narrow the area of interest can be made. STAGE simulates motion by moving entities in steps based on their current speed. If the area of interest is too small the entity will pass through the area but never be in it to be detected [42]. The amount of time each step covers, which for this model is one millisecond, is called the iteration time. This is the clock that many functions are based around. This means that to detect an adversary the area of interest representing the beams width must be at least one thousandth the ground covered in one second. For a car going at 100 km/h this is a width of about three centimeters. Although

this is larger than the real width of the beam, it is a reasonable approximation. This means an infrared sensor can be approximated with two visual sensors and a narrow area of interest with detection only occurring if both sensors detect an adversary within the area of interest. An example of the implementation of this detector can be seen in Figure 6.2.

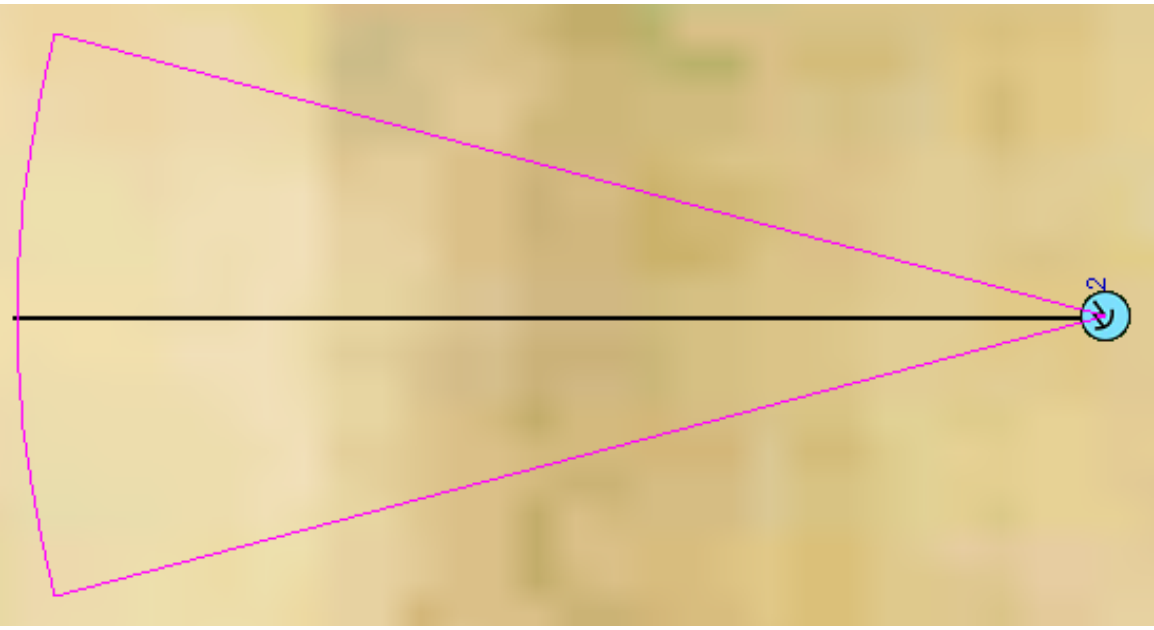

Figure 6.2: Implementation of an active infrared detector in STAGE. Black lines are a rectangular area of interest, pink lines are detection area.

6.2.3 Fence Associated

Line Sensors

There are many types of fence associated detectors that are line sensors, however they can all be modeled the same way with the different detection properties being implemented when placed into a scenario. These types of detectors were more difficult to implement as they do not require line of sight which is the only kind of sensor that STAGE natively has available. Adding to the difficulty, walls and fences are not

intractable objects in the STAGE environment through the mission script. Because a method of penetrating barriers also had to be implemented these two methods were combined. The detection portion of that method will be presented here.

Two different implementations of fence associated detectors were devised, those where detection occurs at the start of a penetration attempt and those where detection occurs throughout the attempt. These two implementations are desirable as some sensors give the probability of detection per second while other give a flat probability and it is desirable to be able to implement both. For the first method once the barrier penetration method begins once every second a pseudo-random number between zero and one is found. If it is less than the detection probability the entity communicates the alarm. The second method is very similar however it generates a pseudo-random number once at the beginning, with usually a higher given probability. This method is simple to modify to adjust for detection at any point during the penetration attempt if so desired by the end user. One curiosity of this method is that it is the adversaries' mission that reports the detection, however the true source is the fence associated sensors.

Volume Sensors

Volume fence associated sensors where simpler to model as techniques used in other volume sensors could apply. An area of interest is placed around the fence and extended out to the range of the sensor. A unit with a visual sensor is placed high above the fence with the sensor aimed down. The visual sensors probability of detection curve can be used to define the likelihood of detection based on distance from the fence. To use this method, simple trigonometry must be applied to convert distance from fence to distance to detector. Detection is then reported the same as other volume methods discussed here. The problem with this method is that the sensor

can not simply be placed at initial design and left for all scenarios. Because the detection probability can only be given as a function of distance from the detector if the detector is not placed directly above the point were the adversary will try to breach the barrier it will not have the correct detection probabilities. This is fixed by moving the detector based on the scenario being examined which is a minor step when constructing a new scenario.

6.2.4 Buried sensors

Line

Buried line sensors have a very similar implementation to fence associated volume sensors with a few key differences. The area of interest with a visual sensor placws far above is still used, however this type of sensor cannot differentiate if the adversary is touching the ground or not. This is a problem that must be accounted for as buried line sensors cannot detect adversaries that are not touching the ground. The mission scripting engine can be used to account for this as opponent height above ground is a condition that can be checked. The visual sensors probability of detection would be constant for all distances in this case as detection is not a function of distance to the detector. This means that an alarm is communicated if the adversary is in the area of interest, on the ground, and a detection occurs.

Volume

Buried volume sensors were one of the most difficult to model correctly as they are the least similar to the native STAGE sensors. The final result has similar implementation as the buried line sensors with some key differences in the mission scripting. Implementing the same height above ground function shown previously, the detec-

tion probability can be adjusted instead of ignored. As there is no capability to lower detection probability with respect to height that would apply evenly to all points within the area one had to be manually created. Various flat detection probabilities are created for the sensor corresponding to height above the ground, these are given Z values. In this case a radar sensor is used instead of a visual sensor as radar sensors are not commonly found at nuclear facilities. This will cause the least issues with changing the entities z values. When an adversary is in the area of interest for this detector its Z value for radar sensors is constantly being updated based on its elevation above the ground. If bridging of the sensor is desired and the adversary height increases the probability of detection will decrease accordingly. This method allows for the sensors properties to only have to be input once, which is desirable.

6.3 Agent Behavior

6.3.1 Combat Model

Target Selection

STAGE has many functions within the mission scripting language to achieve common tasks, one of these is selecting a target. These functions are desirable as they save time and make the end user's task simpler, however while sufficient for large force on force exercises with human participants this function falls short for smaller engagements with only computerized actors. In STAGE the target selection function is referred to as the track cycle. For every iteration of the simulation the unit calling the function cycles through all entities in the simulation that it is aware of and selects the first that fulfills the criteria provided. This often leads to all entities selecting the same target, which is undesirable as in close quarters this unfairly advantages smaller groups since

many of the larger teams shots are wasted. An even greater issue is that these units are cycled though in alphabetical order making the first unit targeted always the same for every run of the simulation [42]. In real engagements, targets are selected using the response forces' best judgment of threat at that moment. This is not possible to be simulated therefore random target selection is the most logical method to use. This was achieved by creating a sub mission in the mission scripting language.

The new mission that assigns opponents is activated every iteration were the unit can detect at least one opponent and the entity is not currently engaged in a different sub-mission. This mission uses the previously mentioned track cycle to first count how many opponents are present by filtering the track cycle to hostile and alive. This must be done as the entity property that counts the number of opponents visible also counts those that have already been destroyed and thus cannot be used. Once the number of opponents is known, a random number from one to the number of hostiles detected is rolled. The track cycle is then allowed to run again a number of times equal to the the random number generated. The opponent selected is then assigned as the entities' target. In the unlikely event the opponent has been killed in the mean time this will be detected by cycling through all opponents and not selecting one, in this case the function restarts. As this is a small edge case this will likely result in a successful assignment on the second time around.

Firing Order

The order in which units fire during a particular iteration is not a function in STAGE but a part of its built in behavior. This is a problem as firing order uses the same alphabetical order as target selection [42]. This causes further issues as firing a gun takes a set amount of time, meaning that if all entities fire at the same time the first half of the alphabet has a large advantage. At first glance this may seem like a simple

fix using similar techniques as the previous example however it is here that some of the limitations of the simple mission scripting language become apparent. Entities do not have the ability to communicate anything beyond pre-set phrases to one another, making coordinating turn order with a variable number of entities in a user friendly way challenging. This could be done by creating an entity with a set message to each individual entity that randomly determines the order in which to send the messages but this requires a large amount of set up for the user each time they wish to make a scenario and is not user friendly. This is also complicated by the varying number of units that may be involved in a scenario with some being killed and others arriving from off-site.

Due to the issues caused by a method revolving around communication between entities the implemented solution instead does not rely on external communication. This does introduce minor inaccuracies as it is still possible for units to fire at the same time however it can be made unlikely enough to not significantly effect the final results. Paired with the randomized targeting the probability of two targets attempting to fire at the same time and one getting killed unfairly before it can do so is very small, as will be shown later. The method chosen involves using the 'wait' command in conjunction with the 'return at frequency' command. The return at frequency function causes an event every specified number of iterations. The same frequency is given to all units at creation to allow a semblance of coordinated action. Every time this event occurs every entity checks to see if it has an assigned opponent, if it does it will begin the fire command. The fire submission starts with a wait command that will wait a random short amount of time before firing, which decreases the likelihood that two opposing entities attempt to fire at the same time. This wait time is slightly shorter than the frequency, ensuring that no entity can get lucky and shoot twice before another has shot once. The wait time is shorter than the frequency

by the amount of time it takes to fire their weapons, in this case one second.

The length of the frequency and wait time is determined by the iteration time chosen for the simulation. The iteration time chosen was one millisecond as this is the shortest possible iteration time allowed. For reasons that are not apparent in the simulation, the shortest wait time possible is thirty times the iteration time [42]. This is likely due to a mistakenly hard coded value somewhere in the STAGE software as thirty three milliseconds is the default iteration time giving a default wait time of one second. A maximum wait time of five seconds was chosen, with a wait times being in increments of thirty milliseconds, giving one hundred and sixty seven possible wait times. Assuming five combatants on either side this gives a probability that two entities on opposing team select the same random number of approximately one tenth of a percent. Most combats last around ten rounds, however the probability of occurrence reduces in later rounds as there are less entities through attrition. Given that the gun models used do not have a guaranteed probability of kill at most ranges and firing simultaneously where the second entity is not killed is not a problem, the probability of the error occurring and significantly effecting the results is less than 0.1%. It is therefore reasonable to use this method as it is more user friendly. The model involving communication is left in and can be used if desired.

6.3.2 Barrier Penetration

STAGE has the capability to make barriers using the ability to insert buildings both at a database level before scenario creation begins and after the fact during scenario creation. It however does not have any native ability for penetrating these barriers without pre-defined doors [42]. As the penetration of delay barriers is a crucial part of physical protection system analysis, a method of achieving this had to be implemented. This can be done two ways, both of which were implemented to increase user

friendliness. One feature present in both models is the delay component of the barrier. Since passing through the barrier is mostly cosmetic and does not slow the unit down, the delay of the barrier must be implemented using some other method. This is done using the mission scripting engine. When an adversary encounters a barrier, a sub-mission is called associated with that particular barrier. Due to STAGE's inability to communicate variables between missions a new sub mission has to be created for each new barrier. A template was designed requiring only barrier properties to be input when a new barrier is created to increase user friendlessness. The required input for this submission is the mean delay time of the barrier and the standard deviation of that delay time. The sub mission then samples from a normal probability distribution to determine how long the barrier will take to penetrate. A normal distribution was selected as exercises preformed by Sandia national labs have shown that delay often follows this distribution [7]. The adversary then waits the generated amount of time before initiating the barrier penetration method chosen. This function is linked with the fence associated detector function with the appropriate submission being started in tandem with this one.

STAGE has no native ability to generate a normal distribution; as this was required for delay time when penetrating a barrier a method of generating one had to be implemented. Given the limited support for mathematical operators within the STAGE mission scripting language the techniques that could be used were limited. The approach settled on was to use central limit theorem. This approach generates twelve uniformly distributed random numbers from zero to one and adds them together and subtracts six. The equation used can be seen here,

$$R_N = \left(\sum_{=1}^{12} r_i \right) - 6. \tag{6.1}$$

The resulting variable approximates a normal distribution bounded to six standard deviations with the fraction of possible values that are lost by limiting to this range being less than one in one hundred million [43]. This is considered an acceptably close approximation. The number generated is then multiplied by the standard deviation and added to the mean to give a randomly sampled number on the desired normal curve, seen here,

$$Delay = R_N\sigma + \mu. \tag{6.2}$$

The provided mean and standard deviation are chosen by the user and represent the barrier in question and the penetration method being used. For example an adversary desires to breach a fence using wire cutters, this has a mean delay of 36 seconds and a standard deviation of 6 seconds. The sub mission uses equations 6.1 and 6.2 to generate a random delay that normally distributed around the mean of 36 seconds.

Actual penetration of the barrier can be achieved in two ways. The simplest is to insert doors into all barriers during the modeling of the physical protection system. This has the advantage of being simple to implement and easy to build a scenario around. To penetrate the barrier the delay sub mission is called and then the built in command for passing through a door can be used. This has the downside of limiting the number of scenarios that can be designed as penetrations must occur at specific locations. By making most walls have a high number of doors that graphically still look like walls this can be somewhat mitigated. This can lead to some confusion later when designing the scenario as many doors will have to be explicitly labeled as off limits, reducing some of the user friendliness advantages. The second method is more complicated to initially set up but allows for more flexibility. This method

involves careful usage of the ground navigation functions to pass through the wall. After the unit has waited the prescribed amount of time unit obstacle avoidance and collision interactions can be turned off, allowing the unit to pass through the wall as if it was not there. While this is turned off the adversary can pass through any wall, which creates a problem when many structures are close together. It is also difficult to determine when these features should be turned back on. The first problem was addressed with the placement of areas of interest underneath all of the walls, disallowing the unit from crossing them with only the intended walls being deactivated. With this solution the second problem was also solved, given that the unit could not pass through any additional walls accidentally these features were never activated in the first place.

6.3.3 Weapon Model

It was desired to implement a weapon model that could be basedlined against outside sources. The Sandia report [8] provided a weapon model created using exercise data, this is shown in Figure 6.3 and was implemented into STAGE through weapon objects in the unit library. The probability of kill curves are an identical implementation to those presented in the report, meaning the curves could simply be input directly. These models were then tested using a scenario to compare them to the expected probability of neutralization chart, this was presented in Table 2.1. The weapons can then be given to any unit desired. Should a different weapon model be desired changing it is a simple task and can be re-baselined using the previously mentioned scenario.

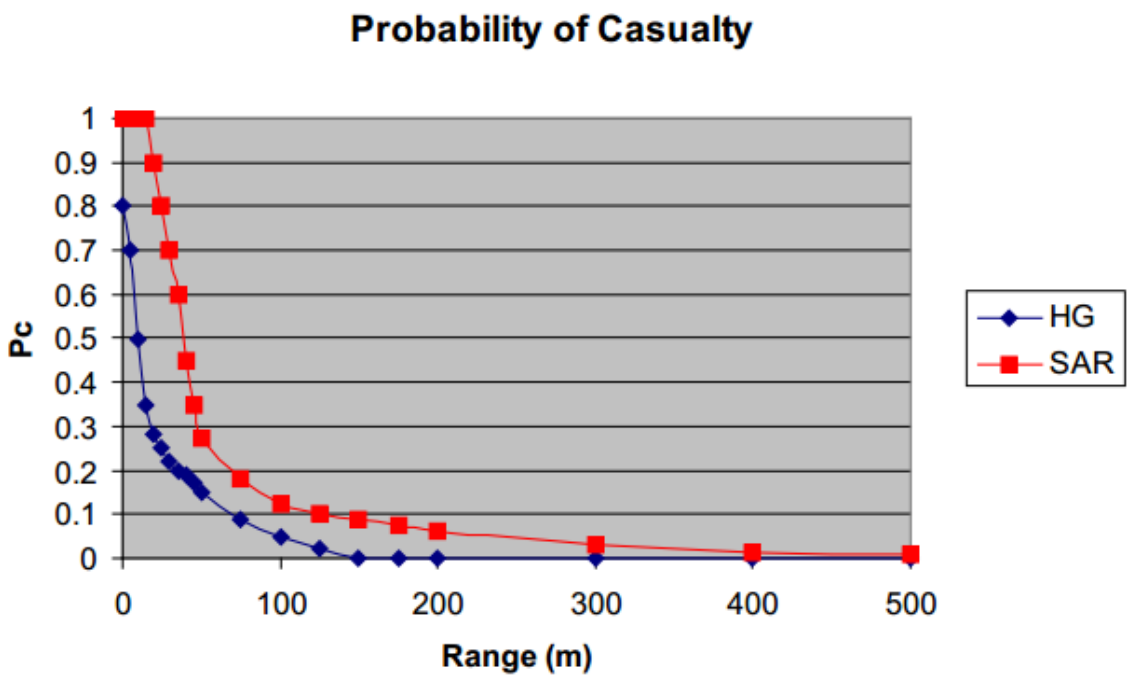

Figure 6.3: Weapon model implemented into STAGE [8].

6.3.4 Navigation

In any scenario developed there is going to be an adversary force and a response force. It would be useful to have predefined functions, tools and methods to assist developers with their behavior during scenario design. One method implemented was to simplify navigational implementation for the end user. This was done by having all groups of units form formations with a non-interacting lead unit. This means that for a group of five adversaries there is a sixth unit that can not interact with any other unit in the simulation that they follow. This lead unit has all the navigational and barrier penetration commands while the other units simply follow it. Since this unit is not target-able by the response force or detectors it will always be present to follow regardless of how many other units are defeated. It also means that from one scenario to another the other adversary unit's missions do not have to be changed, increasing the ease with which changes can be made as only one units mission must

be edited. Navigation missions have also been made more modular with the addition of reference points by grouping commonly desired actions with reference points. This makes implementing missions simpler for the end user as they simply have to pick the action they want and associate it with the correct reference point. The adversary will then approach the reference point and complete the action. All actions were created to function with this modular approach.

6.3.5 Reactions

Another important method to have available to users are reactions for both the guard force and the adversary force. It would be tedious and confusing to have to input exactly what every unit should do in a given scenario. Units should have generic reactions that are valid for any intrusion scenario were the effect of changing these reactions is not something being tested. This means that the defense force needs to sense an alarm, verify it, and dispatch the response force to the correct location. To make management easier, all alarms from the various sensors are reported to an entity representing the central alarm station. This entity then sends out generic messages such as verify alarm that can be used by the relevant entities. The message would be received by all visual sensors and patrolling guards which would attempt to see the adversary at the last detected location.

Most of the response forces behavior can be set when the facility is first implemented into STAGE and remain constant throughout various scenarios unless a change in response force behavior is a component of the scenario. This is appropriate as the guard forces behavior should be entirely reactionary to the adversary as they do not have any foreknowledge of the attack. A consistent response is useful as it means implementing a new scenario requires less time and effort to set up. Many of the modules designed assist the user in the initial set up of the guard forces such

as responding to alarms and approaching the adversary's last known location. One important module is for off site response forces. This is given to the central alarm station and it spawns the response force after sampling from a normal distribution representing how long it takes the response force to arrive on-site.

The last detected location is another method implemented using the mission script. This is a sub mission given to the lead adversary. Whenever a detection is registered a reference point is placed at the unit's current position, this is the reference point which the patrol will attempt to approach. The reference point will also be the location to which the response force will move towards. An alternate to this behavior is provided if the user desires. This alternative has the response force automatically approach the target rather than the best guess location of the adversary force. selecting the most appropriate behavior is a matter of best simulating procedure which can vary from facility to facility.

6.3.6 Modules Interactions

Outlined in the previous sections are the various modules implemented to mimic real life behavior; here how they interact will be shown to clarify how the model functions. A flow chart of this can be seen in Figure 6.4. The guard force and sensors are reactionary, for this reason the flow chart is shown from the adversary perspective. At the beginning of the simulation the adversary begins the existing module 'obstacle avoidance' as well as two created modules: 'search for opponent' and 'navigation'. These tell the agent representing the adversary to move towards its goal and be aware of their surroundings along the way.

Next the adversary will either pass through a sensors envelope of detection, or encounter a barrier it must penetrate. These will both activate one of the detection modules described in the sensors section. If a barrier is encountered this will also

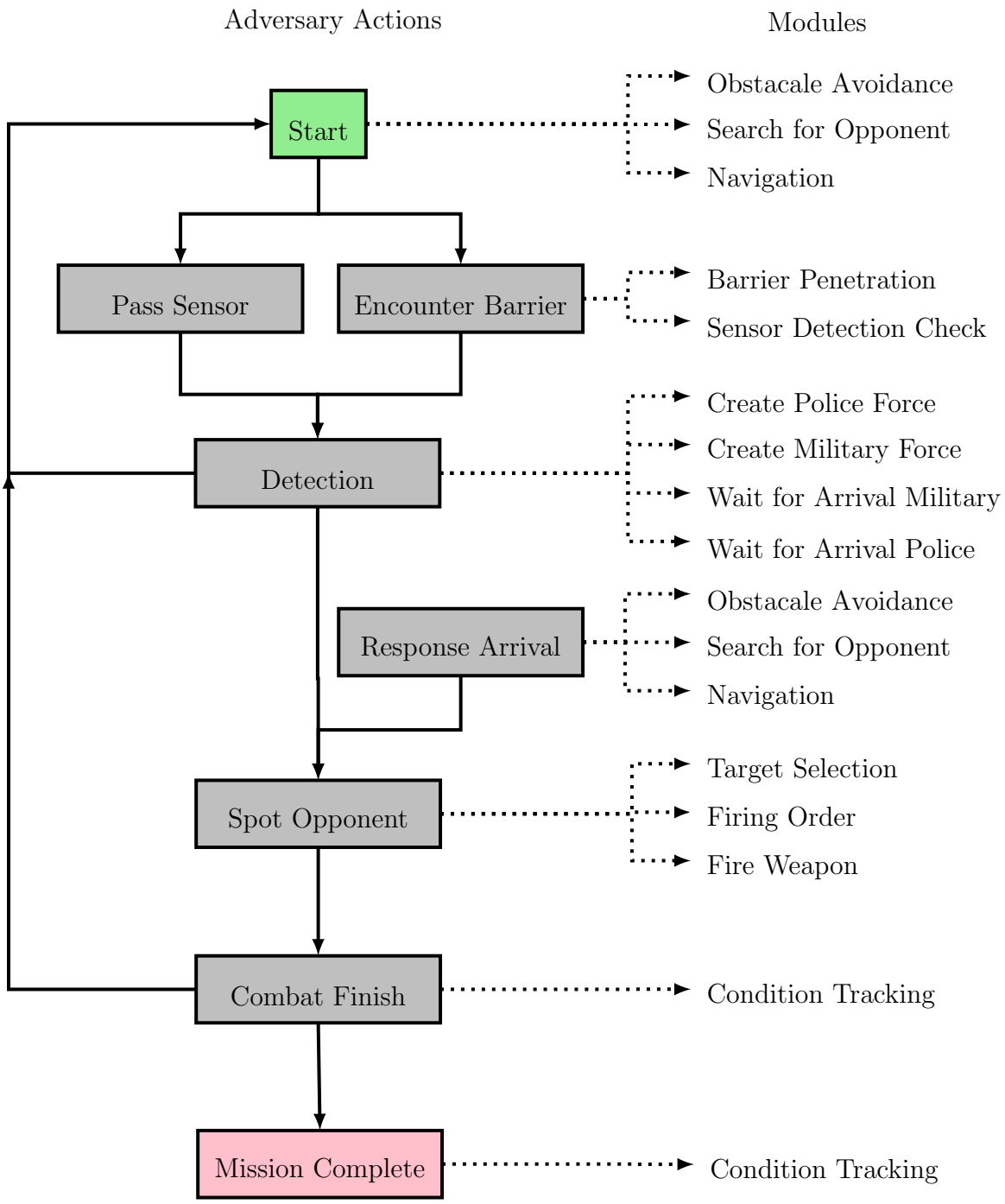

Figure 6.4: Flow chart of various modules interactions during a run.

activate the 'barrier penetration' module. This will end with one of two outcomes, detection or non-detection, seen as a loop in Figure 6.4. In either case the adversary resumes their previous behavior until they encounter another barrier or they encounter the response force. If they are detected both response forces are created and run the 'wait' module described previously to simulate arrival from off-site.

When the response force does arrive they activate the same modules to do with navigation and opponent targeting as the adversary does. It is possible for the simulation to end before the response force arrives, in this case 'spot opponent' and 'combat finish' are skipped. If one team spots a member of the opposing side the 'target selection', 'firing order', and 'fire weapon' modules are started. These are run repeatedly until combat is completed. At combat completion the 'condition tracking' module is run creating an entity that can be tracked for scoring purposes.

The adversary now either resumes the initial mission, possibly encountering another response force, or they are defeated and the mission is complete. This calls the condition tracking module again creating a different traceable entity. With this the run is completed, the Monte-Carlo code will repeat this many times and use the information created to determine the effectivness of the physical protection system portrayed. This will be outlined in the next section.

6.4 Monte-Carlo Code

6.4.1 Methodology

The previous modifications mentioned here were implemented to change how STAGE behaves during simulations and to allow the modeling of components of physical protection systems that were not present. However, it is still necessary to obtain meaningful results from this simulation. This capability is not natively present in

STAGE and it would be impractical to attempt to add it in so an external approach was taken [42]. The STAGE simulation manager can be run without the graphical interface while accepting command line arguments forming the basis for integrating an external program with STAGE [42]. To correctly determine the probability of effectiveness, enough information has to be be taken from the simulation to determine the victor of the engagement. It would also be desirable to obtain the probabilities of interruption and neutralization so additional information must be collected to determine these as well. The information from the simulation must then be parsed and the outcome determined with this information being output in an easy to read manner. This should be repeated many times in order to preform analysis through Monte-Carlo methods. Finally, this output should also have some measure of uncertainty associated with it to determine if the results are valid. This must all be made to function smoothly together to provide a simple user experience.

6.4.2 Process Control

The programing language python was chosen for doing the bulk of the processing and control based on its strong ability to parse text files as well as it easy to use nature. Python was also useful as it is an interpreted language, meaning it is compiled dynamically, allowing for the user to change variables without having to recompile to code [44]. Python was not used for the STAGE communication component of the application. As is discussed in the next section C++ code was used to output text files that the python control code read. The Monte-Carlo code starts by reading in the relevant pathways and control information that have been put in to an external file by the user. This includes the location of the STAGE directory, the key word of the scenarios to be run and the number of iterations desired for each file. The STAGE scenario directory is then parsed for any file containing the key word and these file are

put in a list to be run. Next the Monte-Carlo code must start the STAGE simulation manager as well as the ancillary programs such as the terrain manager. STAGE functions in many different parts and all but the graphical interface must be started before beginning to run a scenario.

At this point the first file to be run is opened in the STAGE simulation manager. A command file that will be read by the STAGE communication component is written containing information on what scenario to run, how long to run it for, and what random seed to use. An entry containing this information is written the number of times equal to the iteration number specified by the user. This will also be repeated for each file specified earlier with the key word. The stage simulation manager does not specify when it has completed running the scenarios. To determine when to move on to the next part of the program python monitors the STAGE manager process for cpu usage, if it is zero for over five seconds it is assumed completed and closed by python. This must be done before starting the next file in order to open a new STAGE process. If the previous instance is still open it will cause conflicts [42].

It is necessary to include a large amount of information in the files output by the communication program as STAGE does not have a method of specifying the victor within the STAGE environment. Information such as unit health at various times must be used to determine the victor. Information such as the timing of certain events may also be of interest so methods were designed to track them. This includes knowing when the adversary has successfully escaped and will be discussed later. Once various scenario runs have been completed these output files are parsed through to obtain the necessary information on the victor, the specifics of which will be discussed below. This information is used to find the probability of effectiveness of the physical protection system. Finally the standard deviation is found and output with the probability of effectiveness for the end user. An example of the code created is

available in appendix A.

6.4.3 STAGE Communication

Communication with the STAGE simulation manager in a scripted manner is not a simple task as this is not one of the primary focuses of the STAGE software [42]. For this reason rather than design this function from scratch, existing code provided by Presagis was taken and modified to suit the needs of the model [42]. This code was written in C++ and utilizes a variety of C++ libraries included in the STAGE installation. The communication code created a plug-in component to stop the simulation after a specified amount of time as well as to take readings of variables at regular intervals. The C++ code outputs all requested information for every step of the simulation, in this case every millisecond. This was unnecessary and produced large unwieldy output files so the reporting interval was reduced to once every second. The code also initially did not track unit health, only location. Unit health was the simplest indicator that can be used to determine the victor so this capability was added in. While the simulation runs this plug in periodically reads the required information and writes it to a text file, an example of which is shown in appendix C. This file is what is used to determine the outcome of the simulation.

6.4.4 Output and Validation

Once every run of a particular scenario has been completed the result must be found and conclusions drawn. This is done by reading through the output files produced and first counting the number of adversaries. To do this the program looks for certain names following a naming convention that will be laid out in the following section. Once all the names of all of the adversaries have been determined it can be found if

their damage ever reached one hundred percent. If all adversaries damage reached this point then they were defeated and this can be counted as a win for the response force. Ideally this would be enough to determine the victor, as if an adversary is left alive at the end of the simulation time then this is an adversary victory, however there are circumstances were this is not the case. If not enough time is given for the simulation or some sort of error occurs, an adversary could be left alive but not have won. For this reason when the adversaries reach their extraction point an entity named adversary-win is created. This entity essentially functions as a binary variable and is not able to interact or be interacted with by any other entity in the simulation. This entity can, however, be searched for in the output file, and if it is present this is an adversary win. The number of adversary wins is only used to calculate the number of null results which is displayed to the user to indicate something has gone wrong; generally that they need to give the simulation more time. A null result is one where neither side has won. The probability of effectiveness is then the number of response force wins over the total number of runs once the null results have been subtracted.

A similar method to determining adversary wins is used to determine both the probability of detection, probability of interruption, and probability of neutralization. When the adversary is first detected an entity named 'detected' is created. This is then searched for when parsing the output file. The number of times the entity is found over the total number of runs is the probability of detection. The probability of interruption can further be found by creating an entity when the first response force member fires their weapon. Finally the probability of neutralization can then be found by dividing the probability of effectiveness by the probability of interruption. Despite already having the probability of effectiveness, these numbers are useful as they can help diagnose were the physical protection system is failing. With some modification a similar technique can be used to count any occurrence of interest by

following the template provided.

Finally the uncertainty of these results must be determined. All of the probabilities discussed here follow a binomial distribution and thus their standard deviation can be calculated in the same manner [45]. A binomial distribution is a discreet probability distribution for the likelihood of achieving n successes in N random trials where the result can be either true or false with a probability of p [45]. This closely resembles the model constructed here, were the physical protection system was either successful or not successful. It is therefore valid to estimate the standard deviation of the probability found as that of a binomial distribution, seen in equation 6.3 [45].

$$\sigma = \sqrt[2]{\frac{p(1-p)}{n}} \tag{6.3}$$

For ease of use this can be presented as a confidence interval by multiplying the standard deviation by the appropriate percentile of a standard normal distribution [46]. This means that for 95 percent confidence that the actual value falls within the confidence interval the standard deviation must be multiplied by 1.96 as this corresponds to that many standard deviations away from the mean [45]. This requires approximating the error in a binomial distribution as that of a normal distribution. This approximation fails for probabilities close to either 0 or 1 as well as for small sample sizes. This approach is often considered valid so long as np and n(1-p) $>$ 5, if this is not the case the program will write a warning to the output [47]. The confidence interval used is supplied by the user in the input file and defaults to 95 percent. The final result given to the user is shown in the form of equation 6.4.

$$p \pm z\sqrt[2]{\frac{p(1-p)}{n}} \tag{6.4}$$

6.4.5 Usage

The elements outlined above are all behind the scenes for the end user, an outline of how they interact has been provided in Figure 6.5. From the user's point of view, implementing this model into any STAGE installation has been made as easy as possible. After placing the work files into the correct directory and compiling, the user modifies the program to their liking through a configuration file. This is where the scenario key word, run time, adversary name, iterations and confidence interval are input. This can be seen in Figure 6.5 as the first green box. A separate file is provided to change the location of where to look for directories if the install is not default. The program is then run by simply double clicking the run icon. While running, a command window will open with debug information so that the status of the simulation can be monitored.

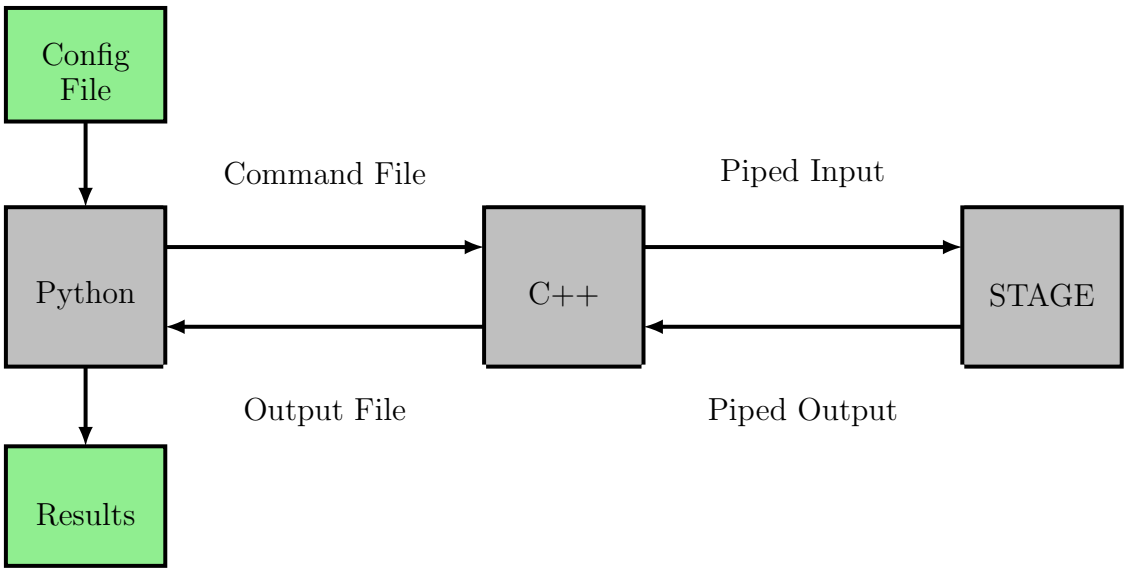

Figure 6.5: Flow chart of how the various codes interact to form the Monte-Carlo model output.

From the information provided by the user a command file is created to be read by the C++ code that interacts with STAGE. The python code then starts STAGE

allowing the C++ code to begin piping the input to the STAGE scenario manager, this is seen with the labels above the arrows in Figure 6.5. As STAGE runs the C++ code sends information as required. This includes what scenario to run, what random seed to use and when to stop. At the same time STAGE is sending information back to the C++ code, primarily the health of entities at various times. Once a run is complete this information is written into an output file that will then be read by the python code. This will be used to determinate agents health as well as if any tracking entities were created. This is repeated as many times as specified by the user.

Once complete the program will exit itself. A file labeled 'results' with a time stamp of the simulation will be created along with a folder of the output files that were parsed. The results file contains the exact path of the file run, the number of wins of both teams as well as null results, the probabilities of effectiveness, detection, interruption, and neutralization along with associated confidence intervals as well as any error messages produced. This is what the user sees once the runs have been completed with the rest of the interactions happening behind the scenes. This is the final green box labeled results seen in Figure 6.5. A example of this file can be seen in appendix D.

6.5 Scenario Development

6.5.1 3-D Model

One important part of creating a scenario for the physical protection system model not covered previously is the creation of the 3-D model that will be the basis of a scenario. This is not done using STAGE as it lacks the capability, this is instead done in an external program and imported to STAGE. Presagis offers two pieces of software that can be used to implement the buildings of the facility, Creator and

Terra Vista, although any 3-D modeling software capable of creating an open flight file can be used [42, 48]. Creator is used to create buildings that can be imported and placed on top of the terrain chosen while Terra Vista can be used to directly integrate buildings into the terrain data base. Examples of the implementation will be shown in Chapter 7.

6.5.2 Design process

Scenarios can be created in STAGE using the modules that are outlined above and placed on units organized within the scenario editor. Initial set up involves implementing the physical layout of the facility using one of the terrain databases mentioned. Next sensors are placed around the facility as close as possible to their real life positions. This is done using the pre-built sensors outlined. The properties of these sensors can easily be adjusted to have the desired ranges and detection probabilities for a given facility by modifying values in the unit library. Next the guard force is implemented using the pre-built units in the unit library. Behavior is pre-set with only a few submissions needing to be swapped around depending on desired actions such as on-site or off-site guard forces. The response force behavior remains constant for all scenarios not explicitly testing the result of modification of this behavior. Finally the actual attack is implemented. Adversaries must have the key word specified in the configuration file in their name to identify them for the Monte-Carlo code. The attack is then pieced together using methods laid out in the mission scripting language for this purpose. The name of the scenario is then placed in the configuration file for the Monte-Carlo code and the model can be run to obtain results. The design of the scenarios has been made as modular as possible to simplify the user experience.

6.5.3 Lagassi Scenario

Facility

In order to demonstrate and test the model developed, a facility had to be chosen to implement its physical protection system. This proved challenging as it is very difficult to find the specifics on many facilities due to security concerns. For this reason the Lagassi General Hypothetical Facility and PTR was chosen. This facility is a hypothetical research reactor used by the IAEA to discuss aspects of nuclear security and as a result a wide range of information is available on it allowing the modeling of a scenario with realistic layout and sensors [49]. An overhead view of the facility can be seen in Figure 6.6. Along with the physical layout of the facility, the Lagassi document was used for the probability of detection of most sensors, the material present, the facilities design basis threat, as well as for guard procedure and adversary actions [49].

The facility has a wide variety of nuclear material stored on site in the reactor building. Its fuel is $BeO - UO_2$ fuel rods with uranium enriched to 36%, with some fresh fuel and used fuel remaining on site in the fresh fuel vault and irradiated fuel pool respectively. The reactor is also used for production of radionuclides and to test experimental fuels. These are stored in the product vault and consist of 100% $^{239}PuO_2$, 95% enriched uranium and products such as cesium, americium and strontium. There is also an on-site waste storage facility that stores liquid and solid waste with trace amounts of various radionuclides [49].

The document outlines the placement of various sensors throughout the facility, primarily those associated with the reactor building. Some of the sensors present around the facility and their placement can be seen in Figure 6.7. Where the sensor placement is unclear best judgment was used to place logical sensors for a facility of

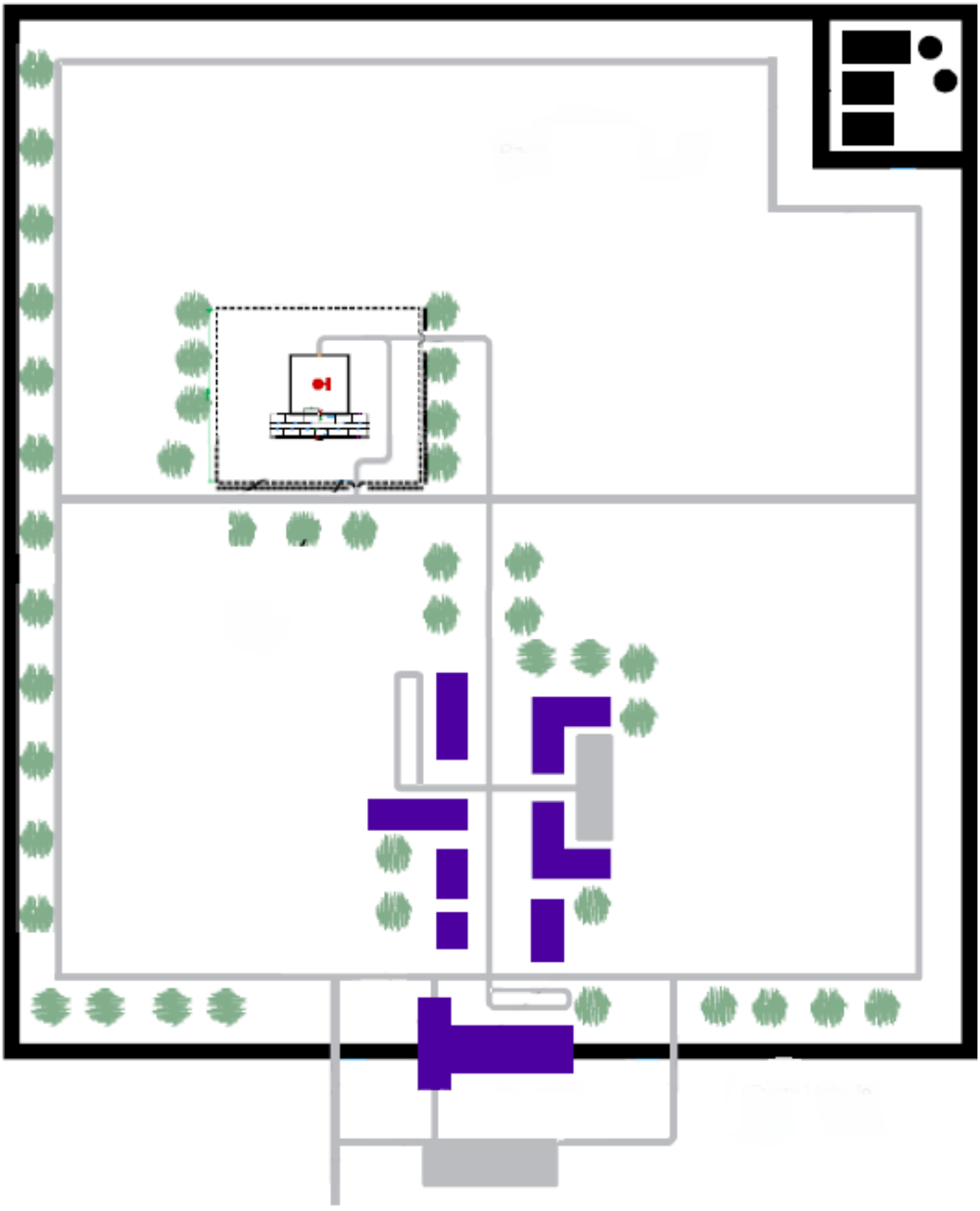

Figure 6.6: Layout of the Lagassi facility [49].

this type. Some sensors were also added for various scenarios to display modifications to the facility. The probability of detection of various sensors as well as the delay time provided by various barriers were also taken from this document [49].

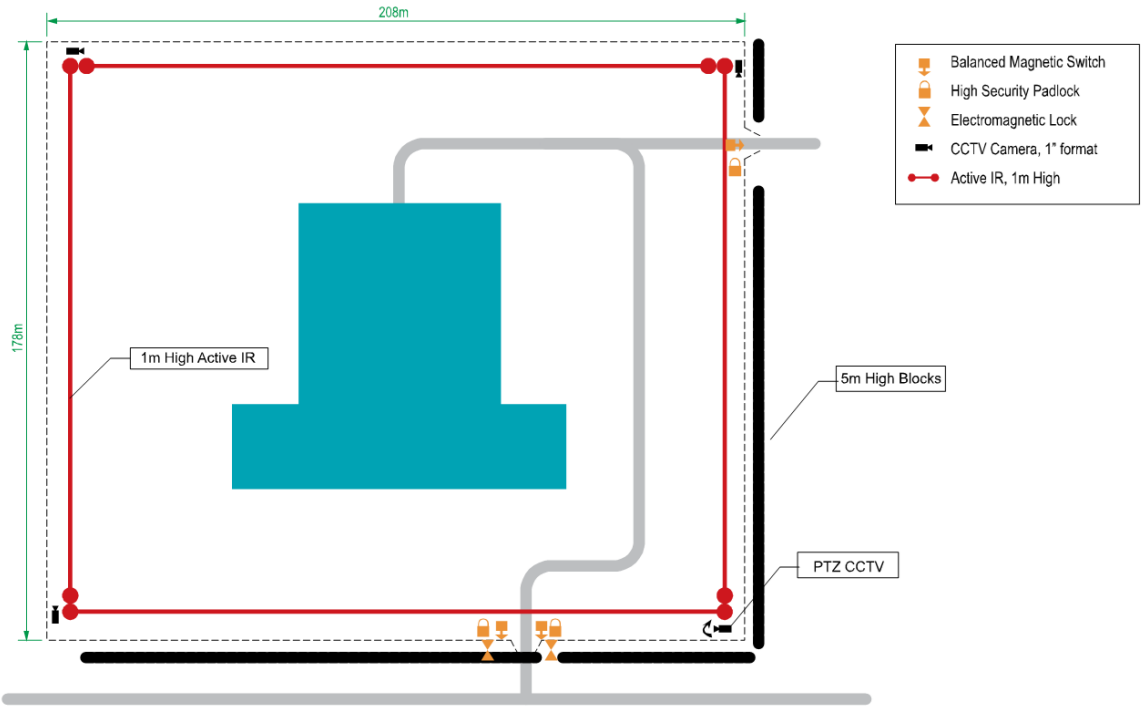

Figure 6.7: Placment of exterior sensors Lagassi [49].

Threat

Attacks on shipments of nuclear material have occurred in neighboring countries. Plans and supplies were taken from a group of political terrorists, including engineering drawings of the reactor site with circles drawn around the PTR Reactor. Also found were automatic weapons and a small amount of explosives. The group is reported as consisting of up to five people. Attempted bribes have also been reported by special forces from individuals requesting training. Local intelligence says that this is only one independent cell of a larger terrorist organization with other cells assuming to have similar composition. This terrorist organization has threatened that

they have the ability to create a radiological dispersal device [49].

For this reason the threat to the facility is assumed to be five highly motivated individuals with a high level of training equipped with assault rifles and explosives. Their goal is to steal nuclear material from the product vault for use in the construction of a radiological dispersal device. They will have inside knowledge and use it when breaching the facility [49].

Guard Force

The guard force consists of five to nine guard personnel depending on the time of day. They are distributed throughout the facility watching various key locations as well as a roving two man patrol. The guard force is armed with batons and two way radios. Response procedure dictates that on receiving an alarm the two man patrol is dispatched to verify if it cannot be verified by camera. Once the alarm is verified the guard force retreats and does not engage the adversary. At this time, a call goes out to both the police and the military who act as the response force. The police consist of two groups of two officers armed with pistols. They have a response time of approximately one hundred and fifty seconds and are the most likely to arrive first. They move to contain the adversary. The military consists of two groups of five soldiers armed with rifles who have a goal to defeat the adversary. They have an approximate response time of six hundred seconds [49].

Modeling

When the model is used for other facilities, the properties of the sensors are easy to change. The accuracy of the model is highly dependent on the accuracy of the data used to create the model. As this model is for demonstration purposes the information in the Lagassi documentation is assumed to be accurate. Weapon models were not

present in the Lagassi documentation and instead had to be taken from the Sandia labs action report [8]. Verifying the weapon models implementation is one of the scenarios discussed in Chapter 7.

Chapter 7

Simulation Results

7.1 Overview

In order to verify that the model was functioning as intended, a variety of scenarios were designed to test the modified components of the STAGE model. These scenarios were used throughout the design process to verify how the methods functioned and how they could be improved. Initial unmodified STAGE and post STAGE modification outcomes are presented below for the combat and weapon models that were two of the primary changes that required testing. These methods were used to ensure that combat was un-biased as compared to the previously biased method, and to aid in analysis using Monte-Carlo methods [25]. The procedures and methods developed can also be used to verify simpler models that are not full scale scenarios. This model was used to investigate the validity of the rule of two mentioned in Chapter 2 in a manner that, in real life, is difficult due the time and expense involved in testing. Finally a full scale model based on the Lagassi facility discussed in Chapter 6 was also modeled. This demonstrates the intended use of the model and how it can be implemented. Multiple versions of this model will be presented to showcase the

approach's ability to rapidly prototype new scenarios.

7.2 Combat Model

7.2.1 Scenario

The first model constructed was a simple scenario to ensure the combat methods were functioning correctly. STAGE's native configuration has units both being targeted and firing in alphabetical order. Default functions also mean that if two units fire simultaneously the first unit alphabetically fires first, meaning that if it scores a kill the second does not fire. The desired behavior is for target selection and firing to be random with a minimal number of units firing simultaneously to ensure as little overlap as possible. To test the new combat model a five vs five scenario consisting of opposing lines of red team and blue team was constructed. The weapons used in this test where 100 percent lethal to exaggerate the problem. Subsequently a scenario was constructed using the weapon model discussed later to view the problem in a more realistic scenario. This test was run many times with slightly different missions as the model was developed and ideas were tested. Presented here are the outcomes using initial STAGE models and again with the final combat model for each weapon. These were run using the Monte-Carlo code, and the probability of effectiveness for each case was compared to what was expected. The layout of the scenarios can be seen in Figure 7.1.

7.2.2 Results

The model was run 1,000 times using each method. The results are presented below in Table 7.1. This shows probability of neutralization using both the original STAGE

Figure 7.1: Combat model test scenario.

alphabetical combat model and the new random combat model using both 100% effective weapons and the implemented Sandia weapons. As the teams are balanced the desired value is 0.5. These results will be discussed in Chapter 8.

Table 7.1: Results of combat model testing.

		Weapon Model	
		100%	Sandia
Combat Model	STAGE	1 ± 0	0.679 ± 0.029
	New	0.512 ± 0.031	0.473 ± 0.031

7.3 Weapon Model

7.3.1 Scenario

As the weapon models where implemented from the Sandia report, seen in Figure 6.3, it was desirable to baseline this as well to ensure similar results to the source were achieved. To test this model a number of red team units were lined up opposite a number of blue team units each using the combat model developed. This was done for everything from one vs one to five vs five with all of the inbetween cases run as well, ie. two vs three, four vs five and so on. Outcomes were compared to the Sandia report neutralization chart, shown in Table 2.1. The weapon models were run using a modified version of the Monte-Carlo code that directly output the desired matrix. The layout of two vs five is seen below in Figure 7.2 [8]. This shows two blue team members in the bottom right corner armed with the assault rifles described in the Sandia report facing five red team members in the upper left hand corner armed with the same weapons. When the scenario is run they will attack one another reproducing the circumstances used to produce the neutralization charts.

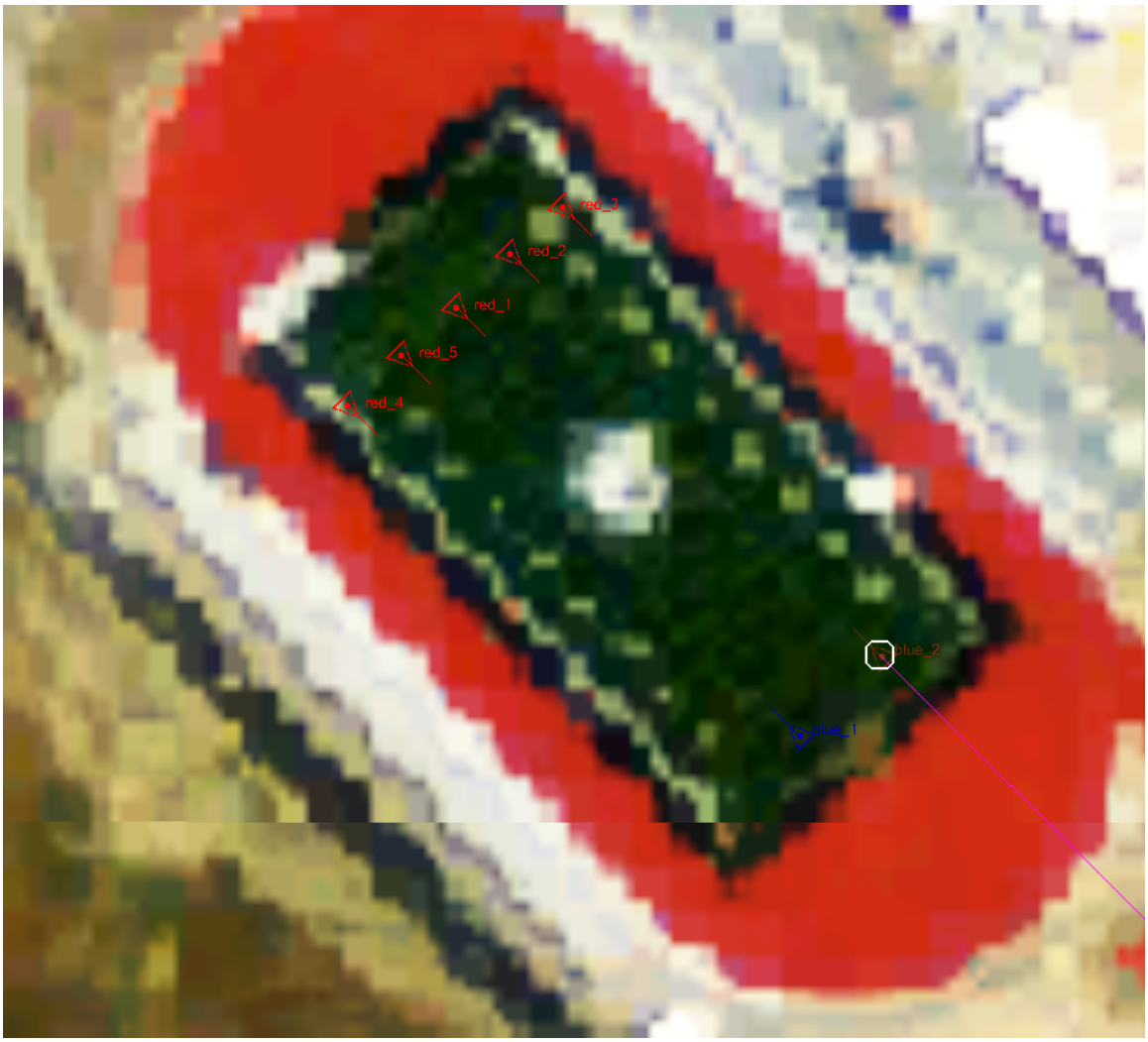

Figure 7.2: Gun test scenario.

7.3.2 Results

Each scenario was run 500 times and the probability of effectiveness found. The results are presented below in Table 7.2. It was desired to reproduce Table 2.1 as closely as possible. Some warnings were output for the probabilities of effectiveness close to zero and one. These occurred for five vs one, four vs one, and one vs five. The results will be discussed in Chapter 8.

Table 7.2: Results of weapon model.

		Adversary				
		1	2	3	4	5
Response	1	0.479 ± 0.031	0.148 ± 0.031	0.022 ± 0.018	0.012 ± 0.010	0 ± 0
	2	0.852 ± 0.022	0.504 ± 0.031	0.230 ± 0.026	0.099 ± 0.019	0.034 ± 0.011
	3	0.970 ± 0.011	0.753 ± 0.027	0.456 ± 0.031	0.256 ± 0.027	0.114 ± 0.020
	4	0.995 ± 0.004	0.927 ± 0.016	0.711 ± 0.028	0.509 ± 0.031	0.278 ± 0.028
	5	0.998 ± 0.003	0.977 ± 0.009	0.890 ± 0.019	0.709 ± 0.028	0.507 ± 0.031

7.4 Rule of Two

7.4.1 Scenario

Previous tests where done to verify functionality of the model, however these did not test the model's utility. In Chapter 2 the rule of two was mentioned as a tool for modifying probability of neutralization charts to adjust for differing weapons in simple numerical calculations [8]. The model constructed gives an opportunity to verify this rule and determine if and when it is valid. The scenario was constructed by placing red team and blue team opposite one another in a line. Blue team had five members each armed with assault rifles, red team had ten members each armed with pistols. The weapon models used were those provided by the Sandia report, which also used the rule of two. Each entity used the combat model developed. According to the rule

of two the expected result would be a probability of effectiveness of fifty percent. It was theorized that separation distance between the two sides would have a noticeable effect on the outcome so for this reason multiple scenarios were constructed varying this distance. Space between entities on the same team was minimized to reduce the difference in separation between two targets. The set up of the scenario with 50 meters of separation can be seen in Figure 7.3. These were run using the Monte-Carlo code to determine probability of effectiveness.

7.4.2 Results

The scenario was run with various separations from ten meters to two hundred meters and the probability of effectiveness for each was found. The results are presented in Figure 7.4. A linear relationship was fit to the data for illustrative purposes and will be discussed in Chapter 8. The equation of the line of best fit was found through linear regression to be $f(x) = 0.005455x - 0.055361$. The R^2 value for this fit was 0.94. This means that the expected value of 0.5 occurs at approximately one hundred meter separation. This fits with the prediction of increasing rifle effectiveness vs. distance as the pistols accuracy drops off as a function of distance more quickly, as seen in the weapon models in Figure 6.3.

7.5 Lagassi

7.5.1 Scenario

The goal of the model and many of the functions developed is to test the effectiveness of the entire physical protection systems not just the combat component shown in previously scenarios. For this reason, the Lagassi facility outlined in Chapter 6 was

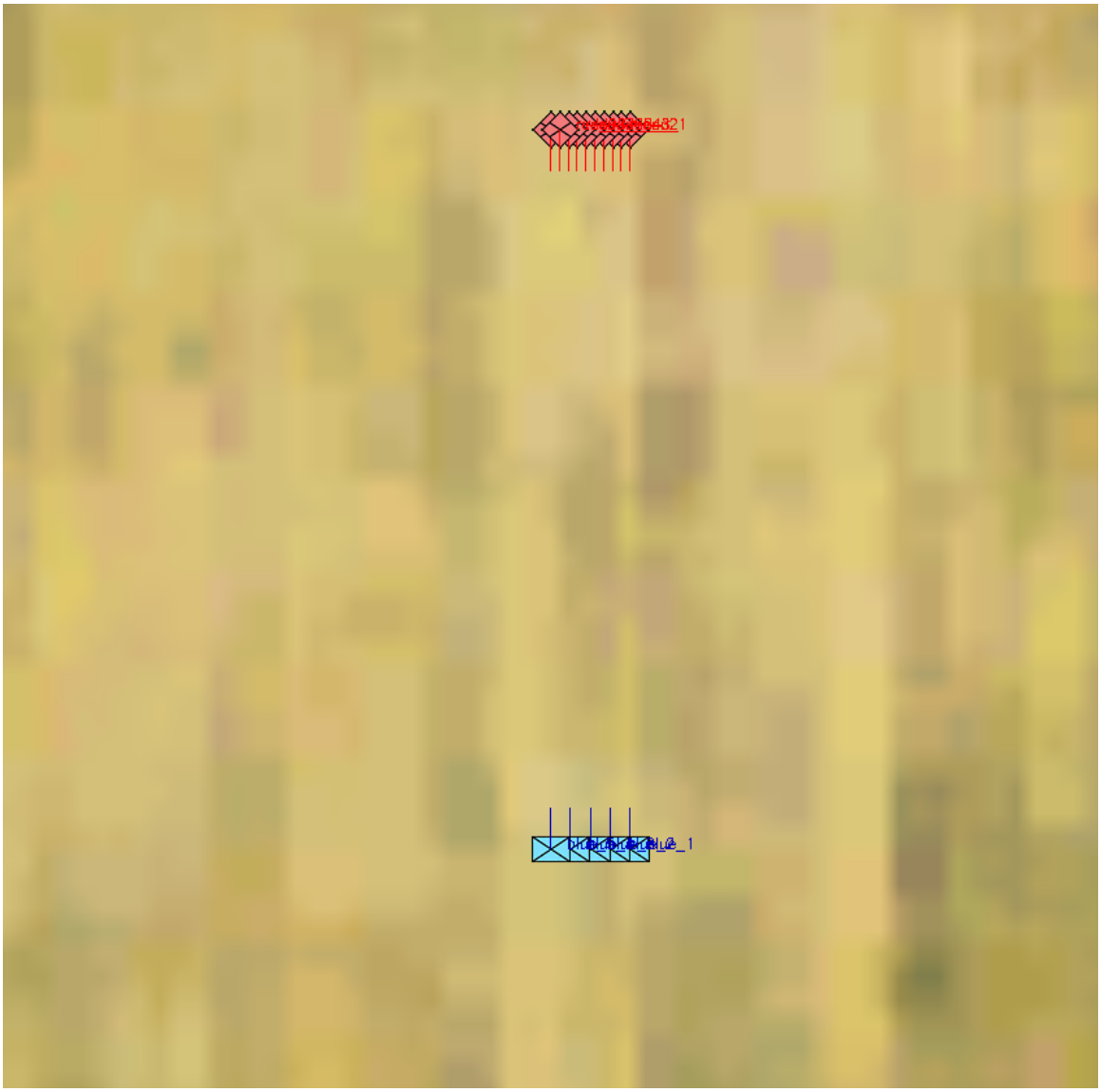

Figure 7.3: Rule of two scenario.

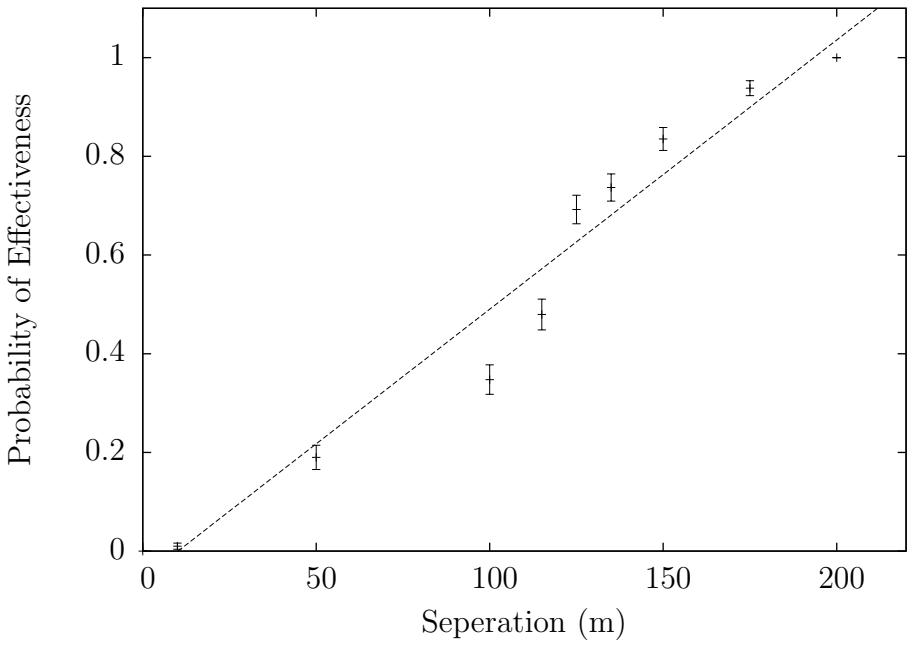

Figure 7.4: Rule of 2 results.

implemented into STAGE and an attack scenario developed. The physical layout of the facility was implemented in the 3-D modeling software Creator and can be seen in Figure 7.5. The Lagassi documentation includes the dimensions of the reactor building itself as well as the fencing immediately surrounding it, which was used to implement these into the model when possible. Other dimensions where not given and had to be implemented as best guesses using relative measurements to known facility features. This model was output as an open flight file to be placed into the STAGE scenario editor as the base for the model.

Once implemented into STAGE the various sensors described in the documentation were placed around the facility. These sensors were given properties also laid out in the Lagassi document were applicable. When lacking information, best judgment was used for sensor properties or placement. These choices were verified as best as possible and will be discussed in Chapter 8. Guards were also stationed at appro-

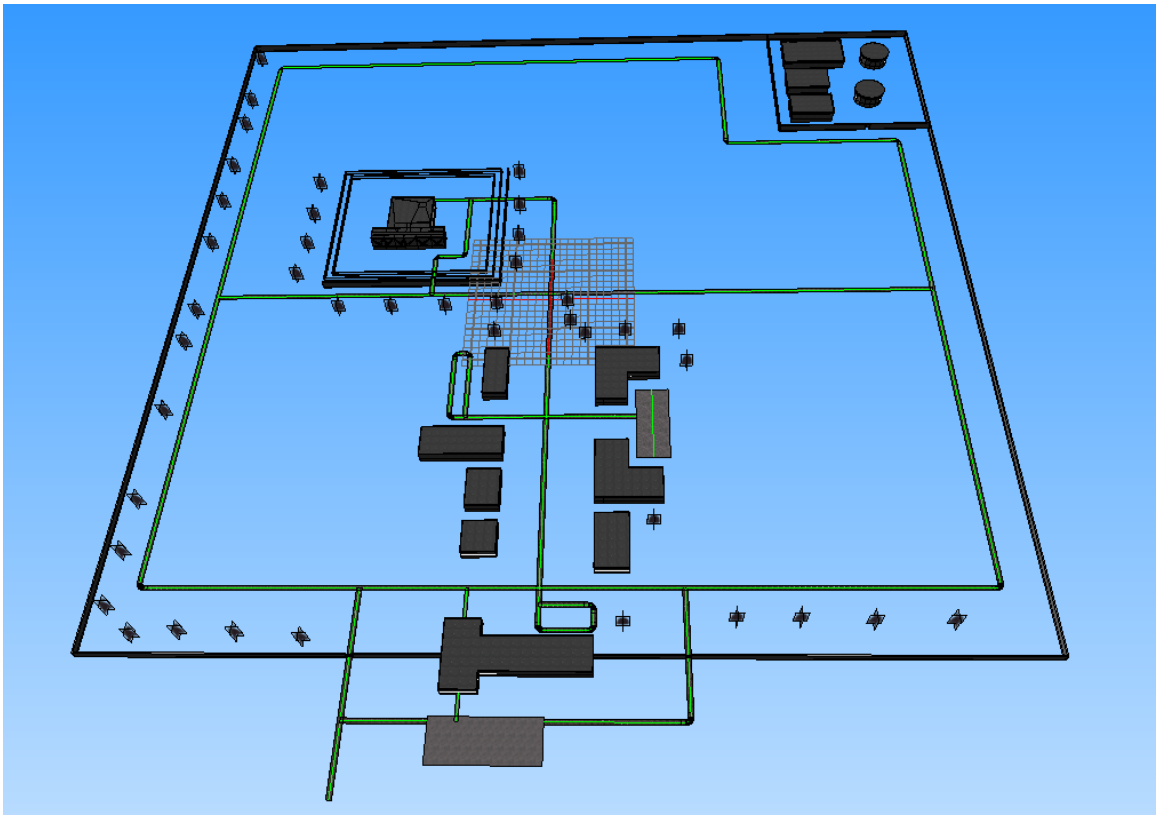

Figure 7.5: Creator model of Lagassi facility.

priate points and behavior was implemented using designed modules to follow the procedures outlined in the Lagassi documentation. As per the design basis threat the attack modeled goal was the theft of material from the isotope storage vault within the reactor building. The STAGE implementation can be seen in Figure 7.6. This scenario was set up using a variety of adversary capabilities to outline the flexibility of the system [49].

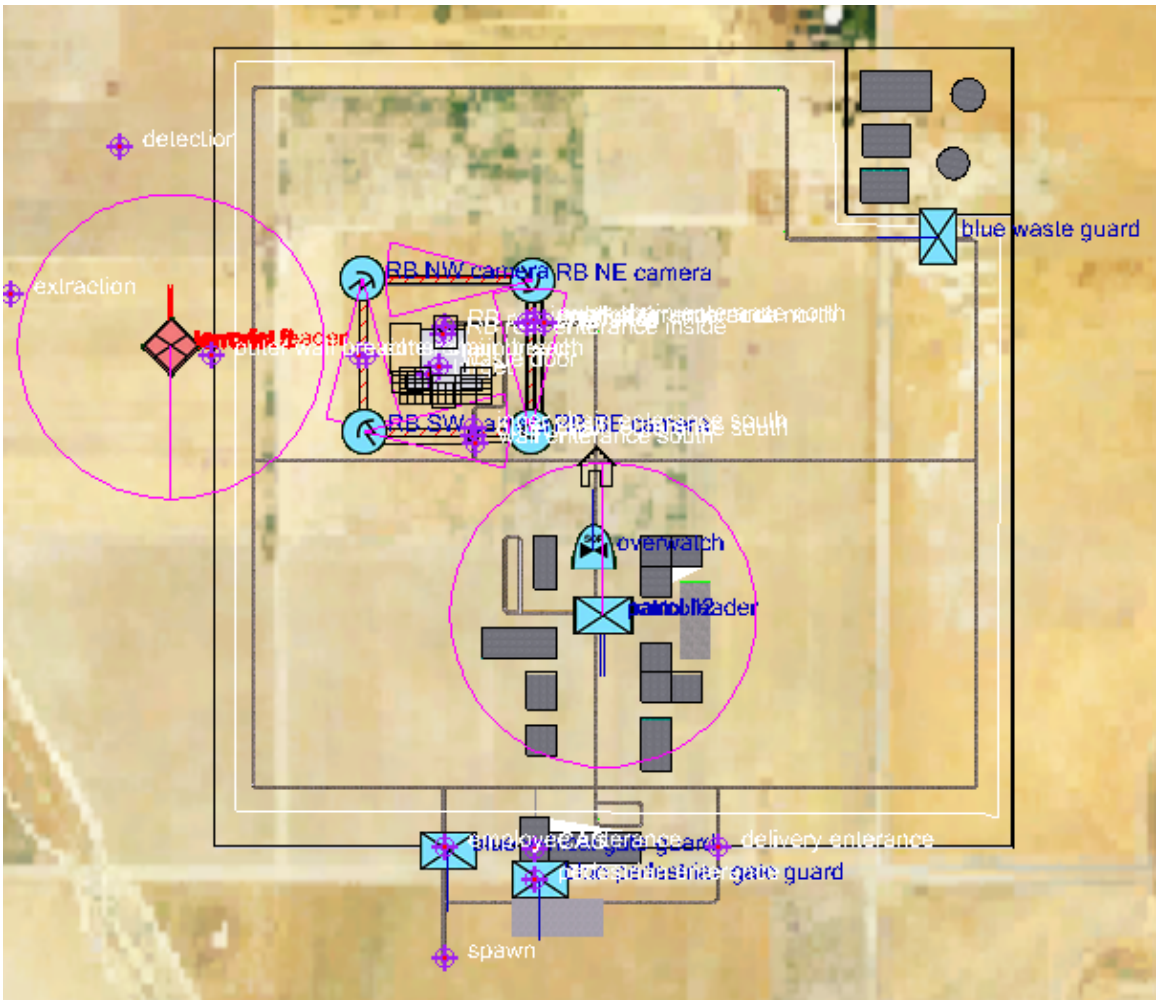

Figure 7.6: Lagassi facility implemented in STAGE.

7.5.2 Results

To illustrate how the model functions, a typical run is presented here. This scenario implemented an adversary with the fastest penetration times outlined in the Lagassi documentation for each barrier. Figure 7.7 shows the adversary's initial approach to the outer fence. The adversary attempts to breach the fence and is detected.

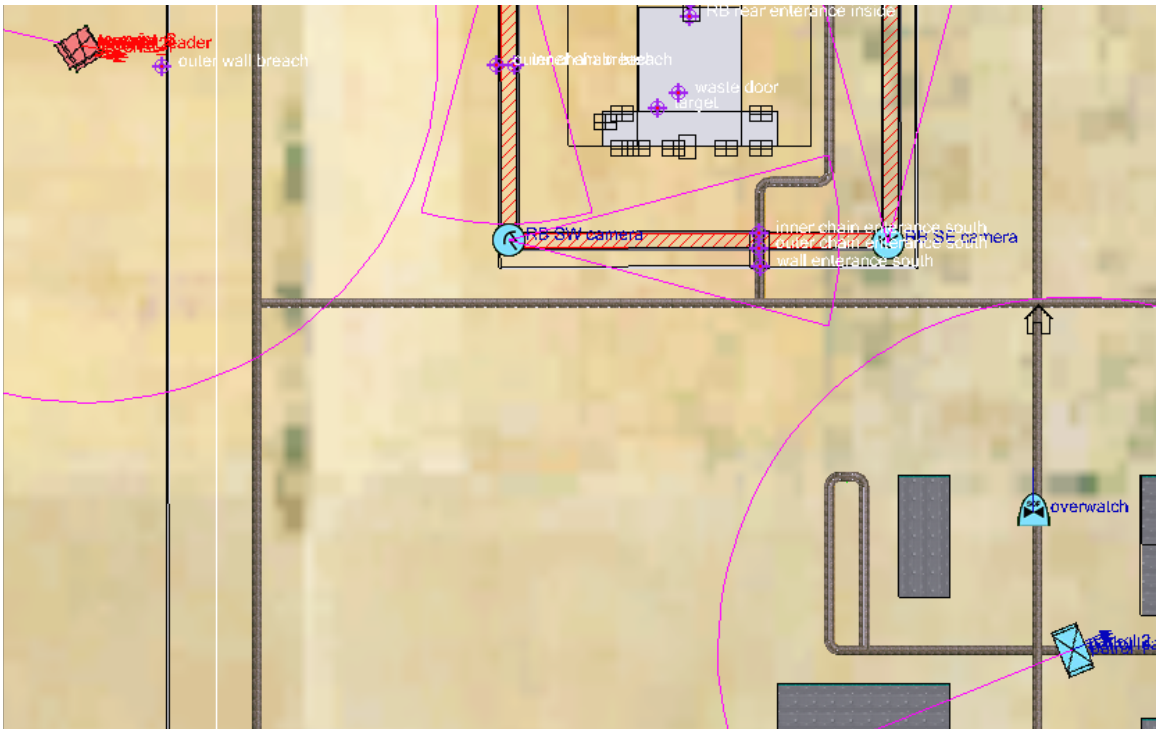

Figure 7.7: Adversary breaching outer fence.

Next the guard force dispatches the roaming patrol to verify the alarm as it is not within camera range. Figure 7.8 shows the defense force approaching the outer fence and verifying it is not a nuisance alarm. At this point they retreat and the off-site response force is called.

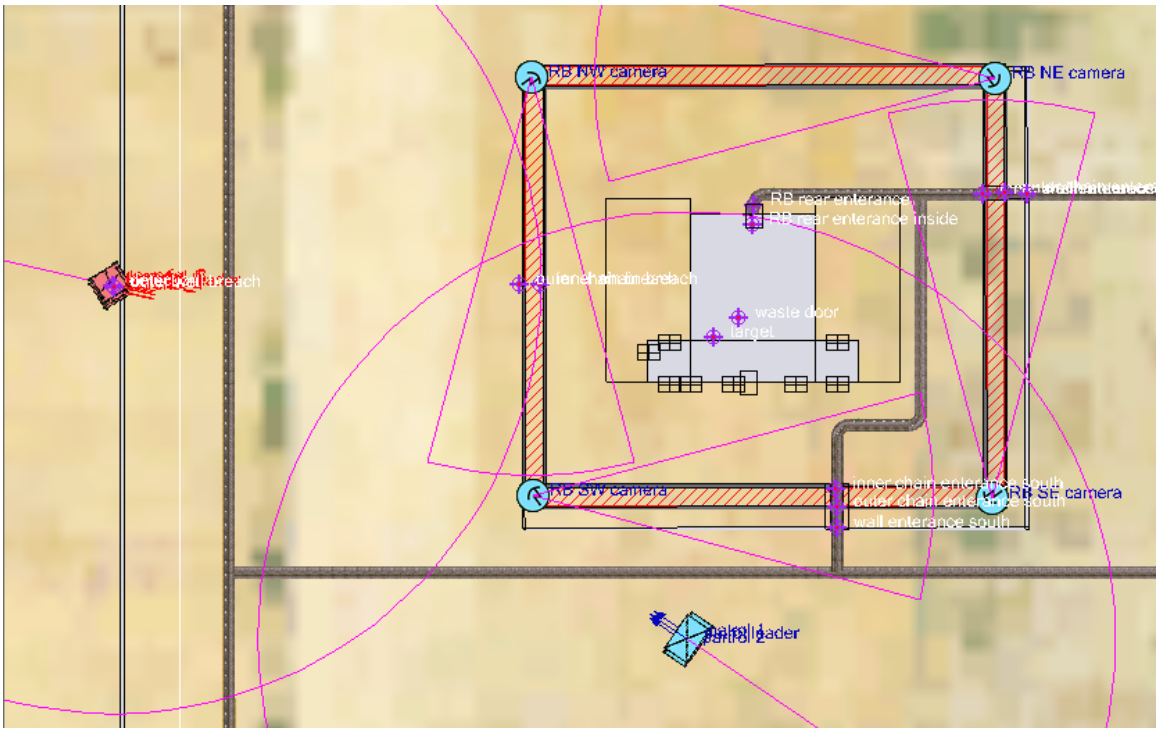

Figure 7.8: Guard force verify adversary presence.

Figure 7.9 shows the adversary having broken through the outer fence and now attempting to breach the inner fence. They are also detected at this point and their last known location is updated.

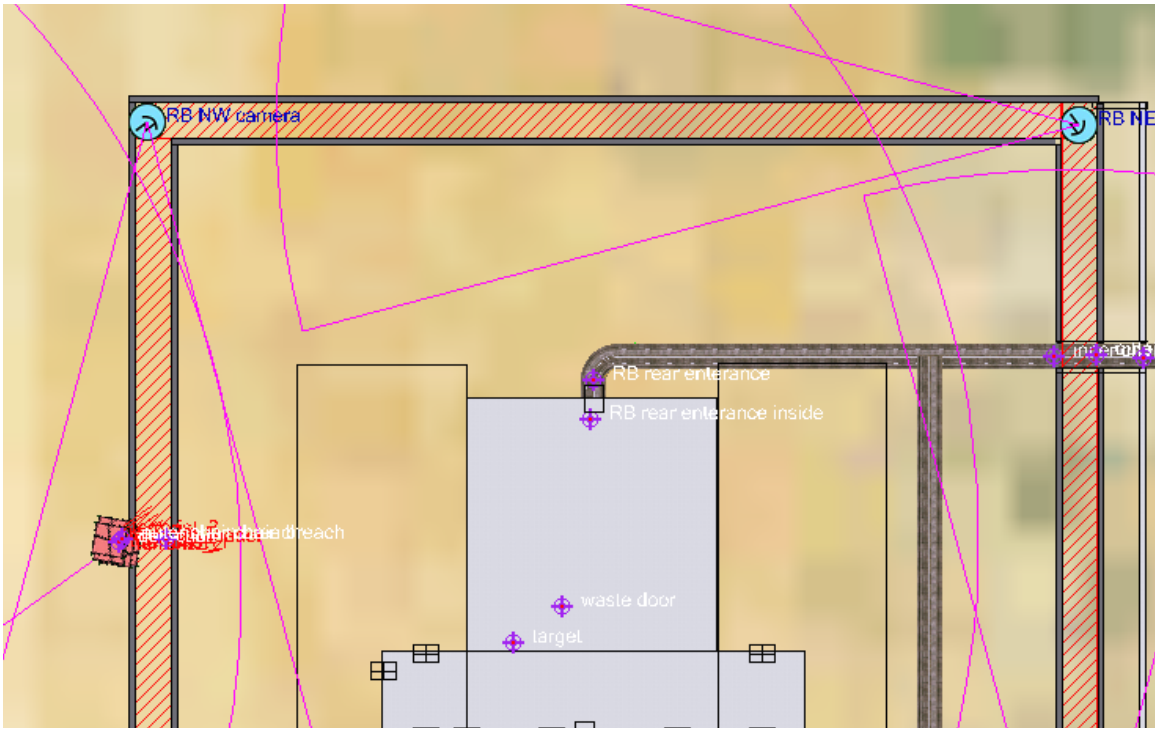

Figure 7.9: Adversary breaching inner fence.

The adversary has successfully reached the rear shipping door of the facility and has begun to breach it. The police have arrived and are on their way to attempt to stop the adversary. They will first head towards the breach in the double fence as that is the last detected point then approach the facility as that is the assumed target. This can be seen in Figure 7.10.

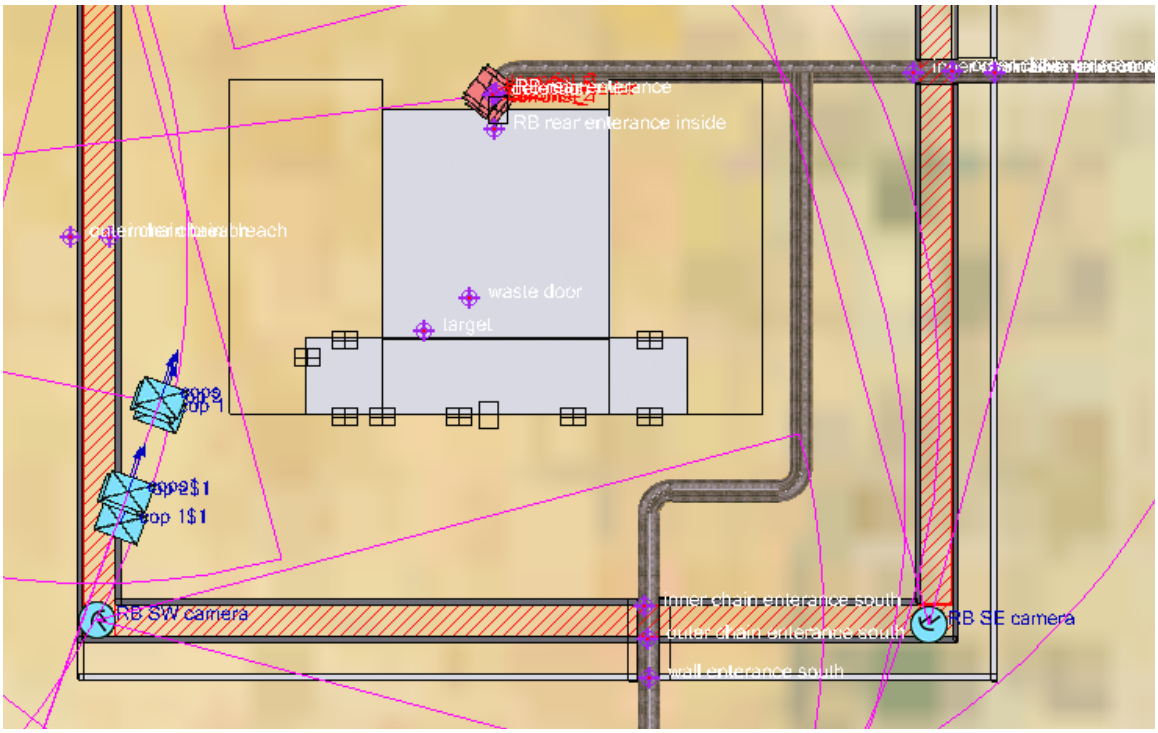

Figure 7.10: Adversary breaching rear door.

Before the adversary can breach the rear door the police response force arrive and engage the adversary. This is a blunder on the polices part as their goal is to contain the adversary not engage. However given the adversary was not detected at the rear doors the police response force had no way of knowing their location and moved to their containment position. This is a desired response as it accurately reflects the police's knowledge of the situation. The combat model is run in this case. The scenario can be seen in Figure 7.11.

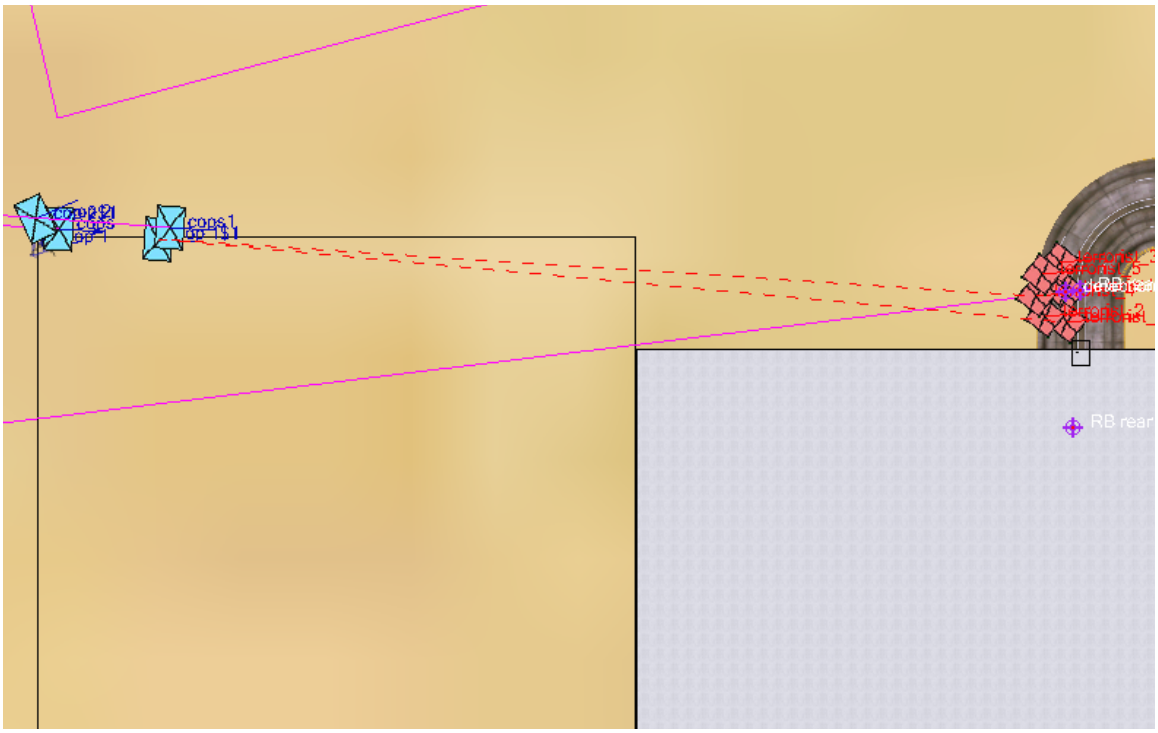

Figure 7.11: Initial engagement between adversary and police response force.

The police response force are defeated in the engagement and the adversary manages to break into the vault before the military response force arrives. Their escape from the facility can be seen in Figure 7.12. The police being defeated in this engagement is expected as the police are outnumbered and have inferior weapons. Given the long response time of the military it is also unsurprising the adversary escaped prior to their arrival when using methods with the least delay time. Images taken using STAGE's 3-D visualizer can be seen in appendix E.

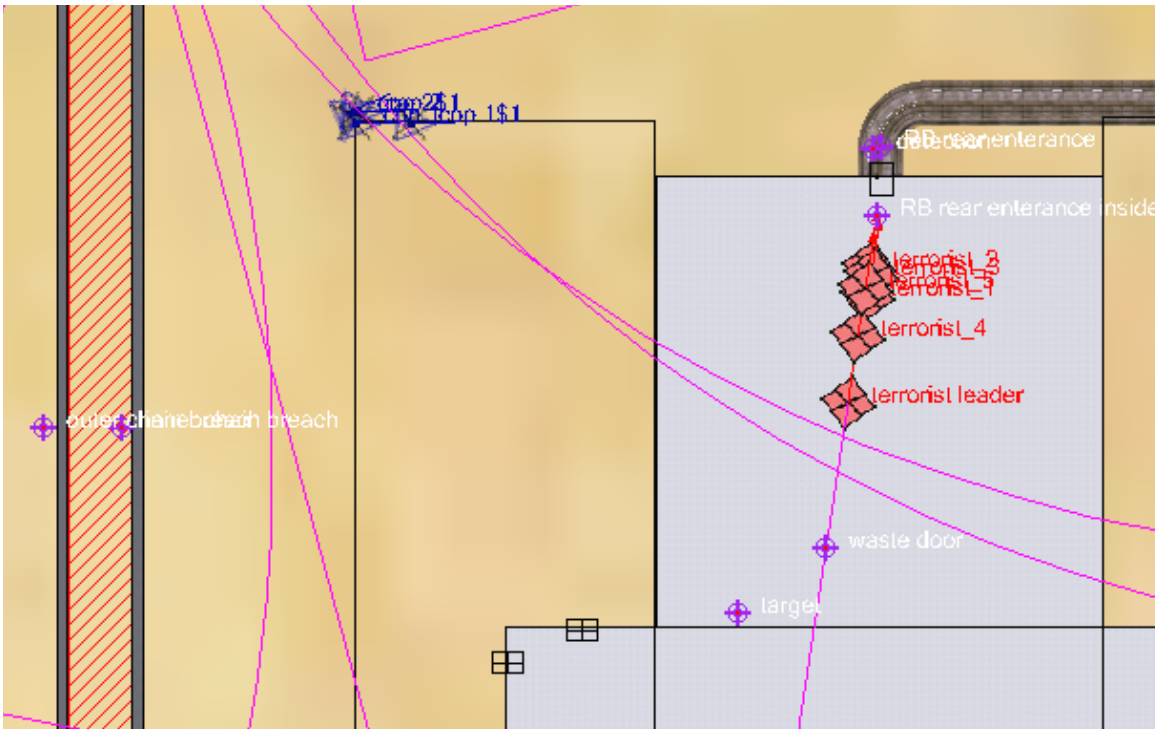

Figure 7.12: Adversary escape from facility.

The scenario described above as well as those with different adversary strategies where run 1,000 times each. The results of these are presented in Table 7.3. Scenario one is the scenario detailed above, the adversaries in this case took the fastest route to the target. This includes using explosives and power tools. These methods often had high detection probabilities. Scenario two's approach had the adversaries route taking a balance between detection probability and delay time. Finally, in scenario

three the adversary took the route through the facility which had the lowest detection probability but often resulted in the highest delay times.

Table 7.3: Results of the Lagassi simulations.

	Scenario 1	Scenario 2	Scenario 3
Probability of Detection	1 ± 0	1 ± 0	1 ± 0
Probability of Interruption	0.731 ± 0.027	0.904 ± 0.018	0.857 ± 0.022
Probability of Neutralization	0.073 ± 0.016	0.435 ± 0.031	0.830 ± 0.023
Probability of Effectiveness	0.053 ± 0.014	0.393 ± 0.031	0.712 ± 0.028

Adversary sequence diagrams were constructed as best as possible for each of these scenarios. These were used in conjunction with neutralization charts and interruption analysis tables to estimate the probability of effectiveness for each scenario using this method. Numbers used for detection and delay came from the Lagassi documentation and probability of neutralization was estimated using charts from the Sandia document used to produce the weapon models [8, 49]. An example adversary sequence diagram for scenario one can be seen in Table 7.4. The probability of neutralization was found using Figure 2.3. Results for all three scenarios can be seen in Table 7.5. These results will be discussed in the following chapter.

Table 7.4: Interruption analysis for scenario 1 using single path analysis.

	Estimate of Adversary Sequence Interruption	Probability of Guard Communication		Response Force Time(s)	
		0.95		Mean 150	STD 45
				Delays (s)	
Task	Description	P(Detection)	Location	Mean	STD
1	Approach site	0	M	24	7.2
2	breach outer fence	0.1	B	18	5.4
3	approach inner fence	0	M	18	5.4
4	breach inner fence	0.8	B	36	10.8
5	move toward facility	0	M	12	3.6
6	breach facility	0.99	B	66	19.8
7	approach vault	0	M	6	1.8
8	breach vault	0.9	B	60	18
9	escape	1	M	60	18
	P (Interruption)	0.867			

Table 7.5: Interruption analysis results.

	Scenario 1	Scenario 2	Scenario 3
Probability of Interruption	0.873	0.938	0.842
Probability of Neutralization	0.1	0.225	0.75
Probability of Effectiveness	0.087	0.211	0.632

Chapter 8

Discussion

8.1 Scenarios

8.1.1 Combat Model

The results from the weapon model tests shown in Table 7.1 depict the problem with the native STAGE combat model. With 100% lethal weapons and the same number of combatants on either side, every combat will have the same result. Every member will target the opposite team alphabetically lowest which then guarantees they kill that opponent, this continuous until only one unit is left on either side. Once this happens both units again attempt to fire with the the blue unit winning every time as they have priority in firing order due to the alphabetical order leading to a 100% probability of effectiveness. This same behavior is present when using the Sandia weapon models, however as a kill is not guaranteed the result is blue victory only 68% of the time. The separation distance between the two sides effects this result as the weapon models are less lethal the further away the target is. At closer ranges this probability will be higher and at further ranges lower until it is almost unnoticeable. This is because the lower the probability of kill of the weapon, the less pronounced

the issue is. This is because it only occurs when a member of the red team should have been able to score a kill but did not as the biased firing order did not allow them to.

This default combat model is not useful for a Monte-Carlo simulation. The desired outcome was a 50% win rate for either side when evenly matched which the default model does not produce as shown in Table 7.1. This was rectified through the development of the new combat model outlined in Chapter 6. The results using the new model for both kinds of weapons can be seen to be approximately 50% in each case. This result can be used to conclude that the new combat model is sufficient for the simulation to produce fair results representative of realistic combat. Throughout the design process this scenario was used to refine the combat model until the desired results were achieved.

A warning occurred for the STAGE model when using 100% effective weapons. This is due to the result being equal to one. This is one of the cases were the approximation of the error breaks down and the error presented for the scenario is not a good representation of the actual error. However, the error can be ignored as the intent was to show that the result is very far from 0.5, which was achieved.

8.1.2 Weapon Model

After the combat model was designed it was desirable to verify that the implemented weapons were functioning as intended. As the Lagassi documentation does not outline any weapon characteristics, weapons from the Sandia report were implemented instead [8]. The Sandia report presents a probability of neutralization chart for these provided weapon models, as shown in Figure 2.1, and will be used to baseline the weapon model. This chart was reproduced using STAGE as shown in Figure 7.2. The data from the STAGE model and from the Sandia document show good agreement

with each other. The only scenario where the Sandia documents data does not fall within the 95% confidence interval is the scenario of three vs three, and even then not by a large margin. This could be attributed to the limited number of runs done for each scenario due to time constraints. The warnings produced for five vs one and one vs five are to be expected as these are close to one and equal to zero. These errors could be resolved by increasing the number of runs as this would give more runs for the very unlikely events to occur and increasing N in the error calculation. However, the results are for demonstration purposes and are sufficient for this. Despite the warnings the good agreement is encouraging and indicates that the weapon and combat model are working well together to produce expected results.

8.1.3 Rule of Two

The rule of two offered an interesting opportunity to use the model for analysis of simpler scenarios. While the rule of two states that two individuals armed with pistols are equal to one armed with a rifle it was theorized that the separation distance between the forces would play a role. This is primarily due to pistols and rifles effectiveness as a function of distance change at different rates, this can be seen in the earlier Figure 6.3. To test this five units with assault rifles were matched against ten with pistols with the rule pf two stating the result should be 50% wins for either side. The hypothesis was that this would not be the case and distance would be a major factor. This was confirmed when the model was run with varying distances producing the chart seen in Figure 7.3. This shows that initially, when separation between the two sides is very low, pistol's have an overwhelming advantage due to their increased number of units. As separation increases the pistols advantage in numbers is slowly reduced with the break even point occurring at approximately 100 meters. Rifles continue to increase in effectiveness until 200 meters when this reaches

approximately 100%.

These results confirm the prediction that separation between the two forces is an important factor and that the rule of two is not valid at all distances. The probability of effectiveness of 50% expected from the rule of two was found to occur at approximately 100 meters. Of interest is the approximately linear relationship found. While the relationship is not perfectly linear, with the approximation over estimating for low separations and under estimating for high separations, it is useful to have. This result can be incorporated into the simple numerical models used for determining first approximations of the probability of effectiveness when the engagement distance is known. The equation found can then be used to find a better ratio to modify the probability of neutralization chart. This also indicates that when the distance of engagement is not known that the rule of two may still be useful assuming the engagement distance is known not to be very long range. This is because the approximate average effectiveness for ranges under 200 meters is the rule of two's predicted 50%.

Warnings were issued for both the 10 meters run and the 200 meters run, as is to be expected due to how close they are to zero and one respectively. The error in each case was stated to be 0, however this is due to the approximation of error no longer being valid and is not actually the case. This means that these effectiveness numbers should not be accepted to have the very high confidence they seem to have, there is still a slight chance of the opposite side winning at these points, it is just so small it did not occur during the 1,000 runs. For this reason these errors should be be ignored.

8.1.4 Lagassi

Monte-Carlo Model

The Lagassi scenario was used to implement into STAGE and a variety of attack scenarios were constructed for it. These were intended to demonstrate its functionality for the intended use of modeling an attack on a physical protection system. The results of these scenarios can be seen in Figure 7.3. The three scenarios constructed demonstrate the model's flexibility in implementing a variety of different attack scenarios, they will be discussed below.

Scenario one had the shortest total delay times but the highest detection probabilities. This means the adversary is often detected early, however delay time was often insufficient for the response force to arrive in time to prevent the theft. This gave a probability of interruption of 73.1%. Of these interruptions often only the small police force would arrive in time. The lack of time for military response also leads to the low probability of neutralization as the police force has insufficient armaments and numbers to regularly defeat the adversary. The 7.3% probability of neutralization represents the few times the military arrives in time and the occasional unlikely police victory. These combined lead to the very low 5.3% probability of effectiveness, indicating that major steps should be taken by designers to reduce the success of this kind of attack.

In the second scenario the adversary took a more balanced approach, having longer delay times but lower detection probabilities. This lead to a probability of interruption of 90.4%. This is significantly higher than scenario one as this trade off is not equal, leading to more interruptions due to the significantly longer time for penetration of the vault. Because this delay is at the end of the adversary's path it contributes heavily to the probability of interruption. This increased delay time

also allowed a higher probability of the military arriving in time as compared to just the police forces. This vastly increased the probability of neutralization to 43.5% as the military is significantly better armed than the police force. The military however does not reliably get both teams of five on site in time to intercept the adversary. As each military team has the same numbers and equipment as the adversary their arrival significantly increases probability of neutralization but does not ensure victory. The military also does not always manage to arrive in time decreasing probability of neutralization. These factors combined lead to the 39.3% probability of effectiveness, significantly greater than that seen in scenario one. This probability is however still very low and steps should be taken to improve it.

Finally scenario three has the longest delay times but also the lowest detection probabilities. This can be seen to reduce the probability of interruption to 85.7%. This is because the adversary is more likely to avoid being detected until it is too late for even the faster responding police force to intercept. There are, however, more instances where there is a large amount of delay time left after the adversary is detected, meaning the military is more often able to engage the adversary. The increased probability that both groups of military arrive makes defeating the adversary very likely, increasing the probability of neutralization to 83.0%. The adversary still occasionally manages to escape after only having defeated the police forces, somewhat lowering this probability. This results in a 71.2% probability of effectiveness for this scenario, The highest of the three scenarios. This probability however is still likely insufficient for a real life facility.

These scenarios show both the simplicity of making rapid changes to the Monte-Carlo model and the effect that various adversary strategies have on the overall effectiveness of a physical protection system. The results highlight the effect that the timing of individual events can have in relation to one another on the overall outcome

which is lost when using averaged values. It can also be seen that this approach allows the prevalence of certain events to be seen allowing the cause of a low probability of effectiveness to be diagnosed. This is a useful feature to designers as it allows for more targeted improvements to the physical protection system.

Important to note is the warnings produced for all of the probabilities of detection. The uncertainty in this case is misleading for reasons mentioned earlier however the probability of detection being this high is interesting. This means that in every run the adversary was detected at least once. This tells us that there was never a case were the adversary managed to commit the theft without getting detected, only that they managed to evade the response force. This does not mean that the adversary will always be detected as the confidence interval suggests however this does show that detection is very likely.

Adversary Sequence Diagram and Interruption Analysis Comparison

The previous scenarios were run using the Monte-Carlo model, these were compared to results found using simple numerical models involving adversary sequence diagrams and adversary neutralization charts. The adversary sequence diagrams of the simple numerical model were constructed using the same data found in the Lagassi documentation used to make the model implemented into STAGE. The results of these model were shown in Figure 7.5 and are presented in the same manor as those from the Monte-Carlo model to aid comparison. The probability of detection is not presented, this is because it is not an important factor for the simple numerical model.

For scenario one it can be seen that the adversary sequence diagram had a probability of interruption of 87.3%. This is higher than the one found by Monte-Carlo model found previously. In this circumstance, this is not surprising. Adversary sequence diagrams have only one constant response force time and do not take into

account the location of the adversary at the point of interruption. This means that the adversary sequence diagram does not account for the difference in the adversary being interrupted at a fence breach or at the facility itself. Also not taken into account is the response force not knowing the exact location of the adversary. These factors contribute most when response force arrival time is as large factor such as it is in this scenario. The probability of neutralization is also higher in the simple model for the probability of neutralization chart at 10%. This is due to the fact that the neutralization charts source relies on the rule of two. As timing was a factor, a disproportionate amount of engagements occurred while the adversary was attempting to escape. This suggests that often engagements occurred at maximum range where the police officers' pistols were less effective than the rule of two predicts. The simple method also does not factor in that the two squads of police officers likely arrive at different times often meaning there may have been occurrences were only one squad had to be defeated for the adversary to escape.

Scenario two also has a higher probability of interruption shown by the adversary sequence diagram than the Mote-Carlo model giving 93.8%; however the difference between the two methods is smaller than in scenario one. This is because of the longer delay times causing less instances were the adversary escapes due to the factors mentioned previously. The probability of neutralization shows a much larger difference with the Monte-Carlo being almost double the 25.5% found using the neutralization charts. This is due to the simple method not factoring in the spread out arrivals times present in the scenario and the delaying effect they have. This means that even though the police forces are defeated, the delay time created gives enough time for the military to arrive. The simple method also does not account for the separate groups of response force, instead assuming that all forces that will arrive in time are present at once. The overall probability of neutralization is also higher due to the

fact that the military forces are not effected by the rule of two.

Finally, Scenario three, shows the smallest difference between the probabilities of interruption between both methods however in this instance the Monte-Carlo methods result was higher. The probability found using the adversary sequence diagram was 84.2%. This smaller difference is likely due to the long delay times further reducing the reasons for the difference described in the scenario one description. In addition the splitting of the police and military forces into two groups increases the overall probability of one of them arriving as opposed to if they where one unit. This effect would be present in the other scenarios as well but the effect of decreased probability of interruption caused by accounting for interruption location was larger and masked its effect. The probability of neutralization are also larger using the Monte-Carlo method versus the 75% found using the neutralization chart method. This is again likely due to the delaying effect of the police force allowing the military to arrive as well as the military being split into two groups.

While the two models give different results the developed model never intended to exactly mirror adversary sequence diagrams. The results presented serve to highlight where these differences are between the two models. These show that the Monte-Carlo model can account for some of the weaknesses present in analysis using adversary sequence diagrams and it is hoped give a more accurate picture of an attack on a facility.

8.2 Validity of Model

8.2.1 Advantages

The scenarios presented here serve to highlight a number of advantages the developed model has over adversary sequence diagrams paired with neutralization charts and

live actions exercises.

- **Combined detection and combat models.** Unlike adversary sequence diagrams the developed model does not treat combat and detection separately. By combining the two, this allows phenomena such as part of the response force delaying the adversary while waiting for the bulk of the force to arrive as seen in the Lagassi scenario. This is difficult to account for in adversary sequence diagrams and is no extra effort in the Monte-Carlo model.

- **Separated barriers and detectors.** In an adversary sequence diagram detection probabilities are associated with barriers; in real life this is not necessarily the case. There are many detection methods that an adversary must cross that do not constitute a barrier themselves. These are difficult to include in an adversary sequence diagram. The model developed includes these in a manner as close as possible to real life.

- **More realistic sensor models.** Adversary sequence diagrams do not allow the modeling of detection as anything other than a flat detection probability. By more accurately modeling how a sensor functions to detect the adversary, the model can function closer to its real life counterpart. This allows detection to be spread out over a time period rather than simplisticly taken to be at the beginning middle or end of a delay.

- **Stronger statistical abilities.** As has been mentioned previously, by imitating live action exercises that are able to be completed much quicker, a larger sample size is feasible when determining effectiveness as compared to live action exercises.

- **Stochastic event modeling.** By modeling individual stochastic events as

opposed to averages, more control over small occurrences can have a large effect. This is shown most clearly in the arrival of the various response forces separately yielding a different result than the adversary sequence diagrams approach by simply assuming all arrive at the same time.

8.2.2 Requirements

The model's goals of a user friendly implementation of a synthetic environment for the analysis of physical protection systems was achieved. The model can be used to create a synthetic environment that is visually recognizable as the facility's physical protection system and offers a graphical manner to construct and monitor scenarios. These scenarios can be built using easy to use modules, implemented for common features such as barrier penetration and detection and easily assembled to create a wide variety of attack scenarios on a facility. The Monte-Carlo model then allows information such as the probability of effectiveness to be easily determined from the model through repeated runs of of the scenario. Through this the requirements of the model are met.

8.2.3 Uncertainty

The confidence intervals implemented ensure that the designer can have some degree of confidence in the results of the model. These are made using the assumption that the results of the model follow a binomial distribution, however the model can only be as accurate as the properties of the sensors and barriers input to it by the user and the behaviors implemented. For the purposes of this model, the example data from the Lagassi documentation was taken to be accurate. For a real life facility this may not be the case. Other statistical techniques must be used to evaluate the uncertainty

of these values and apply them to the final results [50]. These techniques are not the focus of this work so it is left for the end user to apply whatever technique they decide to account for the uncertainty of the input data.

8.2.4 Other Codes

Where possible the model developed was compared to results from other codes and approaches. This can be seen in the Lagassi scenario with the comparison to the results obtained from EASI and Neutralization. These results were reasonably close. The difference in probability of neutralization was expected as the simple model assumes the probabilities of naturalization and interruption are independent which is not always the case. The difference in probabilities of interruption are smaller and are reasonable as the new model developed accounts for location response time dependance as well as well as response force uncertainty. The model was not compared to other codes as these are either unavailable to the public or prohibitively expensive.

Chapter 9

Conclusions and Future Work

9.1 Conclusions

Based on the outcomes presented in this thesis, the need for a synthetic environment analysis tool has been demonstrated, as has the utility of the approach outlined. The model developed accounts for the dependance of the probability of neutralization on probability of interruption missed by interruption analysis methods used by EASI and other simple models. Time and location dependance of interruption that is missed by these simple models was also accounted for. The model developed is also comparable to live action exercise increasing its utility by allowing simple comparison. Unlike live action exercise the model can be run many times is a short time period allowing many more scenarios to be run with larger sample sizes. This allows designers and operators to make more educated decisions about what scenarios to focus their attention on. The model developed can be used to test scenarios in an easier way than previous methods assisting in the design itself using a user friendly graphical approach. These features and abilities allow designers to better focus their attention on scenarios that are of most threat to the facility. The information obtained on the effectiveness of the

physical protection system can not be used to license the facility, however it is still useful in narrowing down cases to focus on for licensing requirements. In conclusion this author believes that this model, or an approach like it, will eventually become a common technique used in the industry for the analysis of the effectiveness as well as the design of physical protection systems.

9.2 Future Work

There are many ways that the model presented here could be improved or made more user friendly that could not be implemented due to time constraints. Presented below are some of the possible improvements that could be implemented;

- **Increased simulation speed.** Currently, simulations must be run in almost real time due to multiple reasons. These include the iteration time, the wait time bug mentioned, and another STAGE issue that prevents it from fully utilizing all available processing power. These are all properties of the STAGE engine that will hopefully be addressed by Presagis in future updates. It may also be possible to modify this behavior using a plug in, however this is outside the plug in system's intended usage. By fixing this, scenarios could be completed more quickly, thereby increasing the utility of the model. The speed could also be increased using parallel computing methods. This requires more than one STAGE license so could not be tested, however running multiple instances of the scenario at once should allow for more runs in a shorter span of time.

- **More and improved sensors.** Those sensors present in the Lagassi facility were implemented however these are not the only kinds of sensors available. To increase the ease of use for future end users, a wider variety of sensors would

be desirable. The sensors implemented are also more involved in their set up than is desired. It would be desirable to have, for example, a microwave sensor that is a single unit were the distance between emitter and receiver is a variable set by the user rather than manually placed. This is a large departure from how sensors currently function in the STAGE environment and likely requires extensive overhaul of the unit library functions.

- **Increase user friendliness.** While efforts have been taken to make the model as user friendly as possible there are still further steps that can be taken. The Monte-Carlo code must be compiled in two separate parts using two different compilers. It would be desirable if an installer could be written to make this easier on the end user. The model is also operated through text files, a graphical user interface would help most users. Finally many variables must be looked up in a chart and set manually by the user for things such as sensors. It would be useful to have a feature such as a drop down menu within STAGE were the user can select properties from within STAGE.

- **Implement using another method.** It is difficult to verify that the model is functioning as intended as there is very little to compare it to. For this reason it would be desirable to use a similar methodology to implement the model using another force on force simulator or from scratch. This would be very time consuming but would allow for a higher degree of certainty in the results. This is desirable in the nuclear industry as it allows a higher degree of confidence in the results.

- **Open Source.** While STAGE is cheaper and more available than many other physical protection system analysis software it may still be too expensive for many small facilities and research groups. For this reason it may be desirable

to implement the methodology outlined into open source software. These environments will be less feature rich than a force on force simulator however may allow more freedom in other areas. One avenue to explore are video game engines with a large amount of modding support such as ARMA II. These have many tools available that may assist in modeling a physical protection system.

Bibliography

[1] C. Talmadge, "Deterring a nuclear 9/11," *Washington Quarterly*, vol. 30, no. 2, pp. 21–34, 2007.

[2] J. A. Blankenship, "International standard for design basis threat (dbt)," *INIS*, vol. 34, no. 10, 2002.

[3] M. L. Garcia, *Vulnerability assessment of physical protection systems*. Butterworth-Heinemann, 2005.

[4] IAEA, *Development, Use and Maintenance of the Design Basis Threat*. IAEA, 2009.

[5] M. L. Garcia, *Design and evaluation of physical protection systems*. Butterworth-Heinemann, 2007.

[6] M. Holt and A. Andrews, "Nuclear power plant security and vulnerabilities," tech. rep., Congressional Research Services, 2009.

[7] S. E. Jordan, M. K. Snell, M. M. Madsen, J. S. Smith, and B. A. Peters, "Discrete-event simulation for the design and evaluation of physical protection systems," in *Proceedings of the 30th conference on Winter simulation*, pp. 899–906, IEEE Computer Society Press, 1998.

[8] M. K. Snell, "Report on project action sheet pp05 task 3 between the us department of energy and the republic of korea ministry of education, science, and technology (mest).," tech. rep., Sandia National Laboratories (SNL-NM), Albuquerque, NM (United States), 2013.

[9] S. Kondratov and F. Steinhausler, "Why there is a need to revise the design basis threat concept," *International Journal of Nuclear Law*, vol. 1, no. 2, pp. 182–188, 2006.

[10] CNSC, "Nuclear security regulations sor/2000-209," tech. rep., CNSC, 2000.

[11] CNSC, "Security programs for category i or ii nuclear material or certain nuclear facilities g-274," tech. rep., CNSC, 2003.

[12] USNRC, "Standard format and content of a licensee physical protection plan for strategic special nuclear material at fixed sites (other than nuclear power plants) 5.52," tech. rep., USNRC, 2011.

[13] USNRC, "Standard format and content for a licensee physical security plan for the protection of special nuclear material of moderate or low strategic significance 5.59," tech. rep., USNRC, 2011.

[14] USNRC, "Physical protection programs at nuclear power reactors (sgi) 5.76," tech. rep., USNRC, 2014.

[15] USNRC, "Training and qualification of security personnel at nuclear power reactor facilities 5.75," tech. rep., USNRC, 2009.

[16] B. H. Thacker, S. W. Doebling, F. M. Hemez, M. C. Anderson, J. E. Pepin, and E. A. Rodriguez, "Concepts of model verification and validation," tech. rep., Los Alamos National Lab., Los Alamos, NM (US), 2004.

[17] M. J. Arata, *Perimeter security*. McGraw-Hill, 2006.

[18] E. Waller, "Introduction to nuclear security lecture," 2015.

[19] M. Hicks, M. Snell, J. Sandoval, and C. Potter, "Physical protection systems cost and performance analysis: a case study," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 14, no. 4, pp. 9–13, 1999.

[20] H. A. Bennett, "Easi approach to physical security evaluation," tech. rep., Sandia Labs., Albuquerque, N. Mex.(USA), 1977.

[21] J. Mirkovic, P. Reiher, C. Papadopoulos, A. Hussain, M. Shepard, M. Berg, and R. Jung, "Testing a collaborative ddos defense in a red team/blue team exercise," *Computers, IEEE Transactions on*, vol. 57, no. 8, pp. 1098–1112, 2008.

[22] N. Abdellaoui, A. Taylor, and G. Parkinson, "Comparative analysis of computer generated forces' artificial intelligence," Tech. Rep. RTO-MP-MSG-069, DRDC Ottawa, 2009.

[23] C. M. Macal and M. J. North, "Tutorial on agent-based modeling and simulation," in *Proceedings of the 37th conference on Winter simulation*, pp. 2–15, Winter Simulation Conference, 2005.

[24] E. J. Dowdy and D. L. Mangan, "Review of effectiveness-evaluation methodologies for safeguards and security systems," tech. rep., Los Alamos National Lab., NM (USA); Sandia National Labs., Albuquerque, NM (USA), 1982.

[25] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo method*. John Wiley & Sons, 2011.

[26] I. Beichl and F. Sullivan, "Monte carlo methods," *Computing in Science and Engineering*, vol. 8, no. 2, p. 7, 2006.

[27] C. L. Smith, "Understanding concepts in the defence in depth strategy," in *Security Technology, 2003. Proceedings. IEEE 37th Annual 2003 International Carnahan Conference on*, pp. 8–16, IEEE, 2003.

[28] J. Rodriguez, J. Matter, and B. Dry, "Interior intrusion detection systems," tech. rep., Nuclear Regulatory Commission, Washington, DC (United States). Div. of Safeguards and Transportation, 1991.

[29] R. L. Barnard, *Intrusion detection systems*. Gulf Professional Publishing, 1988.

[30] F. P. Martinez and F. C. Galeano, "New microwave sensors for intrusion detection systems," in *Security Technology, 1999. Proceedings. IEEE 33rd Annual 1999 International Carnahan Conference on*, pp. 49–53, IEEE, 1999.

[31] southwest, "Integrated perimeter security solutions - microwave sensors," August 2015. [Online]. Available: `http://www.southwestmicrowave.com/products/microwave-sensors/`. [Accessed:21-Aug-2015].

[32] M. Moghavvemi and L. C. Seng, "Pyroelectric infrared sensor for intruder detection," in *TENCON 2004. 2004 IEEE Region 10 Conference*, vol. 500, pp. 656–659, IEEE, 2004.

[33] Steinel, "Passive infrared (pir): Infrared systems for detecting heat radiated from the body," August 2015. [Online]. Available: `https://pirtechnology.wordpress.com/2011/09/09/hello-world/`. [Accessed:24-Aug-2015].

[34] N. Security, "Electronic security systems - active infrared sensors," August 2015. [Online]. Available: `http://www.nasatka.com/electrical-security-services/active_infared_sensors/`. [Accessed:24-Aug-2015].

[35] R. Gomery and G. Leach, "Fence vibrations [intruder detection]," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 15, no. 9, pp. 3–6, 2000.

[36] U. F. Service, "Electronic physical security tool box," August 2015. [Online]. Available: `http://www.fs.fed.us/t-d/phys_sec/alarms/ext.html`. [Accessed:28-Aug-2015].

[37] Bristorm, "Total perimeter solutions," August 2015. [Online]. Available: `http://www.bristorm.com/bristorm-temporary-products/index.html`. [Accessed:29-Aug-2015].

[38] P. K. Davis and R. H. Anderson, "Improving the composability of dod models and simulations," *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, vol. 1, no. 1, pp. 5–17, 2004.

[39] T. M. Cioppa, T. W. Lucas, and S. M. Sanchez, "Military applications of agent-based simulations," in *Simulation Conference, 2004. Proceedings of the 2004 Winter*, vol. 1, IEEE, 2004.

[40] N. A. T. ORGANISATION and R. L. SERIES, "Simulation of and for military decision making," 2003.

[41] A. Tolk, "New m&s challenges derived from the nato research & technology organization (rto) systems analysis studies (sas-071) task group on analytical tools for irregular warfare," in *Winter Simulation Conference*, pp. 2844–2851, Winter Simulation Conference, 2009.

[42] Presagis, *STAGE Documentation, User Guide*. Presagis, 2015.

[43] N. L. Johnson, S. Kotz, and N. Balakrishnan, *Continuous Multivariate Distributions, volume 1, Models and Applications*, vol. 59. New York: John Wiley & Sons, 2002.

[44] M. Lutz, *Programming python*, vol. 8. O'Reilly, 1996.

[45] I. Miller, J. E. Freund, and R. A. Johnson, *Probability and statistics for engineers*, vol. 1110. Prentice-Hall Englewood Cliffs, NJ, 1965.

[46] A. M. Pires and C. Amado, "Interval estimators for a binomial proportion: Comparison of twenty methods," *REVSTAT–Statistical Journal*, vol. 6, no. 2, pp. 165–197, 2008.

[47] L. D. Brown, T. T. Cai, and A. DasGupta, "Interval estimation for a binomial proportion," *Statistical science*, pp. 101–117, 2001.

[48] Presagis, *Creator Documentation, User Guide*. Presagis, 2014.

[49] IAEA, "Exercise data for the general hypothetical facility and ptr data," tech. rep., IAEA, 2013.

[50] M. Snell, "Collecting statistical data for security evaluations based on physical protection performance assurance and testing methodologies," in *INMM 56th Annual General Meeting*, INMM, 2015.

Appendix A

Monte-Carlo Code

Presented here is the python code used to implement Monte-Carlo methods over specified S.T.A.G.E scenario files.

```
#Runs a stage scenario many times in order to get statistics
#requires the stage monte carlo sample be compiled (found in
    stage samples)
#to properly compile must use Visual Studio 2010 SP1 in
    release mode
#need to install numpy which requires python 3.2 for 32 bit
#to compile run setup.cmd from presagis then run this line in
     same command prompt devenv filelocation /build
#changed pathway for data base in parser.cpp
#requires stage 14 x64

import time, re, subprocess, random, glob, collections, os,
    csv, numpy, shutil, sys, string
```

```
#######################################################################

#reads configfile.txt to find all of the relevant locations
    of files used
def readpathways ():
  #allows variables created here to be used in main program
  global configfile, stageloc, database, scenariopath,
      scenarioname, runtime, keyname, iterations
  pathfile = open (workdir+'\\setup\\configfile.txt','r').
      read()
  #parses the file removing all headers etc.
  temp, scenarioname, runtime, keyname, iterations = pathfile
      .split('\n')
  temp,scenarioname = scenarioname.split('= ')
  temp,runtime = runtime.split('= ')
  temp,keyname = keyname.split('= ')
  temp,iterations = iterations.split('= ')


#starts the terrain server so the config file can find the
    CDB correctly
def startterrainserver ():
  #writes a bat file that will run the server
  config = open(workdir+'\\config.bat','w')
  config.write('@echo off\n')
  config.write('call ' + stageloc + '\\setup.cmd\n')
```

```
    config.write('start /MIN /W ' + stageloc + '\\bin\\startApp
        .exe -C '+ configfile)
    config.close()
    p=subprocess.Popen(workdir+'\\config.bat', shell=False)
    p.wait()
    os.remove(workdir+'\\config.bat')

#writes file that tells stage how to run, this includes the
     number of iterations
def writecommandfile ():
    temp, name = currentscenario.split(scenarioname)
    name =scenarioname + name

    #for some reason crashes if first run does not have 5
        second delay
    file = open(stageloc + '\\samples\\monte_carlo\\stage.
        commands','w')
    for x in range (1,iterations+1):
      num = 5 if x==1 else 0
      file.write('RUN         : scenario%d\n' % x)
      file.write('Scenario    : %s\n' % name)
      file.write('Delay       : %d\n' % num)
      file.write('Seed        : %d\n' % random.randrange
          (0,1000000000))
      file.write('Stop_Cond :Stop_At_Time 00:%s\n\n' %simtime)
    file.close()
```

```
#writes a txt file that has the desired statistics
def writeoutput ():
  result = open(workdir+'\\output\\results-%s.txt' %
     starttime,'a')
  scenario = currentscenario
  #temp,scenario = currentscenario.split(scenarioname+'.') #
     seperates unique run identifier
  scenario,temp = scenario.split(r'.')
  result.write('%s\n' %scenario)
  result.write('red win =(%d)\n' %redwin)
  result.write('blue win =(%d)\n' %bluewin)
  result.write('tie = %d\n' %tie)
  result.write('blue win ratio=%f\n\n' %(bluewin/float(redwin
     +bluewin)))
  result.close()

#converts txt to csv
def writecsv ():
  chart = numpy.empty((maxnum+1,maxnum+1))
  chart[:] = numpy.NAN
  csvfile = open(workdir+'\\output\\resultscsv-%s.csv' %
     starttime,'w')
  result = open(workdir+'\\output\\results-%s.txt' %
     starttime,'r').read()
  result = result.split('\n\n')
```

```
    #set up x and y axis
    for x in range (0, (maxnum+1)):
        chart[0][x]=x
        chart[x][0]=x

    #write output file
    for pos in range (0,len(result)-1):
        blue , temp = result[pos].split('v')
        #red , temp = temp.split('\n') for when specific win
            stats are suppressed
        ################## for when wins not suppresed
        red , temp1, temp2,temp3,temp = temp.split('\n')
        red,temp1 = red.split(',') #remove
        ################
        temp , win = temp.split('=')
        chart[int(blue)][int(red)]=100-(float(win)*100)
        chart[int(red)][int(blue)]=float(win)*100
    numpy.savetxt(csvfile, chart)

############################################################

workdir = os.path.dirname(os.path.realpath(__file__))
stageloc = 'C:\\Presagis\\Suite14\\STAGE'
configfile = 'C:\\Presagis\\Suite14\\STAGE\\samples\\
    monte_carlo\\config.cfg'
```

```
database = 'C:\\Presagis\\Suite14\\STAGE\\samples\\
    monte_carlo\\Lagassi.xml'
scenariopath = 'C:\\Presagis\\Suite14\\STAGE\\samples\\
    monte_carlo\\Lagassi\\scenario\\'
starttime = time.strftime("%Y_%m_%d-%H_%M_%S")

#clean out old files
shutil.rmtree(workdir+'\\Monte_carlo_data')
os.makedirs(workdir+'\\Monte_carlo_data')

#sets up run
readpathways()
startterrainserver ()

#find all files to be run
filelist=glob.glob(scenariopath + scenarioname+'*.scenario')
filelist.reverse()
iterations=int(iterations)
#######################################################################
#runs the number of scenarios given
for x in range (0 , len(filelist)):
   currentscenario = filelist.pop()
   redwin = bluewin = tie = 0
   runtime =int(runtime)
   simtime= '0%d:%d'%((runtime-runtime % 60)/60,runtime % 60)
```

```
#runs the command file
writecommandfile ()
p=subprocess.Popen('' + stageloc + '\\samples\\monte_carlo
    \\start_batch_cmd.cmd', shell=False)

time.sleep(10)
while True:
  if any('StageBatchCmd' in s for s in os.popen('tasklist')
      .read().split()):
    time.sleep (5)
  else:
    temp, process =os.popen('tasklist').read().split('
        stageSIM.exe')
    process = process.partition('Console')[0]
    os.kill(int(process),-1)
    break
print('done simulations')
movefiles= glob.glob(workdir+'\\out_data*')
for y in range (0, len(movefiles)):
  currentfile=movefiles.pop()
  temp,name=currentfile.split('out')
  name='out'+name
  shutil.move(currentfile,workdir+'\\Monte_carlo_data\\'+
      name)

#after iterations done finds winner and reorganises files
```

```
outfiles= glob.glob(workdir+'\\Monte_carlo_data\\out_data
   *')
for y in range (0, len(outfiles)):
  currentfile=outfiles.pop()

  tempnames = set()
  with open(currentfile) as infile:
    for line in infile:
      templine=line.split()
      for s in range (0, len(templine)):
        if (keyname in templine[s]): tempnames.add(templine
           [s])
  tempnames.discard(keyname)
  keynames =[]
  for s in range (0,len(tempnames)):keynames.append(
     tempnames.pop()+' 100.000000')

  #check for key words to determine victor
  dead = 0
  with open(currentfile) as infile:
    for line in infile:
      for n in range (0, len(keynames)):
        if (keynames[n] in line):
          keynames[n] = 'gwsegwegloijoszszh'
          dead += 1
```

```
        if dead==len(keynames):
            bluewin +=1
        else:
            redwin +=1

        #moves output files
        destination,temp=currentfile.split('out_data')
        if not os.path.exists(destination + 'run
        destination= destination + 'run
        os.rename(currentfile, destination)
        print('done reading file')
    writeoutput ()

#######################################################################

#appends the the run finish time onto the result file
result = open(workdir+'\\output\\results-%s.txt' % starttime
    ,'a')
result.write('end time '+time.strftime("%Y_%m_%d-%H_%M_%S"))
result.close()
print('done')

#writecsv()
```

Appendix B

Sample STAGE Run Debug Output

Presented here is output of a typical run. This output is intended to give an idea of what occurred in each run and can be used to spot errors in the run. Its intended purpose is debugging and can be modified in the STAGE environment to output any information of interest. This information is not used to find results, that is done with the results file presented in appendix C.

```
Sim running in Stand-Alone mode
Ground Navigation Service initialized
Maximum Number of Entities = 200
Using TSP Terrain Service.
Stage Batch Sim Plugin Initialized
simScenarioLoader: Loading Database C:\Presagis\Suite14\STAGE\
samples\monte_carlo\Lagassi.xml
simScenarioLoader: Loading scenario C:\Presagis\Suite14\STAGE\
samples\monte_carlo\Lagassi\scenario\night_attack.scenario
```

```
Simulation Scenario Manager: Waiting for load sequence to complete.
Click Ignore to start scenario before background info is ready.
Progress = 0
Progress updated to = 1
Progress updated to = 16
Progress updated to = 17
Progress updated to = 31
Progress updated to = 50
Progress updated to = 63
Progress updated to = 80
Progress updated to = 88
Progress updated to = 94
Progress updated to = 100
Simulation Scenario Manager: Load has completed.
Progress = 100
Parsing SIM configuration file...
Starting run #1, iteration 0 with seed 459786530
Stop conditions: after 1200.000000 seconds,
sim_rtc: Scenario night_attack running
Lagassi night defense
outer wall pen time 208.449963778257
detected at outer wall
verifing detection
verified
through
outer chain pen time 75.8171335458756
```

```
Mission Start (cops): can not find entity (cop 1$1)
Mission Start (cops): can not find entity (cop 2$1)
cops arrive
form up cop 1
form up cop 2
cops arrive
form up cop 1$1
form up cop 2$1
through
inner chain pen time 37.0323807746171
detected at fence gap
detected at fence gap
detected at fence gap
detected at fence gap
detected at fence gap
detected at fence gap
through
detected at fence gap
back door pen time 157.07388676703
saw terrorist_3
stopping
fire  cop 1
fire  cop 2
fire  cop 1
fire  cop 2
dead  terrorist_3
```

```
fire  cop 1

fire  cop 2

saw cop 2$1

stopping

saw terrorist_2

stopping

fire  cop 2

fire  cop 1$1

fire  cop 1

fire  cop 1

fire  cop 1$1

fire  terrorist_4

fire  terrorist_5

dead  cop 2$1

fire  terrorist_2

fire  cop 2

fire  cop 1

fire  terrorist_4

red all clear, move

dead  cop 1$1

fire  cop 2

fire  cop 1

fire  cop 2

fire  cop 1

fire  cop 1

fire  cop 2
```

```
blue all clear
dead  terrorist_5
fire  cop 1
through
detected at rear door
Cannot assign procedure Enter Building to entity terrorist_5:
The entity cannot do ground navigation.
Cannot assign procedure Enter Building to entity terrorist_3:
The entity cannot do ground navigation.
Cannot assign procedure Goto Target to entity terrorist_5:
The entity cannot do ground navigation.
Cannot assign procedure Goto Target to entity terrorist_3:
The entity cannot do ground navigation.
blue all clear
waste door pen time 28.4571060240269
through
detected at waste door
Cannot assign procedure Goto Target to entity terrorist_3:
The entity cannot do ground navigation.
Cannot assign procedure Goto Target to entity terrorist_5:
The entity cannot do ground navigation.
11.7582933872938
Cannot assign procedure Goto Target to entity terrorist_3:
The entity cannot do ground navigation.
Cannot assign procedure Goto Target to entity terrorist_5:
The entity cannot do ground navigation.
```

```
leaving
leaving
Cannot assign procedure Exit Building to entity terrorist_3:
The entity cannot do ground navigation.
leaving
leaving
Cannot assign procedure Exit Building to entity terrorist_5:
The entity cannot do ground navigation.
leaving
ready
saw cop 1
stopping
fire  terrorist_2
dead  cop 1
fire  terrorist_4
fire  terrorist_2
red all clear
dead  cop 2
Mission Start (army): can not find entity (soldier 2$1)
Mission Start (army): can not find entity (soldier 3$1)
Mission Start (army): can not find entity (soldier 4$1)
Mission Start (army): can not find entity (soldier 5$1)
Mission Start (army): can not find entity (soldier 1$1)
army arrives
form up soldier 2
form up soldier 3
```

```
form up soldier 4
form up soldier 5
form up soldier 1
army arrives
form up soldier 2$1
form up soldier 3$1
form up soldier 4$1
form up soldier 5$1
form up soldier 1$1
red win
```

Appendix C

Sample STAGE Run Output

Presented here is a sample of the raw output from the STAGE communication code. This is parsed by the Monte-Carlo code to determine the outcome of the scenario. Each line presents in order: time, entity name, damage, speed, altitude, ground level, and heading. The output here is a sample for a small amount of time in one scenario, each scenario outputs approximately 1 Gb of output per run. in conjunction with properly set output from appendix B the occurrence of any event of interest can be found.

```
499.999000 RB NE camera 0.000000 0.000000 433.029358
   433.029358 -1.570796
499.999000 RB NW camera 0.000000 0.000000 434.589600
   434.589600 -3.141593
499.999000 RB SE camera 0.000000 0.000000 431.927795
   431.927795 0.000000
499.999000 RB SW camera 0.000000 0.000000 431.718567
   431.718567 1.570796
```

499.999000 blue pedestrian gate guard 0.000000 0.000000
435.104584 435.104584 3.139486
499.999000 blue vehical gate guard 0.000000 0.000000
433.161438 433.161438 3.139494
499.999000 blue waste guard 0.000000 0.000000 433.753662
433.753662 −1.570796
499.999000 overwatch 0.000000 0.000000 4000.000000 431.476868
0.000000
499.999000 partrol 2 0.000000 0.000000 431.492371 431.492371
−1.062041
499.999000 patrol 1 0.000000 0.000000 431.594330 431.594330
−1.062041
499.999000 patrol leader 0.000000 0.000000 431.547913
431.547913 −1.062041
499.999000 terrorist leader 0.000000 0.000000 432.563782
432.563782 1.447663
499.999000 terrorist_1 0.000000 0.000000 432.537231
432.537231 1.447663
499.999000 terrorist_2 0.000000 0.000000 432.603760
432.603760 1.447663
499.999000 terrorist_3 0.000000 0.000000 432.635590
432.635590 1.447663
499.999000 terrorist_4 0.000000 0.000000 432.649658
432.649658 1.447727
499.999000 terrorist_5 0.000000 0.000000 432.671692
432.671692 1.445142

499.999000 cops 0.000000 5.000000 432.450531 432.450531
0.342335

499.999000 cop 1 0.000000 6.500000 432.372620 432.372620
0.342637

499.999000 cop 2 0.000000 4.944432 432.403717 432.403717
0.342334

499.999000 cops1 0.000000 5.000000 432.567596 432.567596
0.342339

499.999000 cop 1$1 0.000000 6.500000 432.350830 432.350830
0.342684

499.999000 cop 2$1 0.000000 4.983599 432.481750 432.481750
0.342339

500.000000 RB NE camera 0.000000 0.000000 433.029358
433.029358 −1.570796

500.000000 RB NW camera 0.000000 0.000000 434.589600
434.589600 −3.141593

500.000000 RB SE camera 0.000000 0.000000 431.927795
431.927795 0.000000

500.000000 RB SW camera 0.000000 0.000000 431.718567
431.718567 1.570796

500.000000 blue pedestrian gate guard 0.000000 0.000000
435.104584 435.104584 3.139486

500.000000 blue vehical gate guard 0.000000 0.000000
433.161438 433.161438 3.139494

500.000000 blue waste guard 0.000000 0.000000 433.753662
433.753662 −1.570796

500.000000 overwatch 0.000000 0.000000 4000.000000 431.476868
0.000000

500.000000 partrol 2 0.000000 0.000000 431.492371 431.492371
−1.062041

500.000000 patrol 1 0.000000 0.000000 431.594330 431.594330
−1.062041

500.000000 patrol leader 0.000000 0.000000 431.547913
431.547913 −1.062041

500.000000 terrorist leader 0.000000 0.000000 432.563782
432.563782 1.447663

500.000000 terrorist_1 0.000000 0.000000 432.537231
432.537231 1.447663

500.000000 terrorist_2 0.000000 0.000000 432.603760
432.603760 1.447663

500.000000 terrorist_3 0.000000 0.000000 432.635590
432.635590 1.447663

500.000000 terrorist_4 0.000000 0.000000 432.649658
432.649658 1.447727

500.000000 terrorist_5 0.000000 0.000000 432.671692
432.671692 1.445142

500.000000 cops 0.000000 5.000000 432.450531 432.450531
0.342335

500.000000 cop 1 0.000000 6.500000 432.372620 432.372620
0.342637

500.000000 cop 2 0.000000 4.947232 432.403717 432.403717
0.342334

```
500.000000 cops1 0.000000 5.000000 432.567596 432.567596
   0.342339
500.000000 cop 1$1 0.000000 6.500000 432.350830 432.350830
   0.342684
500.000000 cop 2$1 0.000000 4.986399 432.481750 432.481750
   0.342339
```

Appendix D

Sample Monte-Carlo Code Result File

Presented here is a sample output of the Monte-Carlo code.

```
C:\Presagis\Suite14\STAGE\samples\monte_carlo\Lagassi\scenario\night_attack
red win =(721)
blue win =(279)
null = 0
probability of detection =1.000000 (0.000000) Warning
probability of interruption =0.763000 (0.026357)
probability of neutralization =0.379076 (0.030070)
probability of effectiveness  =0.279000 (0.027798)

end time 2015_08_10-18_14_19
```

Appendix E

STAGE 3-D View

Presented here are 3-D views of a typical run of the Lagassi scenario.

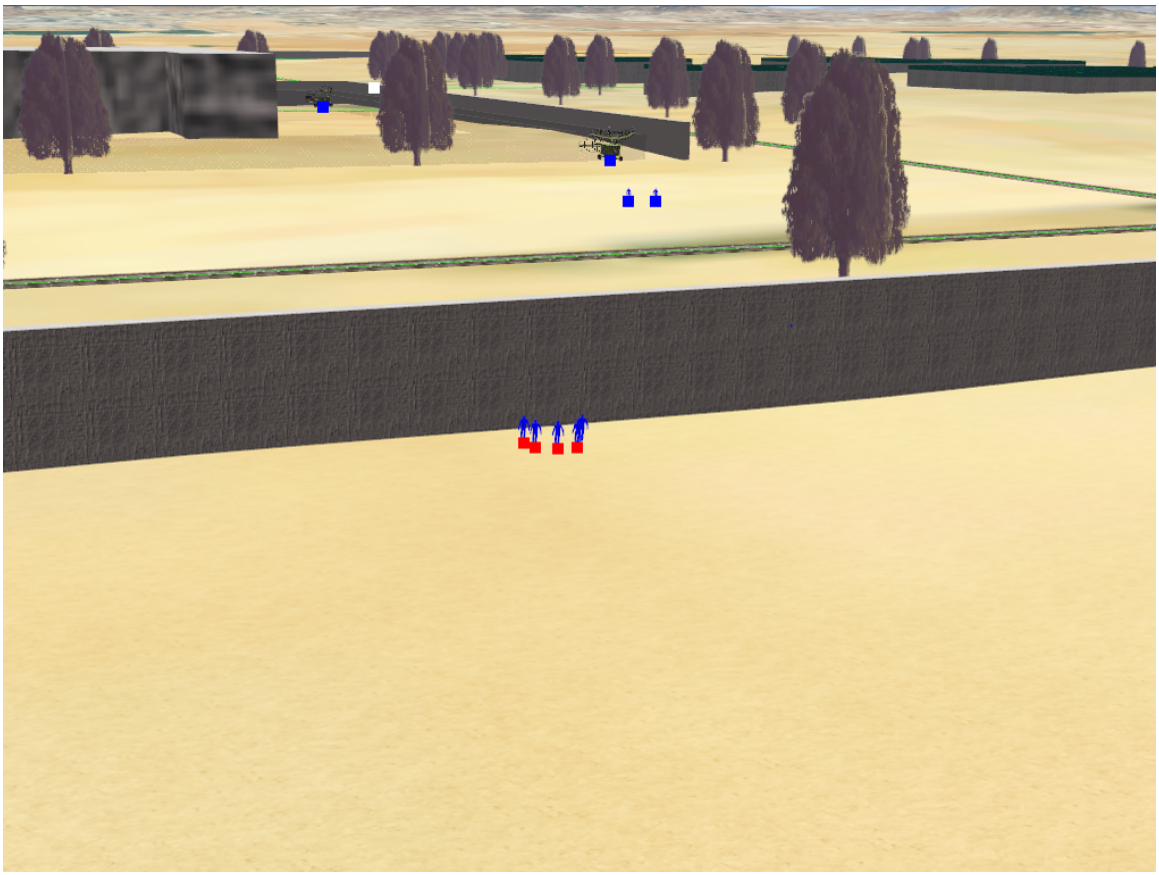

Figure E.1: Adversary approaches wall 3d

Figure E.2: Adversary approaches inner fence 3d

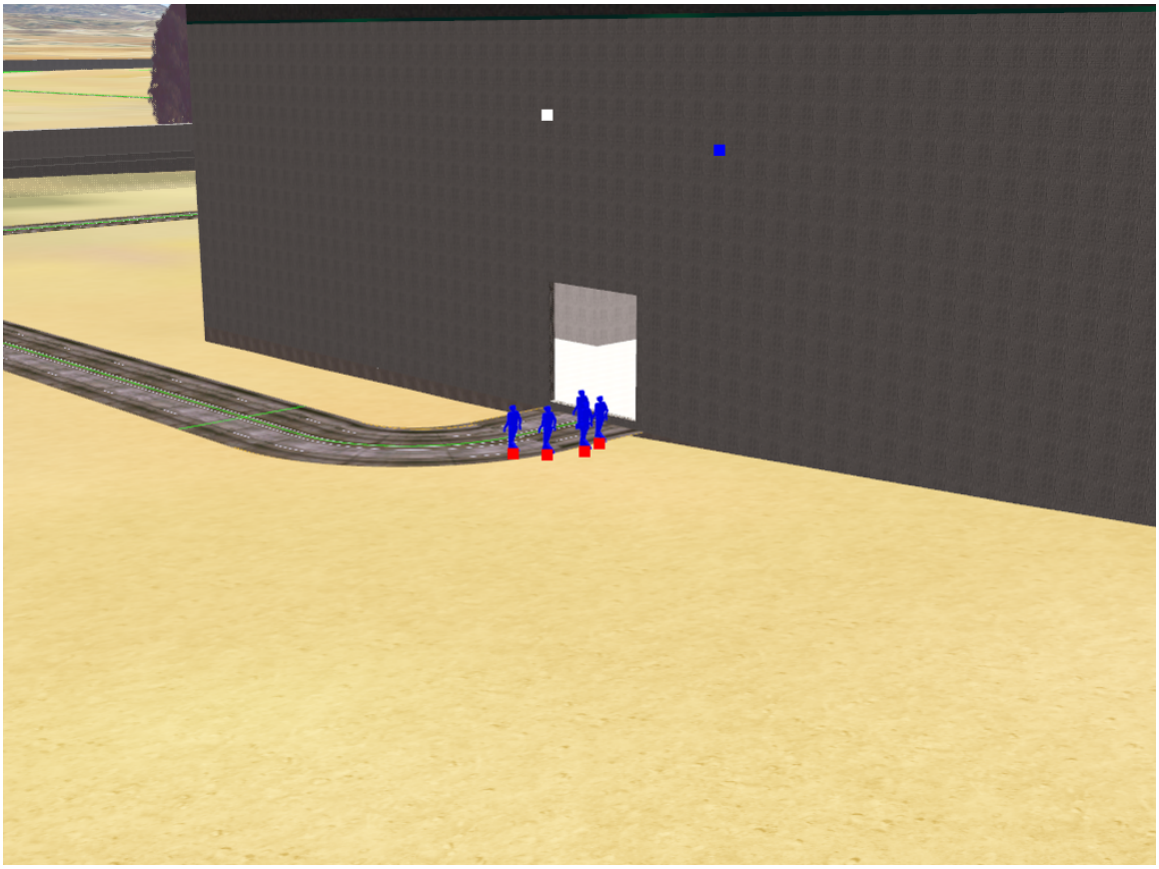

Figure E.3: Adversary breaches rear door 3d

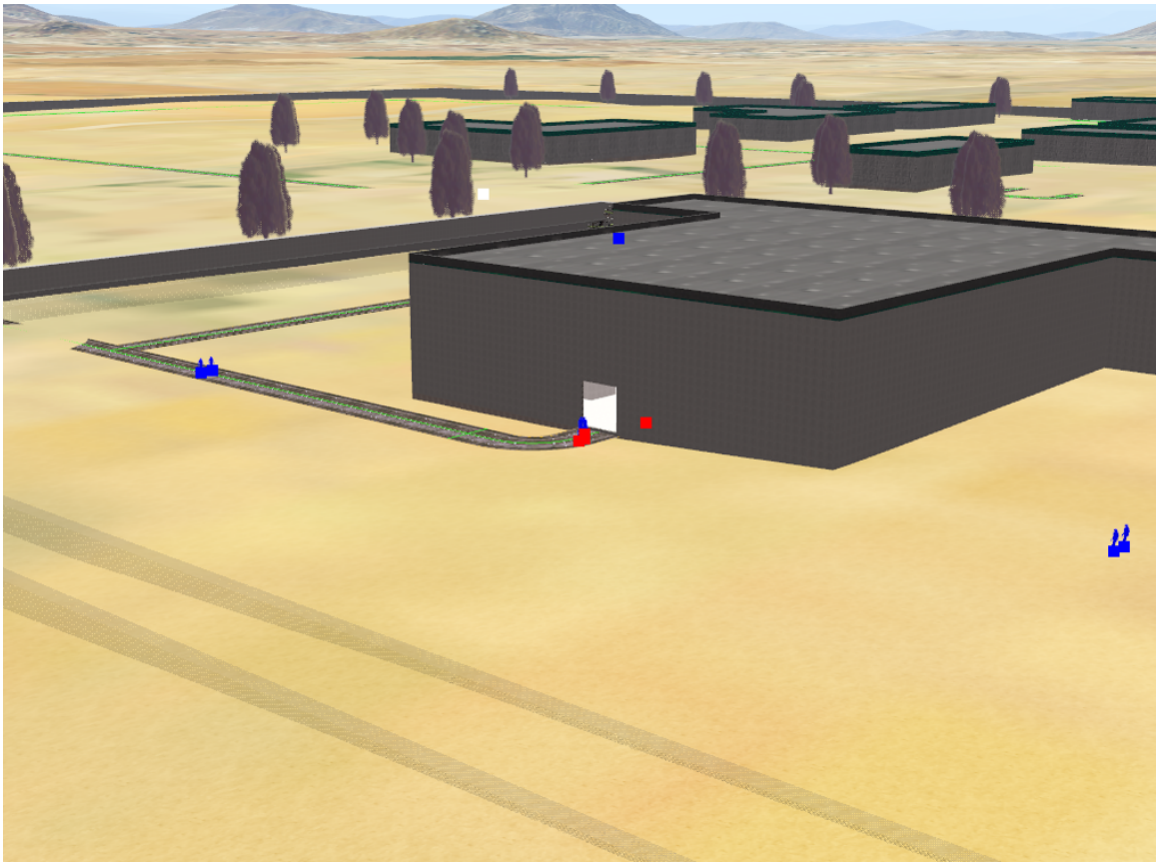

Figure E.4: Adversary engages response force fence 3d

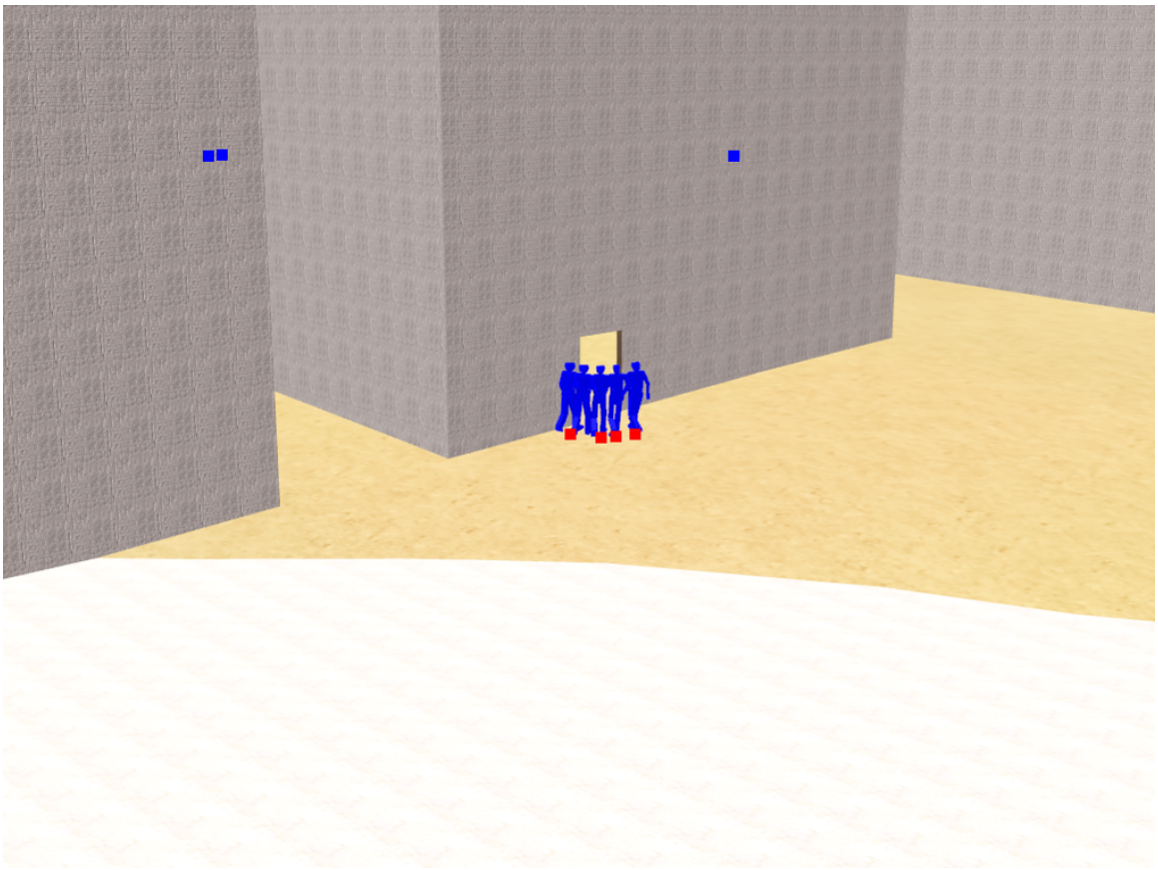

Figure E.5: Adversary breaches products vault fence 3d