

2009

Risk analysis in biometric-based Border Inspection System

Mayra A. Sacanamboy Franco
West Virginia University

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>

Recommended Citation

Sacanambo Franco, Mayra A., "Risk analysis in biometric-based Border Inspection System" (2009).
Graduate Theses, Dissertations, and Problem Reports. 4524.
<https://researchrepository.wvu.edu/etd/4524>

This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

Risk Analysis in Biometric-Based Border Inspection System

by

Mayra A. Sacanamboy Franco, B.A.Sc

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science in Computer Science

Bojan Cukic, Ph.D., Chair
Arun Ross, Ph.D.
Katerina Goseva-Popstojanova, Ph.D.

Lane Department of
Computer Science and Electrical Engineering

Morgantown, West Virginia
2009

Keywords: Risk Analysis, Cost Curves, Software Performance,
Layered Queuing Networks.

© Mayra Sacanamboy, 2009

ABSTRACT

Risk Analysis in Biometric-Based Border Inspection System

Mayra A. Sacanamboy

The main goal of a Border Inspection System is to prevent the entry of individuals who pose a threat to a country. The entry of just one of these persons could have severe consequences. Nevertheless, performing a lengthy border inspection is not possible, given that 240,737 international passengers enter the country in an average day [5]. For this reason, the primary inspection is performed using biometrics traits and information flow processes that have a low false acceptance rate and have a high throughput.

This thesis uses the analytic modeling tool called LQNS (Layered Queueing Network Solver) to solve open models for biometric-based border inspection system and cost curves to evaluate the risk. The contributions of the thesis include a performance model of a biometric-based border inspection using open workloads and a risk model of a biometric-based border inspection using cost curves. Further, we propose an original methodology for analyzing a combination of performance risk and security risk in the border inspection system.

To my family.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Bojan Cukic, for his continuous guidance, support and valuable advices throughout the development of this research and the related papers. I would like to thank my other committee members, Dr. Arun Ross and Dr. Katerina Goseva-Popstojanova for their availability, kindness and directions during my studies.

I would like to thank my sisters and friends for their moral support. In particular, I would like to thank my husband, for his support and desire to help, and my parents, for their constant encouragement and prayers.

I would also like to thank Paola Bracchi for her permission to adapt some diagrams and performance parameters related to the performance analysis of the border inspection system previously studied by her under close workloads to our approach of open workloads.

Mayra A. Sacanamboy
Morgantown, April 2009

Table of Contents

List of Figures	vii
List of Tables	ix
Chapter 1 : Introduction	1
1.1 Software Performance	1
1.2 Software Performance Engineering	2
1.3 Thesis Contribution	3
1.4 Thesis Outline	4
Chapter 2 : Literature Review	5
2.1 Layered Queuing Network Models	5
2.1.1 General Description	5
2.1.2 Task	8
2.1.3 Entry	8
2.1.4 Activity	8
2.2 UML Profile for Schedulability, Performance and Time (UML SPT)	9
2.2.1 Performance Modeling	9
2.3 Transformation from UML to LQN	12
2.4 Characterization of Workloads	15
2.4.1 Open workload	15
2.4.2 Closed workload	15
2.5 Risk	15
2.5.1 Architectural-Level Risk Analysis	17
2.5.2 Performance-based Risk Analysis	19
2.5.3 Risk Analysis Methodology	20
Chapter 3 : Performance analysis	23
3.1 Problem definition	23
3.2 Software Specification	24
3.2.1 Use Case Diagram	25
3.2.2 Sequence Diagram	27
3.2.3 Deployment Diagram	27
3.3 Performance Model Parameters	29
3.3.1 Execution environment	32
3.4 Experiments and Results	38
3.4.1 Public Key Certificate	38
3.4.2 Screening Policies	41
3.4.3 Biometric False Match Rate	42

3.4.4	Replication of PKD.....	46
3.5	Analysis.....	47
Chapter 4	: Risk Analysis	49
4.1	Biometric System.....	50
4.1.1	Receiver Operator Characteristic (ROC) Curve	54
4.1.2	Vulnerabilities of Biometric Systems	54
4.2	Risk Model.....	56
4.2.1	Cost Curves.....	60
4.2.2	Cost Curves for Face Recognition and Fingerprint Recognition.....	64
Chapter 5	: Conclusions.....	73
5.1	Research Summary	73
5.2	Future Work.....	74
Bibliography	76
Appendix A:	Detailed Uses Cases and Sequence Diagrams	80
Appendix B:	Performance Parameters.....	88

List of Figures

Figure 1 LQN model.....	7
Figure 2 LQN forwarding message.....	7
Figure 3 Performance analysis domain model adapted from [16].....	11
Figure 4 Use Case Diagram of a Building Security System [35].....	13
Figure 5 Annotated Sequence Diagram for the access control scenario [35].....	13
Figure 6 Deployment of a Building Security System [35].....	14
Figure 7 Layer Queueing Network model for the Building Security system [35].....	14
Figure 8 Architectural-Level Risk Analysis Methodology. Source [12].....	18
Figure 9 Performance-based Risk Analysis Methodology. Source [6].....	20
Figure 10 Risk Analysis Methodology Processes. Adapted from [24].....	21
Figure 11 Likelihood Levels and meaning. Adapted from [24].....	22
Figure 12 Use Case Diagram for the Biometric-based border Inspection System. Adapted from [4].....	26
Figure 13 Deployment Diagram for the Biometric-based border Inspection System. Adapted from [4].....	28
Figure 14 Average Passengers in the month of December 2007.....	29
Figure 15 Passenger Distribution at the Terminal Booths.....	30
Figure 16 Waiting time in December 2007.....	30
Figure 17 PDF for Dulles Data and Approximated Distributions.....	31
Figure 18 Cumulative PDF for Dulles Data and Approximated Distributions.....	31
Figure 19 LQN Model for the biometric-based Border Inspection using open workloads.....	36
Figure 20 Expanded LQN Model for the primary and secondary inspection options using open workloads.....	37
Figure 21 Average waiting time using different architectural options.....	39
Figure 22 LQN model for the border inspection system using the shared-PKD option... ..	40
Figure 23 Average secondary inspection time for different screening scenarios.....	42
Figure 24 Total waiting time for different screening scenarios.....	42
Figure 25 Total waiting time for different FMR scenarios, assuming $P(\text{impostor})=0.001$	44
Figure 26 Total Inspection time for different impostor prior probabilities.....	45
Figure 27 Total average Inspection time at different combinations of FMR and Watchlist size.....	45
Figure 28 Identification of High-Risk aliens and other violators.....	50
Figure 29 Fraudulent Documents.....	50
Figure 30 Common Biometric Error rates.....	53
Figure 31 FMR vs. FNMR ROC curve. Source [30].....	54

Figure 32 possible Attack points in a biometric authentication system, adapted from [21].	56
Figure 33 Technical threats in the biometric-based border inspection system.....	58
Figure 34 Possible regions in a cost curve.....	61
Figure 35 ROC curve for performance matching of 7 algorithms for Face images. Source [9].	65
Figure 36 ROC curve for performance matching of 18 algorithms for fingerprints. Source [10].	66
Figure 37 Total Inspection time for different impostor prior probabilities using FMR=0.0001 and FNMR=0.001	68
Figure 38 Cost curves for face matching algorithms – Severe condition.	69
Figure 39 Cost curves for face matching algorithms - Guarded condition.....	69
Figure 40 Cost curves for face matching algorithms - Low condition	70
Figure 41 Cost curves for fingerprint matching algorithms – Severe condition.....	71
Figure 42 Cost curves for fingerprint matching algorithms – Guarded condition.....	72
Figure 43 Cost curves for fingerprint matching algorithms – Low condition	72
Figure 44 Sequence Diagram for the traveler examination use case. Adapted from [4] ..	84
Figure 45 Sequence Diagram for the Biographic Checking use case.....	85
Figure 46 Sequence Diagram for the Secondary Inspection in the traveler examination use case. Adapted from [4]	85
Figure 47 Sequence Diagram for the Biometric Verification Use Case.....	86
Figure 48 Sequence Diagram for the Biometric Identification Use Case.....	87

List of Tables

Table 1 Permitted connections between activities in LQN.....	8
Table 2 UML extensions defined for performance modeling.....	11
Table 3 LQN parameterization for Public Key Certificate Experiment	38
Table 4 LQN parameterization for different screening scenarios.....	41
Table 5 FMR variation in the LQN model (Biometric False Match Rate Experiment) ...	43
Table 6 LQN parameterization (Biometric False Match Rate Experiment).....	43
Table 7 Utilization	46
Table 8 Severity Analysis for a biometric-based border inspection system.....	58
Table 9 Databases at port of entry and their approximate update time [18].....	60
Table 10 Definition of Probabilities associated with impostors at the primary and/or secondary inspection points	62
Table 11 Proposed Methodology combining Risk and Performance for a Software System	63
Table 12 Assumed probabilities for the risk scenarios	66
Table 13 Cross-relation between cost curves for Face modality and the performance models.....	67
Table 14 Cross-relation between cost curves for Fingerprint modality and the performance models.....	71
Table 15 Traveler Examination expanded use case.....	80
Table 16 Biographic Checking expanded use case.....	81
Table 17 Biometric Verification expanded use case.....	82
Table 18 Biometric Identification expanded use case	83
Table 19 Hardware Platform Devices and Transfer Rates Specification. Adapted from [58].....	88
Table 20 Estimated Data Size. Adapted from [27][58]	88
Table 21 Scenario Steps with Performance Annotations for the Sequence Diagram of the Traveler Examination. Adapted from [4].....	89
Table 22 Scenario Steps with Performance Annotations for the Sequence Diagram of the Biographic Checking	92
Table 23 Scenario Steps with Performance Annotations for the Sequence Diagram of the Biometric Verification	92
Table 24 Scenario Steps with Performance Annotations for the Sequence Diagram of the Biometric Identification.....	92

Chapter 1 : Introduction

An increasing number of computer systems strives to maximize performance or to minimize cost for a given functionality without reducing the functional requirements or including new system components. To achieve this goal, risk assessment in the early design phase of software lifecycle followed by performance analysis in all phases in the lifecycle guides the selection of the most prominent design among design alternatives, avoiding the implementation of unacceptable designs.

Information generated by a critical system cannot be trusted given the possible existence of software faults. Failures may result in either the loss of life, injury or damage to the environment, high economic losses or failure of a goal-directed activity. Mission-critical system, such as a border inspection system for example, has to quantify the uncertainty whether a traveler represents a treat to the country he wants to enter in. An inspector makes his final decision based on the output given by the biometric and database search modules, and his expertise.

This chapter presents an introduction to software performance and software performance engineering. The section that follows summarizes our contributions to modeling techniques used to solve performance and risk models of biometric-based border inspection systems. This chapter concludes with the thesis outline.

1.1 Software Performance

Performance is a “pervasive quality of software systems” which means that the software components and the underlying layers like hardware, middleware, operating system, among others, exert their influences on it [26]. The most frequently used techniques for evaluating system performance are measurement, simulation and analytic

modeling. Measurement relies on the existence of a live system while simulation and analytic modeling depend on a model of the system under consideration [20].

Among these techniques, the costliest one is measurement, since it is feasible only if there exists a real system with the configuration being studied. Nevertheless, the measurement technique gives the most accurate results, because parameters, such as workload, are representative. Simulation uses the model of the system under study by creating a program which traces the progress of events as discrete steps over time.

Analytic modeling applies mathematical expressions to obtain performance results for the system being studied. This kind of modeling requires simplifying assumptions, which allow analyzing large systems. It is computationally efficient and its parameters are easier to obtain due to their higher level of abstraction. Accuracy is one advantage that simulation has over analytic modeling, nevertheless simulation models are often time consuming and difficult to design, debug, parameterize and execute.

1.2 Software Performance Engineering

Software performance engineering (SPE) is a methodology that has been successfully incorporated into software development [23]. SPE uses quantitative methods to identify architectural and design alternatives that will fulfill performance objectives. Furthermore, SPE allows to improve the design through a better understanding of the performance properties within the system.

Performance issues that can be easily addressed by SPE are identification of potential bottlenecks, determination of the maximum system load, analysis the impact of architectural changes on performance and understanding the influence of particular components to performance. The activities in the SPE include assessment of performance risk, identification of critical Use Cases, selection of key performance scenarios, construction of performance models, analysis of software resource requirements, evaluation of the models and finally verification and validation of models. We briefly discuss these steps below:

Assessment of performance risk is addressed by identifying potential risks and their impact on the project's success in order to deal with them systematically. The impact is a combination of the probability that a failure occurs and the severity of the damage it may cause.

Identification of critical Use Case and selection of key performance scenarios are handled by identifying and selecting the scenarios that are important to responsiveness as seen by users or have performance risk. Those use cases can produce failures in the system or reduce the success of the system if the performance goals are not met.

Establishing performance objectives is carried out by specifying quantitative criteria. Response time is the time interval between a user's request to the system and response from the system. Throughput is the rate at which the requests are processed by the system; Utilization is the fraction of time the component is busy processing requests; Workload intensities specify the level of usage in component.

Construction and Evaluation of performance models: performance models are obtained from performance scenarios, and they are evaluated by quantifying design changes which assess trade-offs and highlight the best alternative. Sensitivity studies indicate the model parameters which produce large changes in the model.

Verification and Validation are the activities that proceed in parallel with the above activities. Model verification is intended to establish whether the model predictions are a truthful reflection of the software's performance, answering to question "are we building the model right?". On the other hand, model validation establishes whether the model exhibits the execution characteristics of the software, addressing the question "are we building the right model?".

In this thesis, an analytical method for solving performance models and a risk methodology are developed and applied to border inspection systems with the goal of establishing the optimal thresholds between the risk and performance. The Stochastic Rendezvous Network Model (SRVN) proposed by Woodside [52] and the Layered Queueing Network Solver (LQNS) [53][54] are the basis for solving the performance models. The SRVN is used to model the system, and LQNS is used to solve the obtained SRVN model.

1.3 Thesis Contribution

We consider the following as the contributions in this work.

- Analyzing the open arrivals in the border inspection model and providing some basic characteristics of the traveler data traffic.

- Comparing the results in this thesis with previously published results.
- Analyzing the risk for a biometric-based border inspection using cost curves.
- Combination of performance risk and security risk in the border inspection system in order to find the most suitable thresholds in the system.

1.4 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 provides basic material about Layered Queueing Network, UML profile for schedulability, performance and time, derivation of LQN models from annotated UML diagrams, and finally overviews risk analysis concepts and methodologies. Chapter 3 introduces the characteristics of travelers in the environment of border inspection points. The result obtained by analyzing the distribution is provided too. A layered prediction model is devised and shown to be accurate. Chapter 4 explains our approach to risk analysis of a biometric-based border inspection system. Chapter 5 presents conclusions and future work.

Chapter 2 : Literature Review

This chapter supplies a brief overview of analytic performance modeling with Layered Queuing Network models and software modeling with UML performance notation, which we use in the analysis in this thesis. We reviewed different risk analysis methodologies: architectural risk analysis, performance risk analysis and a risk methodology. Therefore, we gain understanding of the different types that are suitable for the context, which is defined by misclassification costs from the verification algorithms deployed in the system.

2.1 Layered Queuing Network Models

The Layered Queuing Network (LQN) model, proposed by Franks, is an extended queueing model, which we use in the performance analysis discussed in this thesis. We use LQN performance model to represent system with software queueing and rendezvous. This modeling technique is well suited for systems with parallel tasks running on a network.

2.1.1 General Description

An LQN model denotes model-based performance where each layer of the model is represented by a network of queues. It is described by a directed graph with nodes that represent software entities and hardware devices, and with arcs that represent service requests. A LQN task, represented by a parallelogram, can act as a server or a client interacting with other tasks. Generally, it represents software components. When a task does not receive requests but only generates requests is called a *reference task*, and it represents a pure client or a load generator. On the other hand, a *pure server* task only

receives requests and does not initiate requests. A server usually represents hardware resources such as processors and I/O devices, among others.

Available services through a task are represented by *entries*, where each entry has its own demands for other services and execution time which are given as model parameters. An entry is drawn as a slice of the corresponding task. A single input queue is offered by every task, where requests for its different entries wait together to be served. A single request queue is also shared in a multi-server node. A multi-server node is composed of several identical servers that work in parallel.

Figure 1 shows an example LQN model. At the top there is the client, which sends requests for `verifyData` and/or `writeInfo` services to the task named `Application`. Each `Application` entry requires services from two different entries of the `DB` task. Every software task is running on a processor node, drawn as a circle.

The following parameters must be included in the LQN model. If it is an open workload model, we need to specify the arrival rates. Otherwise, we specify the associated populations and “think” times. In addition, for each device, we need to define the average service time, the average number of visits either when the software task entry is seen as a client to a device, or when the software task entry is communicating with another task entry. Finally, for each software and hardware servers, we establish a scheduling discipline, and the average message delay in each request.

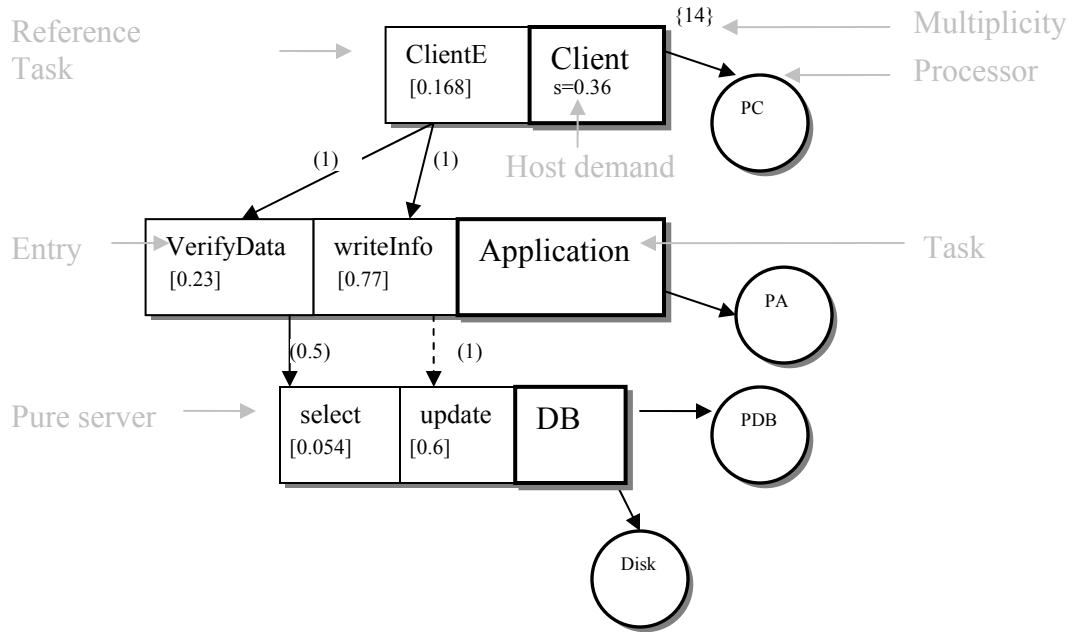


Figure 1 LQN model

Synchronous, asynchronous and forwarded are the type of communication messages between entries that LQN is able to recognize. A Forwarded message uses a forwarding chain, so the client, who submits requests, will wait until a replay from the last server within the chain is received (see Figure 2). Synchronous and asynchronous messages work in the standard way. In the former, the client is blocked until the requested server replays as in remote procedure calls (RPC). In asynchronous communication, the client is not blocked while the server is working autonomously on the received request.

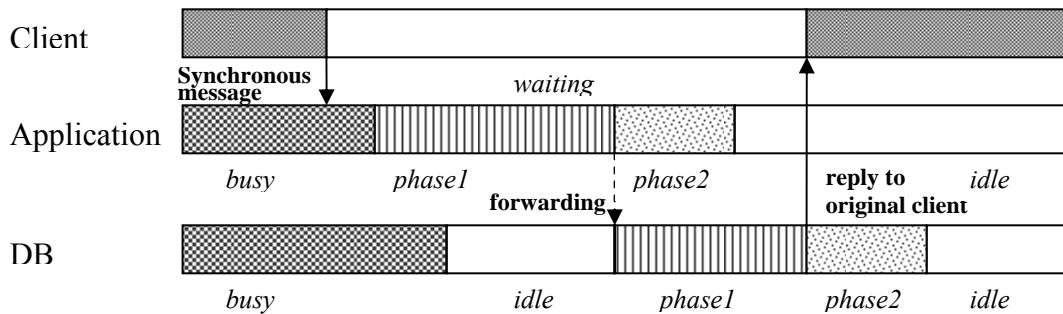


Figure 2 LQN forwarding message

2.1.2 Task

A task represents a resource, where the resource could be a process in a system, client, hardware device, and buffer among others. A task always runs on a processor and has a queue that uses one of the scheduling methods such as, FIFO (first-in, first out), PPR (priority, preemptive resume) and HOL (head-of-line priority).


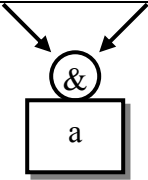
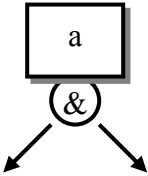
2.1.3 Entry

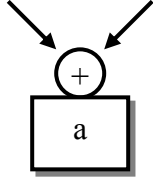
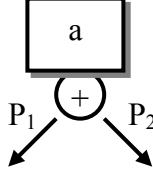
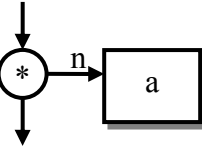
Entries represent different kind of services offered by a task. They are able to receive just one kind of requests at a time. Requests can be synchronous or asynchronous. The parameters in an entry are defined by using either phases or activities. Activities are recommended when the internal behavior of the task is so complex that forks and joins are used. On the other hand, phases are appropriate when the behavior of the task can be specified as a sequence of one to three activities.

2.1.4 Activity

An activity represents the finest level of detail required in order to describe one or more execution scenarios. Parallelism within task can be represented by using activities, because they are able to depict concurrent threads of control and also the randomness in the execution process by introducing probabilities between different paths. Table 1 summarizes the precedence types used to connect activities.

Table 1 Permitted connections between activities in LQN

Name and Description	Representation
<i>Connecting arc</i> Transfer control	
<i>And-join</i> Parallel activities are synchronized at that point	
<i>And-fork</i> Start of concurrent execution	

<p>Or-join</p>	
<p>Or-fork After executing the activity one of the paths is selected with probability p.</p>	
<p>Loop The activity is repeated an average of n times</p>	

2.2 UML Profile for Schedulability, Performance and Time (UML SPT)

UML SPT offers a common framework for annotating UML models with predictive quantitative analyses capability. Existing and future model analysis techniques benefit from the offered features [16]. The process of system design and refinement is done through the parametric understanding of an implementation with functional requirements. The evaluation is typically done by executing test cases and measuring the results. The following section focuses on UML Performance notation.

2.2.1 Performance Modeling

Incorporating performance analysis in UML models facilitates the association of performance quality of service (QoS) characteristics with particular elements of the UML model. Furthermore, the specification of execution parameters in the UML models implies their use in modeling tools that will predict performance characteristics. Finally, performance requirements can be captured straightforward from the design context.

The performance analysis domain model is depicted in Figure 3. A better explanation of each element from the domain model is presented next.

Performance context is used to explore a variety of situations concerning a particular set of resources. In order to explore different QoS values such as load intensity and

response delay, among others, in the same performance context, a parameterization is needed.

Scenario is an ordered and finite set of activities (scenario steps) that describe the performance context, with response time and throughput. A workload is defined for each scenario, representing the intensity of use. The scenario can be either described by paths that fork, join, loop, and are alternative with some probability.

Workload indicates the intensity of requests over resources within a specific scenario. Workload specification can be open or closed. Open workloads are used in systems where jobs enter the system disjointedly of job completions. Generally, they are modeled through Poisson arrivals since jobs follow that predetermined pattern. In closed workloads, a fixed number of jobs enter the system and continue circulating within the scenario with an associated “think time”, which represents an external delay period outside the system.

Resources are represented as servers. They can be active or passive. Active resources have associated service times and they are the servers in the performance models. On the other hand, passive resources have associated holding times and they are acquired and released during scenario execution.

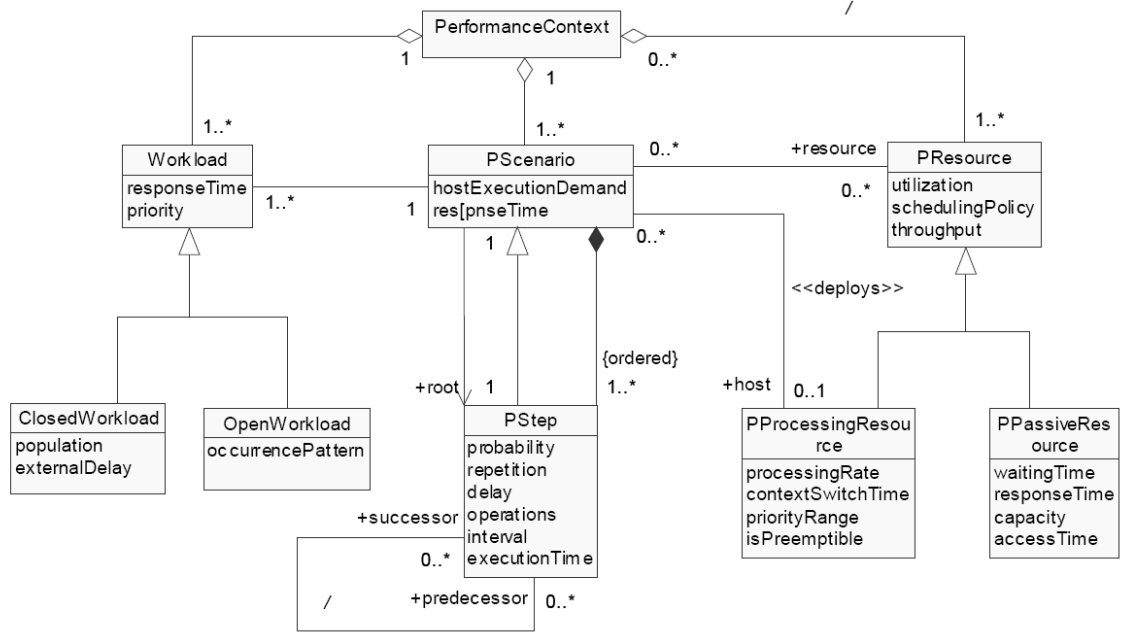


Figure 3 Performance analysis domain model adapted from [16]

The UML extensions required for displaying the relevant performance values are presented in Table 2. Value types such as PAPERfValue and RTarrivalPattern use a BNF notation that is further explained in [16].

Table 2 UML extensions defined for performance modeling

Stereotype	Associated Tags	Value Types
<<PAcontext>>		
<<PAopenLoad>>	PArespTime PApriority PAoccurrence	PAperfValue Integer RTarrival Pattern
<<PAclosedLoad>>	PArespTime PApriority PApopulation PAextDelay	PAperfValue Integer Integer PAperfValue
<<PAstep>>	PAdemand PArespTime PAprob PArep PAdelay PAextOp PAinterval	PAperfValue PAperfValue Real Integer PAperfValue PAextOpValue PAperfValue
<<PAhost>>	PAutilization PAschedPolicy PARate PActxtSwT	Real {'FIFO', 'HeadOfLine', etc} Real PAperfValue

	PAprioRange PApreemptable PAthroughput	Integer Boolean Real
<<PAresource>>	PAutilization PAschedPolicy PAschedParam PAcapacity PAaxTime PArespTime PAwaitTime PAthroughput	Real {'FIFO', 'PriorityInheritance', etc} Real Integer PAperfValue PAperfValue PAperfValue Real

2.3 Transformation from UML to LQN

Previous works [4][19][33][34] have developed methodologies that use information from different UML diagrams in order to incrementally generate performance models carrying out an LQN translation technique. The most convenient ones were proposed by [4][19]. Use Case Diagrams provide information on the workloads, which identify the services and the users of the system. Sequence Diagrams are used to obtain the software execution model because they reflect the system behavior given that they describe in detail the scenarios that are critical to the system performance. Finally, Deployment diagrams state different physical contexts needed for system analysis [19].

An example of developing a performance model of a Building Security System is presented in Figure 4-Figure 7 from [35]. The system provides access control and video surveillance among other functions that are depicted in the use case diagram in Figure 4. For simplicity, we display only the annotated sequence diagram for the access control scenario in Figure 5. In the sequence diagram: a user inserts his card into a door reader, this information is transmitted to a server, which checks the access rights associated with the user's card in the data base and then either grants or denies access. This scenario possesses some requirements, such as the transaction completion time of one second, and the access request load of about 1 request per 2 seconds on average. The information from the UML diagrams is required in order to build the layered queueing network model.

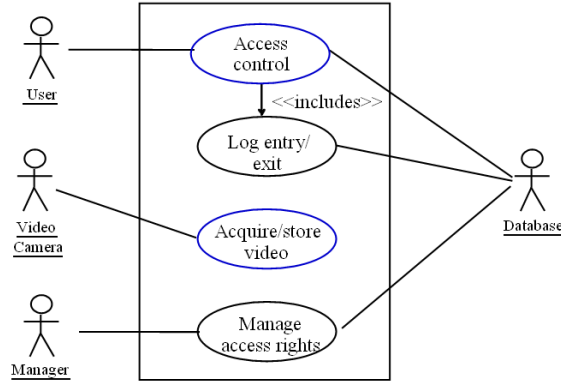


Figure 4 Use Case Diagram of a Building Security System [35]

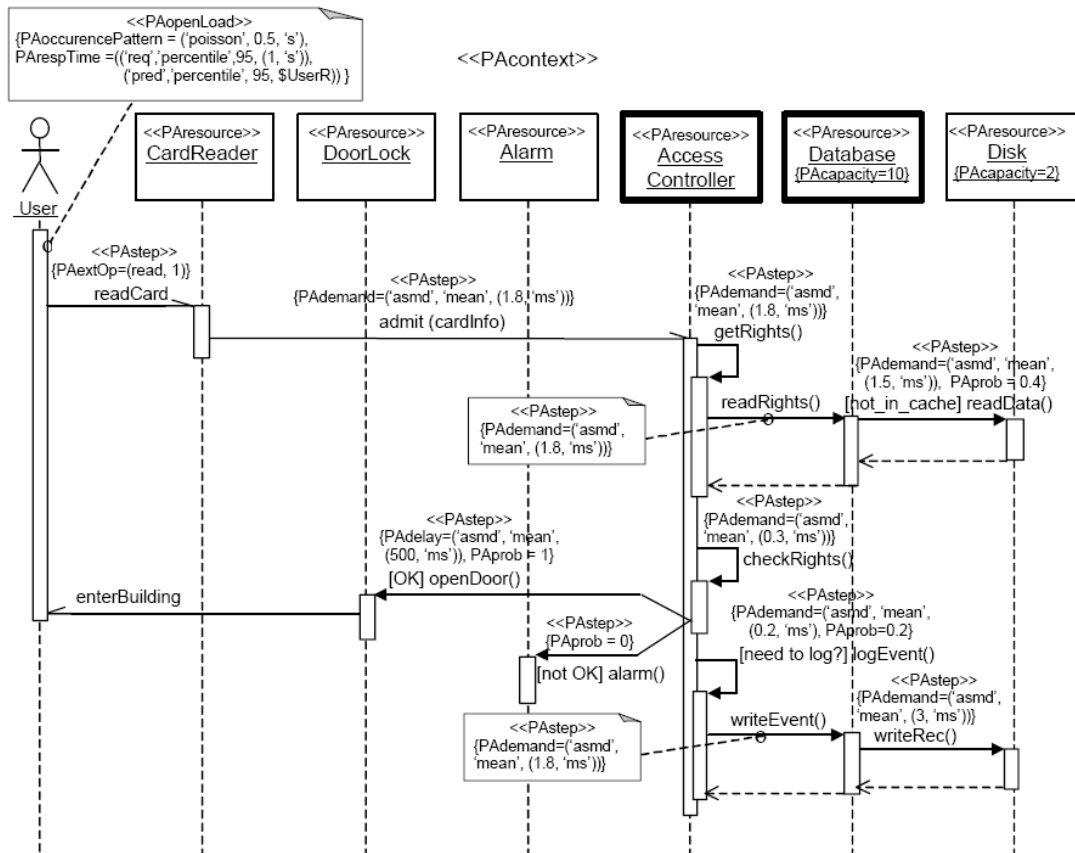


Figure 5 Annotated Sequence Diagram for the access control scenario [35]

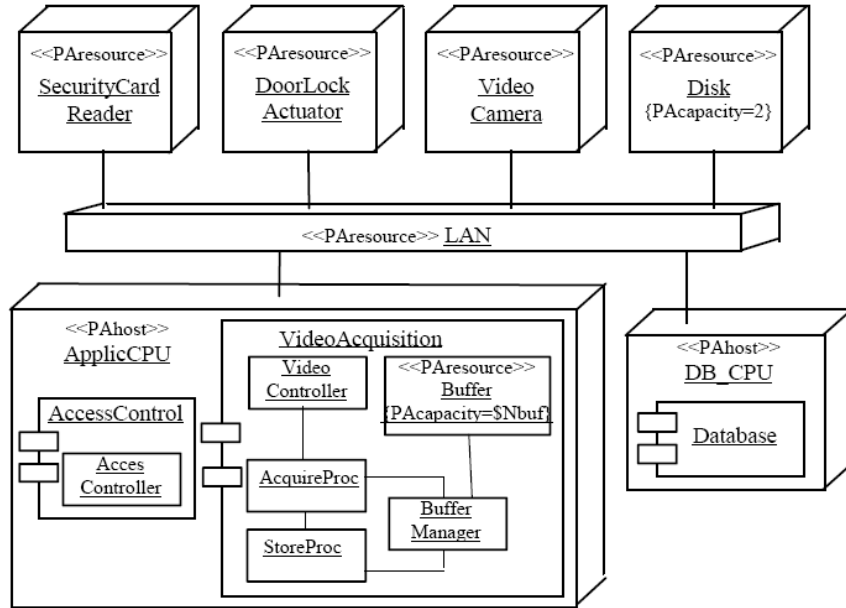


Figure 6 Deployment of a Building Security System [35]

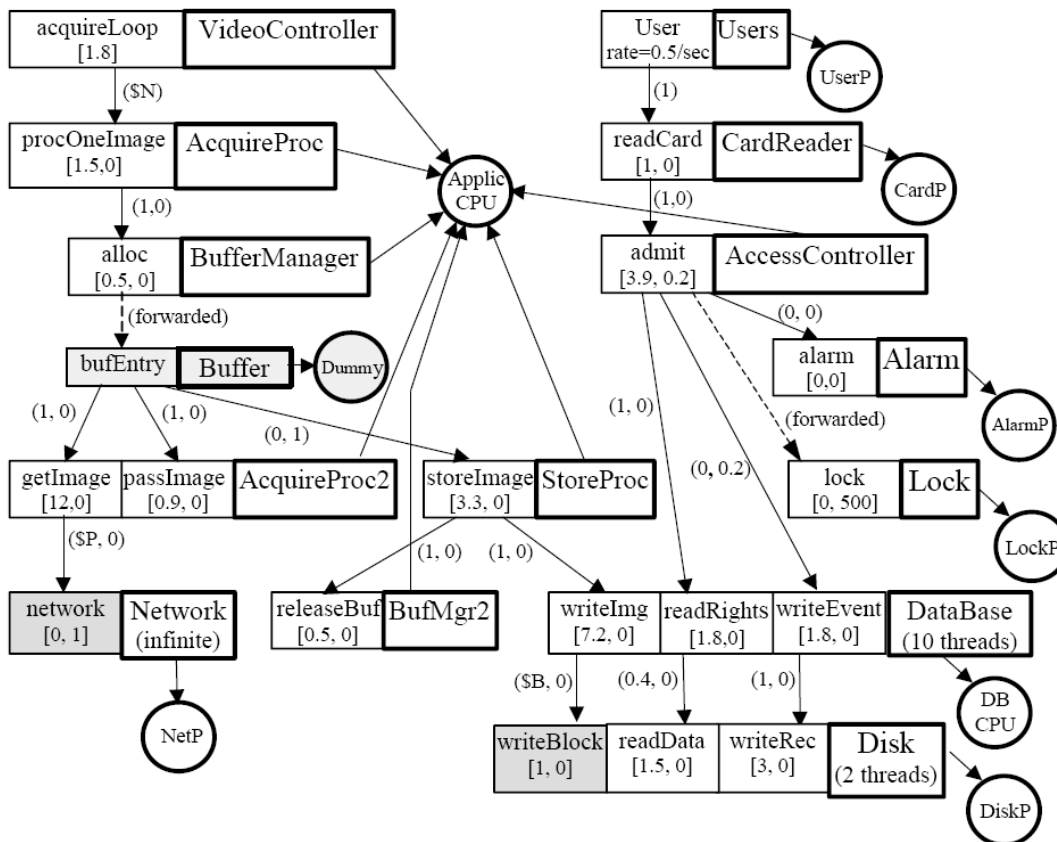


Figure 7 Layer Queueing Network model for the Building Security system [35]

2.4 Characterization of Workloads

The type of workload that is employed in the performance model has to be considered as important as workload metrics such as service demand distribution, think time, etc, in order to provide accurate representations of the system under study [22].

2.4.1 Open workload

Open workload assumes that a user generates a request, waits to receive a response and finally leaves. Basically, a new request to the system is only initiated by a new user arrival. At any instance of time, the system has a different number of jobs. Nevertheless, the throughput X is assumed to be known and equal to the arrival rate λ , therefore the main goal is to characterize the distribution of jobs in the system.

2.4.2 Closed workload

In this kind of workload, the system is used unendingly by some fixed number of users. In general this number of users identifies the multiprogramming level denoted by N . Every user in the system submits jobs and after receiving the response, waits some amount of time in order to submit another job request. New requests are initiated after the completion of previous ones. The total number of jobs in the system is constant and the total number of users in the system can be defined as $N=N_{\text{think}}+N_{\text{process}}$ where, ones are thinking N_{think} , and other ones are either queuing to run jobs or running jobs N_{process} .

Other parameters such as server load and response time are respectively defined as fraction of time that the server is busy resulting from the product of the mean service demand $E(S)$ and the mean throughput X ; and the response time, the interval of time between a request is submitted and resolved.

2.5 Risk

To comprehend the nature of risk, we need several related definitions. *Vulnerability* is defined as a flaw in the system that can be unexpectedly activated or purposely exploited. *Hazard* is a situation or event that potential could lead to accidents. Identifying and assessing those hazards is the first activity that must be carried out within the risk analysis [24].

The *hazard analysis* identifies and assesses those sources of danger on the system. It can be divided in the following steps:

1. Preliminary Hazard Analysis (PHA) identifies crucial system functions and general system hazards, which lead to identification of safety design criteria and requirements in the early life cycle.
2. System Hazard Analysis (SHA) examines possible hazards caused by interfaces between subsystems that working together can impact system safety; its main goal is to advise changes and controls and assess design responses according to safety requirements.
3. Subsystem Hazard Analysis (SSHA) identifies and assesses hazards related with subsystems that individually can affect the overall system safety.
4. Operating and Support Hazard Analysis (OSHA) examines hazards created by human-machine interfaces, it is performed throughout system use and maintenance stages.

There are several models and techniques that can be used in hazard analysis: Checklist, Fault Tree Analysis (FTA), Event Tree Analysis, Failure Modes and Effects Analysis (FMEA), Failure Modes, Effects, and Criticality Analysis (FMECA), formal methods (CSP, CCS, hybrid automata).

After identifying hazards in the system, it is appropriate to define the severity and probability of occurrence for each recognized hazard. Finally, we can define risks associated with the system: the *Risk* is defined as a combination of the likelihood of an event and its associated severity. Therefore, risk increases when either the likelihood or the severity increase and the other component does not decrease by the same proportion [15].

Risk analysis is conducted in order to find answers to questions such as: what can fail? How likely is it to happen? And given that it occurs, what are the effects? [1]. The risk analysis can be performed at the architectural level allowing an early detection and correction of problems that could be less costly as it will be if detected in a late stage of the software life-cycle such as implementation.

2.5.1 Architectural-Level Risk Analysis

It is known that failures of critical components and connectors in the system have a major impact on the overall system reliability. In [12] Goseva-Popstojanova et al, proposed a risk analysis process that can be used in the early stages of software lifecycle. It uses the behavior contained in UML specifications, in particular, use cases and sequence diagrams in order to establish the risk factors of components and connectors. These are determined by measuring their dynamic complexity (2.5.1-1) and coupling (2.5.1-2), respectively. In a given scenario S_x , rf_i^x is the risk factor of component i for scenario x , DOC_i^x is the normalized complexity of component i in scenario x , svt_i^x is the severity level of component i in scenario x . EOC_{ij}^x is the normalized coupling and svt_{ij}^x is the severity level for the connector between the i th and the j th components in the scenario S_x .

$$rf_i^x = DOC_i^x \cdot svt_i^x \quad (2.5.1-1)$$

$$rf_{ij}^x = EOC_{ij}^x \cdot svt_{ij}^x \quad (2.5.1-2)$$

The dynamic complexity is determined by using UML state charts from which we can obtain both the cardinality of subset of states for a component i in the scenario S_x ($|C_i^x|$) and the cardinality of subset of transitions traversed ($t_i^x=|T_i^x|$) (2.5.1-3) and the normalized dynamic complexity (2.5.1-4).

$$doc_i^x = t_i^x - c_i^x + 2 \quad (2.5.1-3)$$

$$DOC_i^x = \frac{doc_i^x}{\sum_{k \in S_x} doc_k^x} \quad (2.5.1-4)$$

The dynamic coupling is determined by using UML sequence diagrams from which we can obtain the number of messages sent from component i to component j during scenario S_x (M_{ij}^x) and it is normalized by dividing over the total number of messages exchanged in scenario S_x as in (2.5.1-5).

$$EOC_{ij}^x = \frac{|MT_{ij}^x|_{i,j \in S_x, i \neq j}}{|MT^x|} \quad (2.5.1-5)$$

For a given scenario, a severity level is assigned to each component and connector based on the Failure Mode and Effect Analysis technique. Next, scenario risk factors are estimated considering multiple failure states in order to represent failure modes with different severity level. The steps of the methodology are presented in Figure 8 .

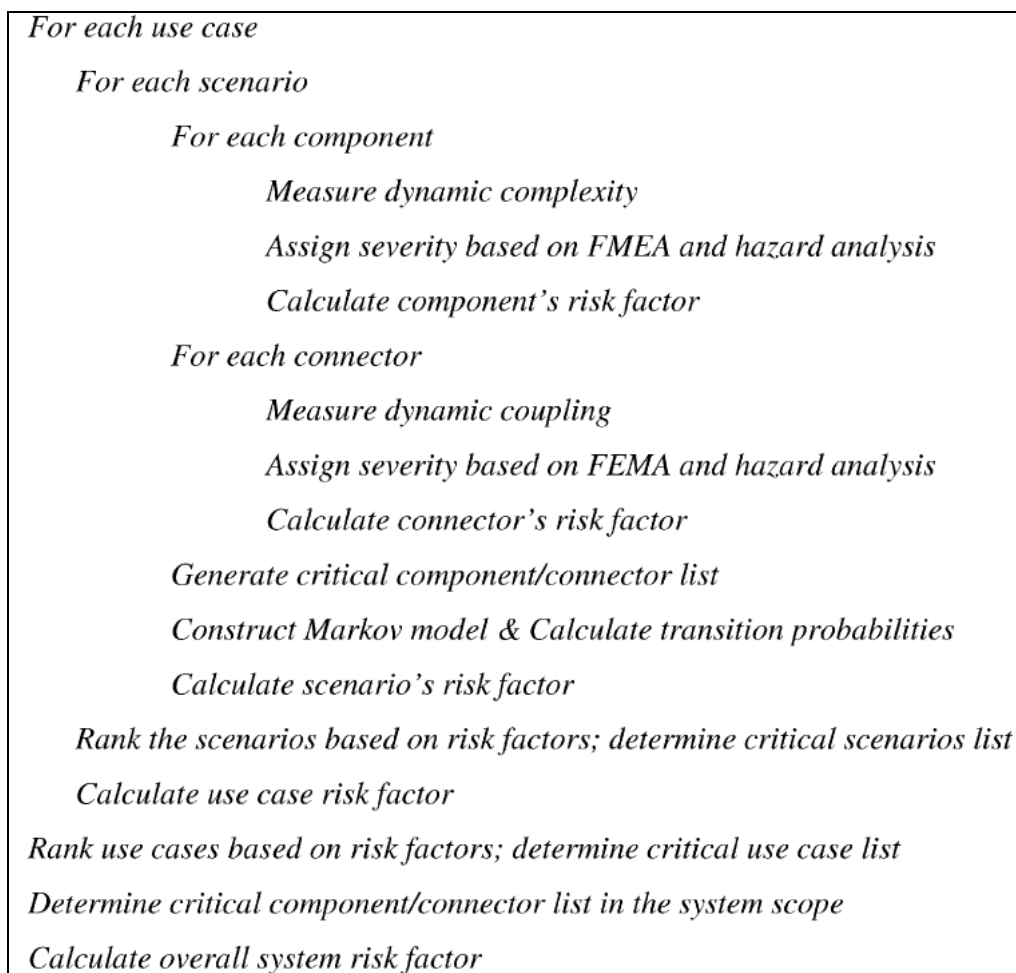


Figure 8 Architectural-Level Risk Analysis Methodology. Source [12]

Construction of the scenario risk model requires first, a control flow graph, in which states map active components and arcs map connectors assuming they have a Markov

property which allows the representation of the model software execution behavior for scenario S_x as an absorbing discrete time Markov chain (DTMC). Second, inclusion of failure states into the DTMC as absorbing states.

Finally, the risk factor for each use case is acquired by averaging the risk factors of all scenarios S_x present in the use case, and similarly, the overall system risk factor is obtained by averaging the use case risk factors.

2.5.2 Performance-based Risk Analysis

Performance risk is created by violations of performance requirements [6]. This methodology indicates both risky software components and risk scenarios by using annotated UML diagrams such as Use Cases, Sequence diagrams and Deployment diagrams in order to estimate the probability of performance failure in combination with the failure severity estimation from the Functional Failure Analysis.

An outline of the methodology is depicted by Figure 9. The methodology, the demand vector is defined for an action/step of a component and by size of the data exchanged for an interaction of a connector. A Software Execution Model is obtained by translating the sequence diagram dynamics into a flow graph whose parameters come from the demand vectors. A Service demand for each hardware device is obtained from the annotated deployment diagram.

A stand-alone analysis is executed, where the completion time of the whole scenario is based on a dedicated hardware platform with a single user workload, if the time value from this analysis is not violating the performance objective then a posterior investigation with a realistic workload in the presence of contention for resources is carried out in order to find the probability of failure. On the other hand, when the time value of the stand-alone analysis violates the performance objective, it is clear that the software system deployed on that specific hardware architecture is not suitable for the requirements, and then the failure probability is set to one.

In the following steps, both calculation of the asymptotic bounds of the performance model and an estimation of probability of failure as a violation of the performance objective are achieved. Additionally, the functional failure analysis is done over the system-level sequence diagram, where information for each event in the system-level sequence diagram is associated with a failure mode, its effects and its severity level.

Finally, estimation of the performance risk of a scenario is executed by calculating the product of the probability that the system fails in meeting the established performance objective with the severity associated with this failure in the scenario. Identification of high-risk components in a scenario is done by finding the component with the highest residence time and identification of high-risk scenarios is done by performing cross scenarios comparisons through normalization of the overall residence time of components in a specific scenario with the response time of the scenario.

INPUT:

Performance objective.

UML diagrams: Use case Diagram, Sequence Diagram, and Deployment Diagram.

For each Use Case**For each scenario**

STEP1- Assign demand vector to each action/interaction in Sequence diagram; build a Software Execution Model.

STEP2- Add hardware platform characteristics on the Deployment Diagram; conduct stand-alone analysis.

STEP3- Devise the workload parameters; build a System Execution Model; conduct contention-based analysis and estimate probability of failure as a violation of a performance objective.

STEP4- Conduct severity analysis and estimate severity of performance failure for the scenario.

STEP5- Estimate the performance risk of the scenario; identify high-risk components.

OUTPUT:

Probability of performance-based risk of the scenarios.

Identification of performance-critical components.

Figure 9 Performance-based Risk Analysis Methodology. Source [6]

2.5.3 Risk Analysis Methodology

The National Institute of Standards and Technology proposed a methodology for risk management for information technology systems [24]. The methodology consists of sequential steps which provide a foundation to assess the scope of the probable threat and the risk associated with the system. Furthermore, an identification of suitable controls for reducing or eliminating risk can be selected in order to provide better protection of the mission-critical system.

Figure 10 presents the selected steps of the risk analysis methodology. System characterization is the first step, followed by hazard identification, vulnerability assessment and control and impact analysis.

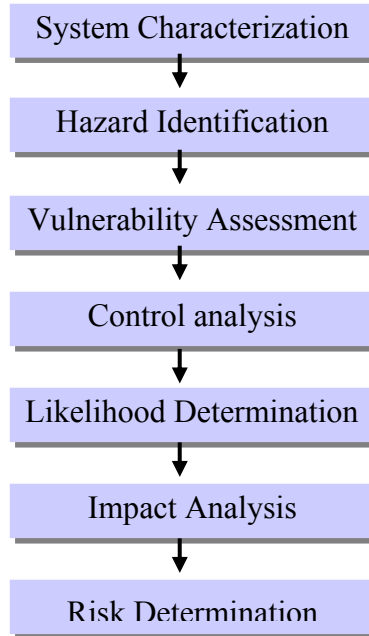


Figure 10 Risk Analysis Methodology Processes. Adapted from [24]

System Characterization delineates the system boundaries and the system related information in order to understand the environment. The information is usually about hardware, software, system interfaces, data sensitivity and criticality, system mission, system security policies, among others.

Threat Identification recognizes potential conditions that might exploit a specific system vulnerability. Given that those potential conditions by themselves in the non existence of vulnerabilities to exploit do not constitute a risk, it is necessary to consider threat-sources, vulnerabilities and current controls in order to establish the likelihood of a threat. Examples of threat-sources are environment, nature or human being.

Vulnerability Assessment identifies and evaluates the vulnerabilities related to the system environment identified in the system characterization step. Finally, it provides a foundation for determining mitigation measures.

Control Analysis studies the current and future controls imposed in the system with the purpose of minimize or eliminate the probability of a potential threat exploiting system vulnerability.

Likelihood Determination obtains a general likelihood which suggests the probability that the vulnerabilities already identified may be exploited within the system. Factors such as source of the threat and its motivation, nature of the vulnerability, and current control mechanisms to block possible flaws, should be considered.

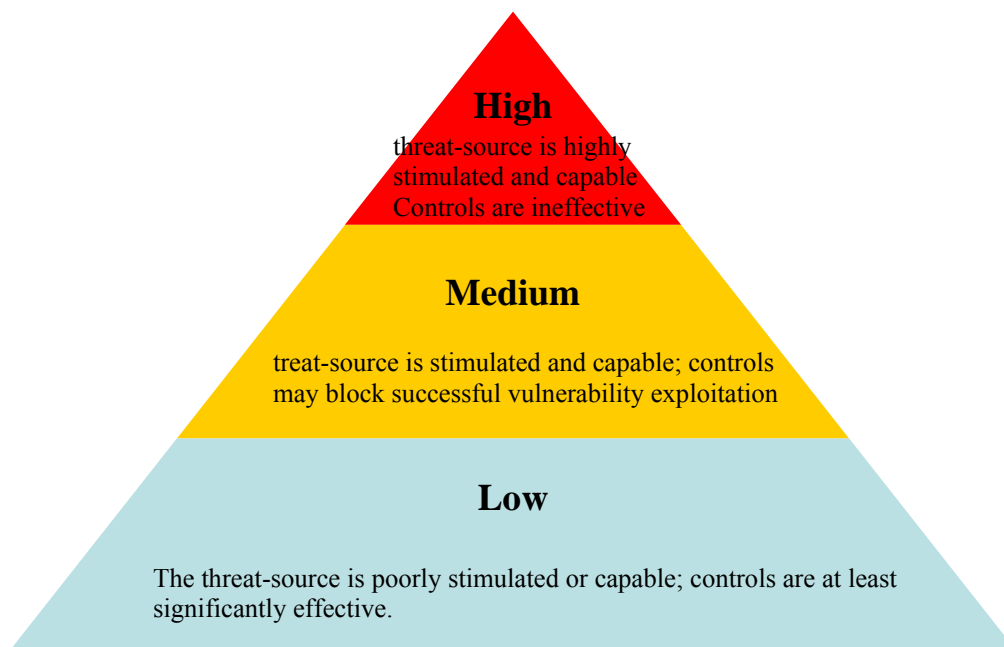


Figure 11 Likelihood Levels and meaning. Adapted from [24]

Impact Analysis measures the unfavorable effect of successful system's vulnerabilities exploitation. The unfavorable impact of a security event can be described in terms of loss or degradation of integrity, availability and confidentiality.

Risk Determination is the determination of risk for a particular threat/vulnerability pair can be expressed as a function of likelihood of a given threat, and the magnitude of the impact.

Chapter 3 : Performance analysis

In this chapter, we present the configuration of the biometric-based border inspection system (structure and functionalities) using UML diagrams. We build and parameterize the performance model using information from similar systems [4][27][28][29] and open workloads. We estimate open workloads by analyzing travelers' arriving pattern from operational data at Dulles International Airport [5]. Finally, we create and evaluate several performance experiments with the purpose of identifying configurations that offer minimum waiting time among the different types of security and architectural options.

3.1 Problem definition

Nowadays, the most trustworthy mechanism to authenticate users is by using biometrics. International travelers at a border inspection point authenticate themselves through their biometrics features, fingerprints and face, which provide an enhancement in the security of the authentication process. Modeling a system like that is a challenging task, because it requires attention to multiple factors that may affect the accuracy of the model [56]. Our research goal is to evaluate the effects on service and waiting time for different types of security and architectural options within our performance models.

Emerging international standards define the format a machine readable documents (MRTD). Together with biographical data, MRTD stores document holder's biometric information. In order to minimize the likelihood of accepting false credentials, the authenticity and integrity of the data inside the traveler's MRTD are reinforced using Public Key Infrastructure (PKI). When presented with decrypted information, the immigration officer shares traveler's information with the Traveler Name Server (TNS). TNS alerts the officer about traveler's admissibility status. Concurrently, the officer

interviews the traveler and collects his or her biometric information, currently a digital photo and ten fingerprints. Collected biometric information is stored and processed at the Traveler Biometric Server (TBS). If this primary inspection has a successful outcome, the traveler is admitted into the country, otherwise he or she is referred to a secondary inspection point where another officer will perform a longer interview and perform multiple checks against watch lists. [56]

We have been engaged in the development of analytical performance models for border inspection points. The goal of performance modeling is to analyze suitability of system requirements, such as the organization of the PKD, optimization of workflows and maximization of the passenger throughput. A related goal has been the comparison of modeling costs and benefit between analytical models and elaborate simulation analysis models.

Our models assume that all travelers possess an MRTD. The performance model requires determining the type of workload generator, possible bottlenecks, types of scheduling policy, and the distribution of servers and resources. Workload generators are generally classified as open or closed [56]. In previous work, Bracchi et. al. [3] [4] used closed workloads for modeling of border inspection points. Those models served as the basis for the present work, which is an extension of [3] [4]. One of our contributions is the use of open workloads. Open workloads are used in systems where jobs enter the system disjointedly of job completions. On the other hand, in closed workloads, a fixed number of jobs enter the system and continue circulating within the system with an associated processing time. From the practical stand point, open workloads represent our applications better and provide modeling results which are easier to communicate to customers.

3.2 Software Specification

The border inspection system can be described by its structural and behavioral software elements [4], where the first one describes logically or physically its software elements and their corresponding interconnections, and the second one describes the software compartment at runtime. In describing the requirements, we will follow the methodology proposed by Bracchi et. al. [4].

3.2.1 Use Case Diagram

This diagram presents the global interactions including variants between the system and the actor who is a physical or logical entity demanding services from the system.

For the biometric-based border inspection system, Bracchi [4] identified four main processes: traveler examination, biographic checking, biometric verification and biometric identification. Additionally, the related actors to the system under study are: travelers and the interagency border inspection system. Those processes and actors are considered as well in the present work.

Figure 12 shows the use case diagram identified for the biometric border inspection system. In “Appendix A: Detailed Uses Cases and Sequence Diagrams”, we present expanded use cases and their corresponding sequence diagrams.

The *traveler examination* process begins when a traveler arrives to the border inspection point and presents his/her MRTD (machine readable traveler document) to the primary officer in the booth. The primary officer validates the MRTD by scanning it through the MRTD reader which returns the MRTD’s corresponding digital signature that is confirmed by employing the Public Key Certificate stored in the Public Key Directory or in the MRTD itself. Additionally, the system verifies the Public Key Certificate Authority, machine readable zone (MRZ) and face image stored in the MRTD. Once the officer has validated that the MRTD is original, he requests a *biographic check* while at the same time he requests the traveler to submit his/her fingerprint and face image, used for *biometric verification* (a one-to-one match). Further, the officer interviews the traveler in parallel while the system is retrieving the traveler’s profile. The system alarms the officer when there is a possible mismatch between the records in the system and the live biometric data, so the officer sends the traveler to a secondary inspection where a *biometric identification* (a one-to-many match) process is initiated. In the absence of problems, the primary officer grants the traveler access to the country.

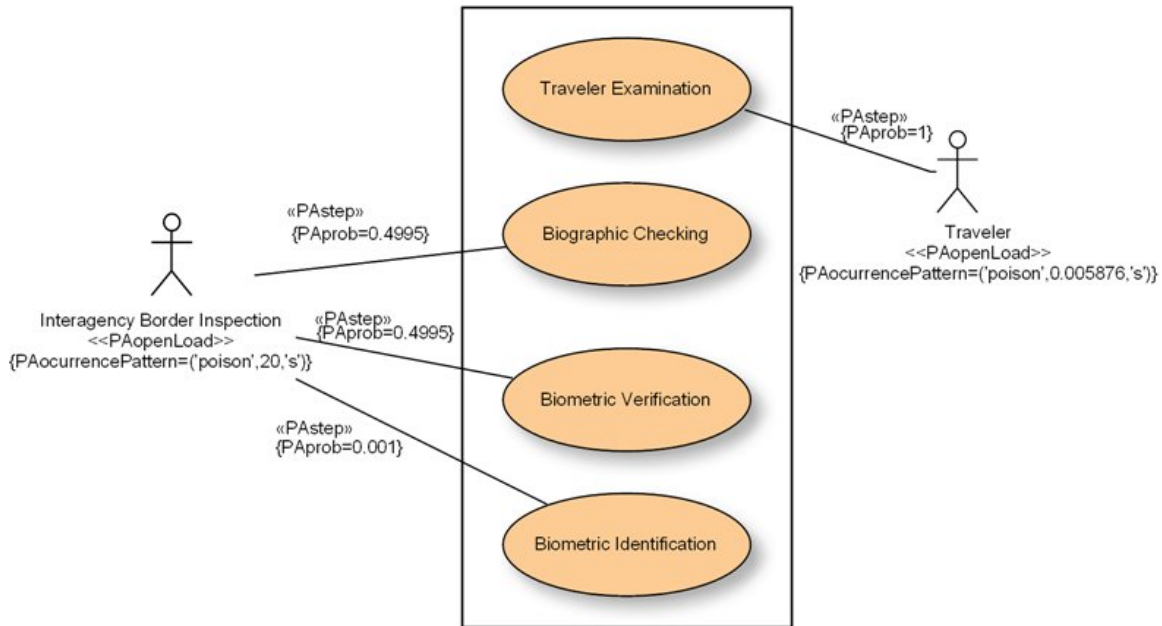


Figure 12 Use Case Diagram for the Biometric-based border Inspection System. Adapted from [4]

The *biographic checking* process initiates when the primary officer sends the basic traveler's information, first and last name, birth-date and MRTD number to the Traveler's Name Server (TNS) and the consular consolidated database (CCD) in order to obtain a consolidated information about the traveler's immigration status. The TNS retrieves the traveler's information from current immigration status and criminal violations, and the CCD retrieves the picture and consular information related to visa applications, approvals, refusals and the biometric identifiers captured during the process of MRTD issuance.

The *biometric verification* starts when the traveler's live fingerprints (right slap, left slap, and thumbs slap) are captured by the fingerprint scanner and a face image is taken by a digital camera. The system checks the quality of the collected biometric traits. When the quality of the collected biometric traits is poor, the system will request to repeat the acquisition of the biometric traits. The collected biometric traits are matched against the traveler's biometric templates in the Traveler Biometric System (TBS). Matching scores for fingerprint and face are generated. The system answers that the traveler is who he/she claims to be if the final match score is above a threshold. If the system decides that the traveler is not who he or she claims to be, then the primary officer sends the traveler to the secondary inspection booth.

The *biometric identification* process is activated when the secondary inspection officer requests a search in a consolidated watchlist with the traveler's face image captured at the primary inspection booth while performing an in-depth interview to the traveler. The traveler biometric database (TB DB) retrieves the top 50 identities after a complete match, and ranks the retrieved templates.

3.2.2 Sequence Diagram

This type of diagram presents in detail the interactions between system components that were outlined by the use cases. These interactions are represented by messages. [23]

Using the performance scenarios identified in the use case diagrams which have the greatest impact on performance and the dynamics of those use cases through the sequence diagrams, we are able to start building the system performance model.

In "Appendix A: Detailed Uses Cases and Sequence Diagrams", Figure 44 to Figure 48 present the dynamics of the use cases Traveler Examination, Biographic Checking, Biometric Verification and Biometric Identification with resource demands for each scenario step.

3.2.3 Deployment Diagram

This type of diagram illustrates both the physical configuration of the system and the software allocation on the hardware device, which provide essential information to the performance model. We are able to estimate resource demands of interactions by identifying the hardware devices executing functionalities provided by the software system and their corresponding service rate.

Figure 13 presents the deployment diagram for the border inspection system under study. The system contains several primary inspection booths. As described by [4], each inspection booth has a workstation (POE workstation) that accesses the traveler name server (TNS) and the traveler biometric server (TBS) through a WAN connection. In our baseline configuration, the POE workstation stores the Public Key Directory (PKD) as a dedicated resource. In section 3.4.1, we will analyze and report the performance results when the PKD is located outside the POE workstation and communicates with it through a network link.

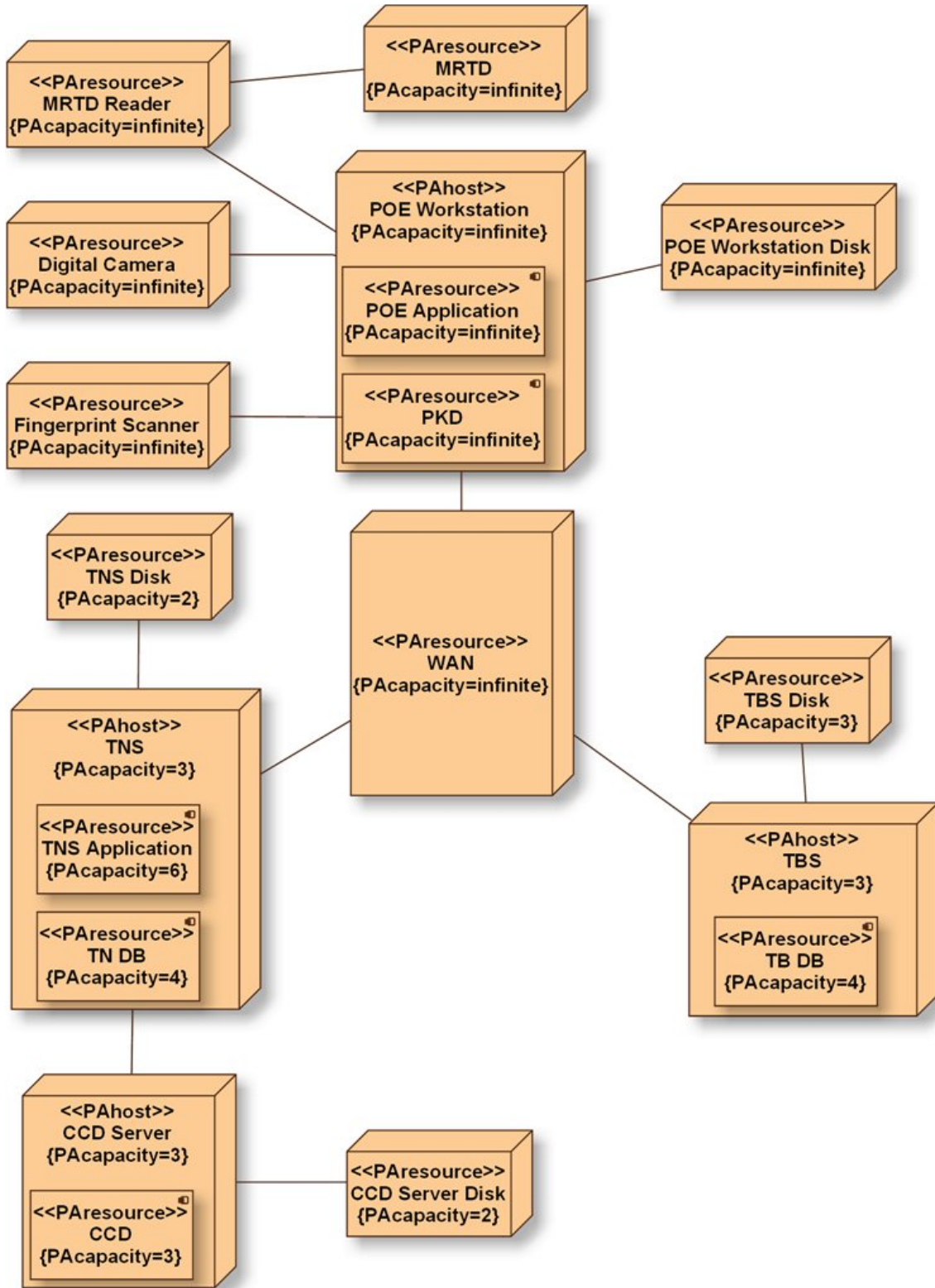


Figure 13 Deployment Diagram for the Biometric-based border Inspection System. Adapted from [4]

3.3 Performance Model Parameters

The U.S. Customs and Border Protection (CBP) monitors arriving passenger processing times in 16 major international airports [5]. CBP data also includes the number of passengers per hour and the average number of open booths.

Figure 14 to Figure 18 present CBP arrival information from December 2007 at one of the terminals of the Dulles International Airport. This distribution was used in our experiments presented later. The data set was normalized to average the number of travelers per booth per hour. Ensuing distribution is presented in Figure 15. For modeling purposes, we assume that the passengers are evenly distributed among the available booths. December of 2007 was chosen because the new regulation of 10 fingerprint acquisition was implemented at that time. Dulles International Airport was the first airport to put into action this security upgrade.

Figure 14 shows the average number of passengers arriving per booth per second throughout the day. It is notorious that there are certain times when no travelers arrive. There are hours where the load is 75% higher than the smallest load. A significant variation (error bars) is observed during the night hours up to 2 am. Columns with no error bars represent a single value. With this kind of pattern in arriving of the travelers at border inspection points, it seems that modeling the system with open workloads presents a more realistic approach than the one previously reported with closed workloads [3].

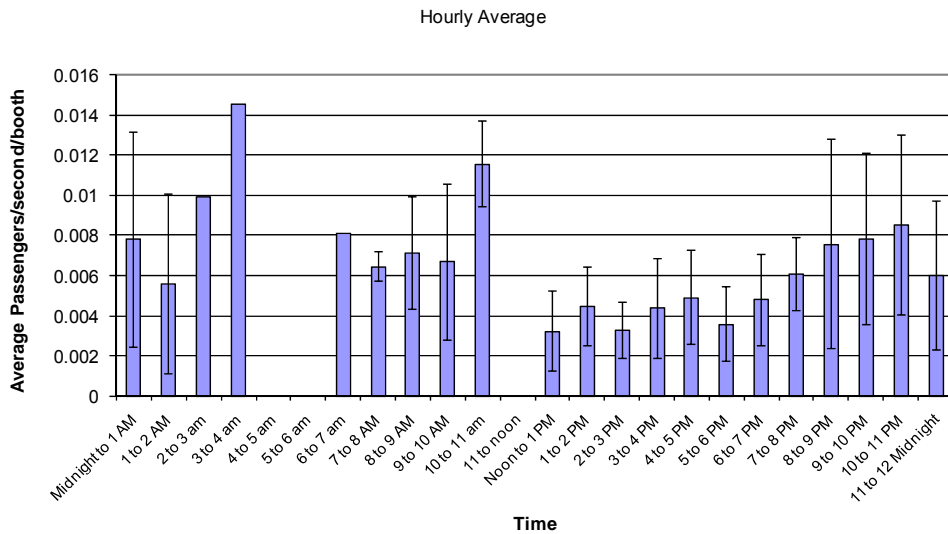


Figure 14 Average Passengers in the month of December 2007

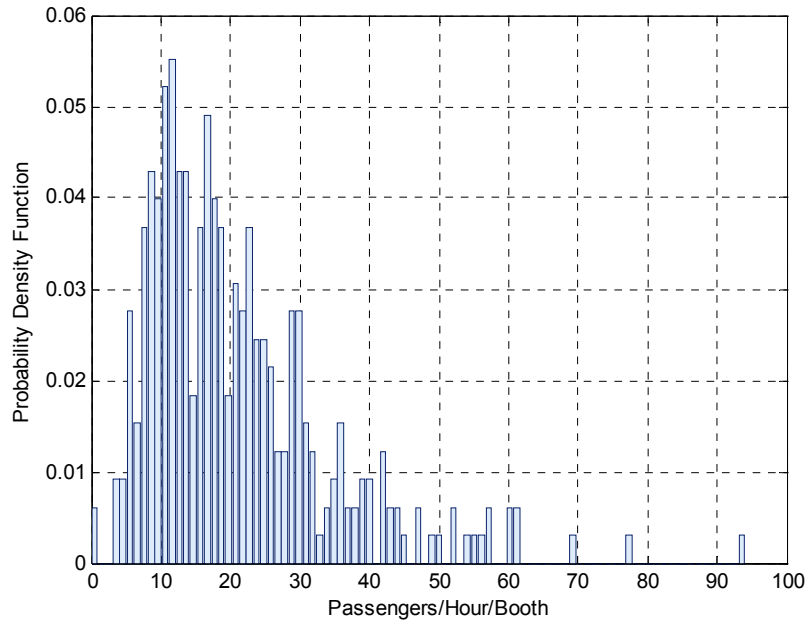


Figure 15 Passenger Distribution at the Terminal Booths

Figure 16 presents the average daily waiting time. We see that the average waiting time range is usually between 30 and 50 minutes, and that the upper-bound, the busiest day, is on Sunday and the lower bound of the system occurs at the middle of the week where the waiting time is nearly 30 minutes. With this information we can validate our models because the graph provides reference values for waiting times. Processing time is not recorded by CBP and therefore it cannot be directly validated.

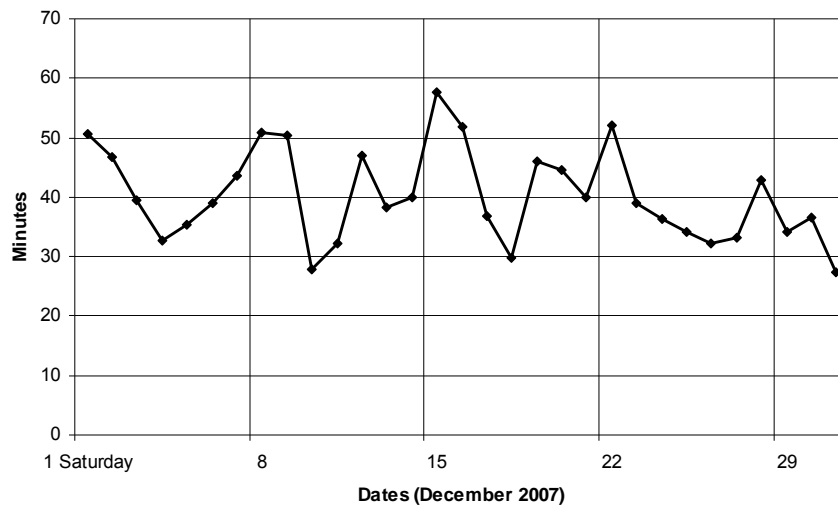


Figure 16 Waiting time in December 2007

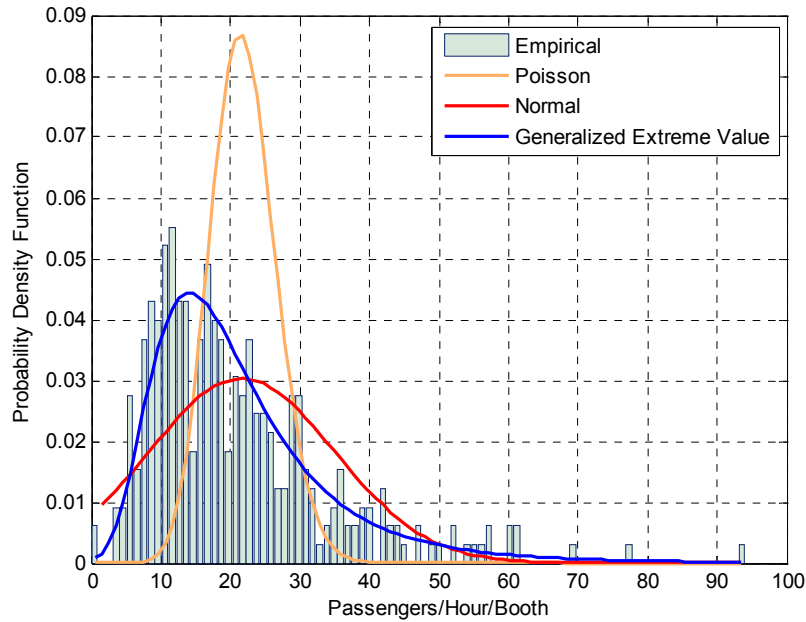


Figure 17 PDF for Dulles Data and Approximated Distributions

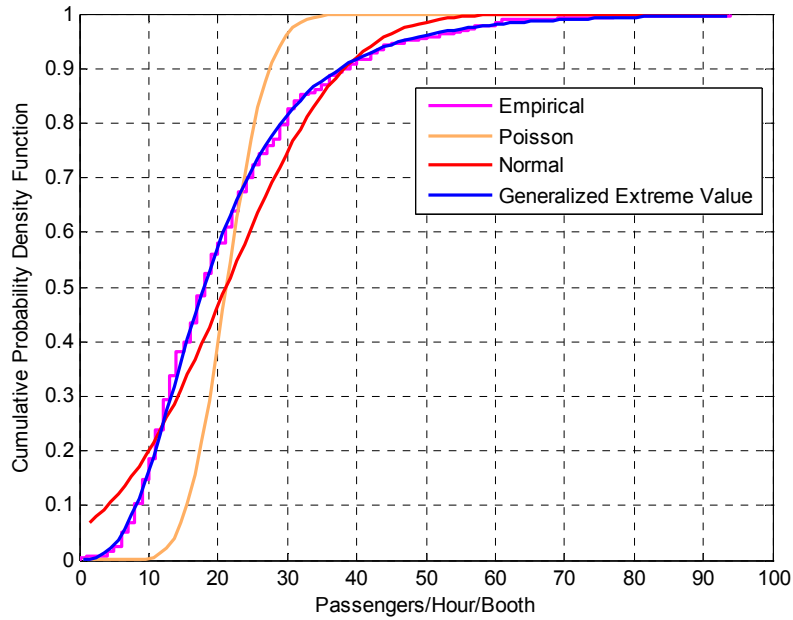


Figure 18 Cumulative PDF for Dulles Data and Approximated Distributions

The distribution of the Dulles Airport data is shown in Figure 15 through Figure 18. Figure 15 presents the probability density function (PDF) of the observed arrival rates (passengers per hour per booth). The mean value for the dataset is 21 passengers/hour/booth and the standard deviation is 13. Figure 17 compares the dataset's

PDF with fitted Poisson, Normal and Generalized Extreme Value (GEV) distributions. Figure 18 presents the cumulative PDF compared to the fitted distributions. The normal distribution is presented as a reference but it yields negative values, which makes it inappropriate for the modeling arrival rates in this situation where the mean is close to zero.

The Poisson distribution may serve as an approximation to the actual distributions when only the mean estimates are available [20], condition that is assumed throughout the models. For example, refer to the access control scenario of Figure 5. Poisson distribution showed to be a poor fit for the observed passenger distribution, as can be inferred from Figure 17. The goodness of fit was evaluated by the Chi-square test. The test determined that the best approximation for the actual distribution is a Generalized Extreme Value (GEV) distribution with $k = 0.15$, $\sigma = 8.46$, and $\mu = 14.80$. The test P value for GEV fit is 0.39 (thus the confidence in the GEV fit is 61%). The actual distribution was not used due to limitations in the modeling tool that did not allow for custom distributions to be implemented.

Using the annotations from the use case diagram and the probability density function for traveler arrival, we are able to calculate user workloads on different performance scenarios based on the probabilities associated with the scenarios and with the related use cases. The probabilities associated with the scenarios are given by <<PAprob>>, which states the probability that a user invokes the linked use case. Therefore, an open workload can be described as a product between the arrival distribution and the probability associated with relevant use cases.

3.3.1 Execution environment

Table 19 and Table 20 in Appendix B: Performance Parameters, present data assumptions that are extended from technical reports of analogous systems [27][28][29]. Table 21 to Table 24 in Appendix B: Performance Parameters present the estimated time of each <<PAstep>> for the sequence diagrams based on the data assumptions and calculated values.

Delays introduced by inspector officers are assumed to be exponentially distributed [4]. For example the time spent by the officer to do both the physical reviewing process of traveler documents and interview process averages 20 seconds. The computing

process time of each server is calculated by focusing on complex operations to be executed such as cryptographic operations, query databases, and process requests among others. The average time for transferring data through communication channel is calculated by dividing the size of the data to be transferred by the throughput of the channel [4][57]. Similarly, the average time for input/output of files from a disk is estimated by knowing the size of the data to be transferred and the throughput of the storing device.

Some important service demands are presented here. For a global picture of the model and a more complete set of parameters see Appendix B: Performance Parameters.

Based on the analysis presented in [4], performance parameters for Biometric Verification and Identification, Traveler Examination, and Biographic Checking are defined as follows.

Performance parameters related to Biometric verification and identification:

- *captureFingerprint*{PAdemand=('pred', 'mean', (15, 's'))} refers to the necessary time to process the captured left slap, right slap and thumbs slap of the traveler.
- *checkQuality*{PAdemand=('pred', 'mean', (0.005, 's'))} is the required time to check the quality of the biometric trait.
- *capture faceImage*{PAdemand=('pred', 'mean', (5, 's'))} is the required time to process an online traveler's picture using a digital camera.
- *send-store&matchBiometrics*{PAextOp=('pred', 'mean', (0.0329, 's'))} is the necessary time to send the biometric data collected in the POE Workstation from the traveler (fingerprint scans + face image) to the TBS, using the WAN connecting them: $0.0329 = (50+20)KB/16.6 \text{ Megabits/s}$
- *read-writeBiometricData*{PAextOp=('pred', 'mean', (0.00637, 's'))} is the time for reading a previously stored face image file (20KB) in the TBS disk and for writing the biometric data collected from the traveler (fingerprint scans + face image) on it:
 $0.00637 = 5.93 \text{ ms} + ((50 + 20 + 20) \text{ KB} / 200\text{MB/s})$
- *MatchFingerPrint* {PAdemand=('asmd', 'mean', (0.005, 's'))} is the time, the TBS spends comparing how similar the traveler's fingerprints with its corresponding fingerprints-probe are.
- *MatchFace* {PAdemand=('asmd', 'mean', (0.05, 's'))} is the time, the TBS spends matching the traveler's face image with its corresponding face-probe.
- *compareWatchlist*{PAdemand=('asmd', 'mean', (5, 's')) PAextOp=('pred', 'mean', (97.66218, 's'))} is the time that the TBS disk spends by matching the traveler's face image with the set of 1 million face images in the biometric watchlist, it is assumed to be 5 seconds; additionally the TBS disk consumes time reading the entire watchlist. Consequently, the actual time required to perform the

computation depends on the size of the watchlist: $5.93\text{ms} + (1,000,000 \times 20\text{KB} / 200\text{MB/s}) = 97.66218\text{s}$

- *rankIdentities* {PAdemand=('asmd', 'mean', (0.005, 's'))} is the required time by the TBS to rank the match scores from comparing the traveler's face image with the set of 1 million face images in the biometric watchlist.

Performance parameters related to Traveler examination:

- *return MRTDData* {PAdemand=('pred', 'mean', (X, 's')) PAextOp=('pred', 'mean', (Y, 's'))}, the PAdemand specifies the time required by the MRTD Reader to read the data stored in the MRTD where $X = \text{MRTD}_{\text{size}} \text{ bytes} / 424 \text{ kilobits/s}$. PAextOp refers to the time the MRTD transfers its data to the MRTD reader. $Y = \text{MRTD}_{\text{size}} \text{ bytes} / 848 \text{ kilobits/s}$
- *validateMRTD* {PAdemand=('pred', 'mean', (X, 's'))} refers to the necessary time to transfer the MRTD data from the MRTD reader to the POE Workstation through a 12 Mbits/s USB link. $X = \text{MRTD}_{\text{size}} \text{ bytes} / 12 \text{ MB/s}$
- *return PKCertificate* {PAextOp=('pred', 'mean', (Y, 's'))} specifies the necessary time to read Public Key Certificates from the Disk of the POE Workstation. The actual reading time depends on the number of certificates to be read ($N=1$ or 2): $Y = 6.5\text{ms} + (N * \text{PKC}_{\text{size}} \text{ KB} / 130\text{MB/s})$
- *validate(MRTD_DS)* {PAdemand=('pred', 'mean', (X, 's'))} refers to the required time to validate the authenticity of the digital signature on the MRTD. Therefore, we need to compute a hash function (SHA-1) of the MRTD data, and to verify the authenticity of the digital signature by applying the RSA algorithm using the Public Key of the MRTD signer (2048 bits) [27]. The time to perform the operation depends on the amount of data stored in the MRTD: $X = t[\text{SHA}_1(\text{MRTD}_{\text{size}} \text{ bytes})] + t[\text{RSA}(2048\text{bits})\text{-verify}(20\text{bytes})]$

Performance parameters related to Biographic checking:

- *findTravelerData* {PAextOp=('pred', 'mean', (0.0118, 's'))} is the necessary time to exchange the TNS traveler's biographic and lookup information and a picture of him/her. Our assumptions about the average size of the traveler's data are: 5 KB for the biographic data and 20 KB for the picture. $25\text{KB} / 16.6 \text{ Megabits/s} = 0.0118\text{s}$
- *return consolidateInfo* {PAextOp=('pred', 'mean', (0.00595, 's'))} refers to the required time to retrieve from the TNS disk traveler's biographic and lookup data: $5.93 \text{ ms} + (5 \text{ KB} / 200\text{MB/s}) = 0.00595\text{s}$
- *return picture* {PAextOp=('pred', 'mean', (0.006, 's'))} is the time a traveler's picture from the CCD server disk is retrieved: $5.93\text{ms} + (20\text{KB} / 200\text{MB/s}) = 0.006\text{s}$
- *reviewDocs* {PAdemand=('asmd', 'mean', (300, 's'))} is the time a secondary inspection officer comprehensively reviews the traveler's documents and belongings and to question him/her.
- *processInspectionData* {PAdemand=('asmd', 'mean', (5, 's'))} refers to the time a secondary inspection officer needs to decide if authorizing the traveler to enter the country based on the outcome of checks.

Following the work of [4] in closed workloads, Figure 19 and Figure 20 below depict the analytic layered queueing network model for the open workload biometric-based border inspection configured with the previous parameter settings. Those parameters are basically service demands on entries and activities, generated workload with reference tasks, and amount of replicated devices.

Figure 19 presents an overview of the system configuration by showing tasks, hardware devices and the flow of messages between entries and entries requiring usage of devices, through request arcs. On the other hand, Figure 20 offers a more detail explanation of tasks internal functionality by their corresponding activities and their communications which are done by configurations using sequences, fork-join, etc.

In our system, travelers are either genuine or impostors. A genuine traveler is a person whose identity recognition is based on the authenticity of his/her traveler documents and his/her biometric traits. On the other hand, an impostor traveler is an individual who has forged his/her traveler documents and/or biometric traits with the purpose of entering the country using a different identity circumventing the system. The implications on the system performance are increasing average waiting times due to in depth search of watch lists, verification of authenticity of traveler's documents, and detailed interviewing.

The *OrFork* in the *beginPrimaryInspection* entry, was used in order to represent the option of possible impostors present at the booth, given a predefined prior probability associated with them. Although, the system triggers an alarm when detecting suspect impostors, it also does that to some genuine travelers, for this reason at the secondary inspection a mixture flow of both genuine and impostor arrives. Furthermore, there are some impostors that the system is unable to detect succeeding in circumventing it, while other genuine travelers are deported or in custody until more evidence is available so their identities are reinforced.

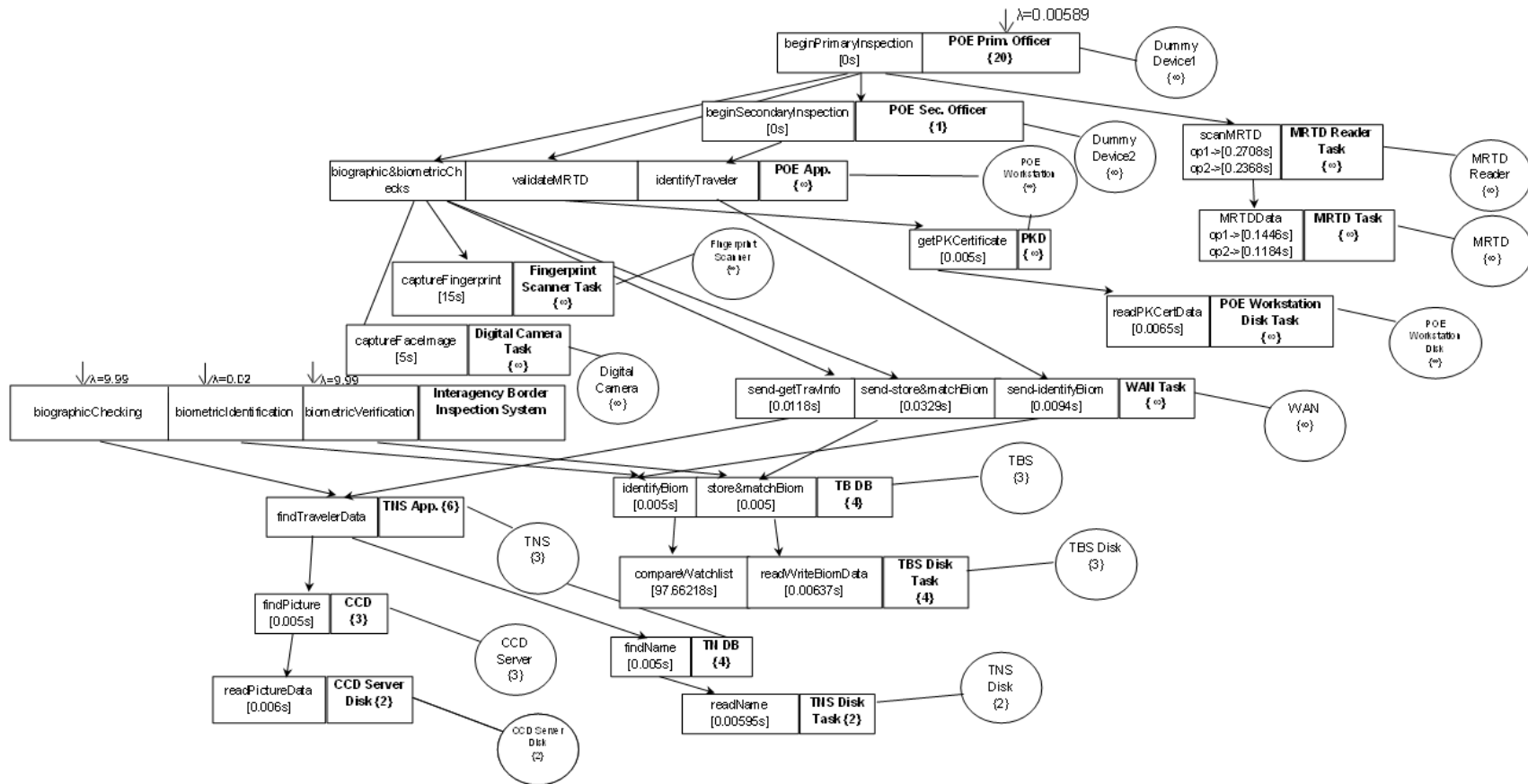


Figure 19 LQN Model for the biometric-based Border Inspection using open workloads

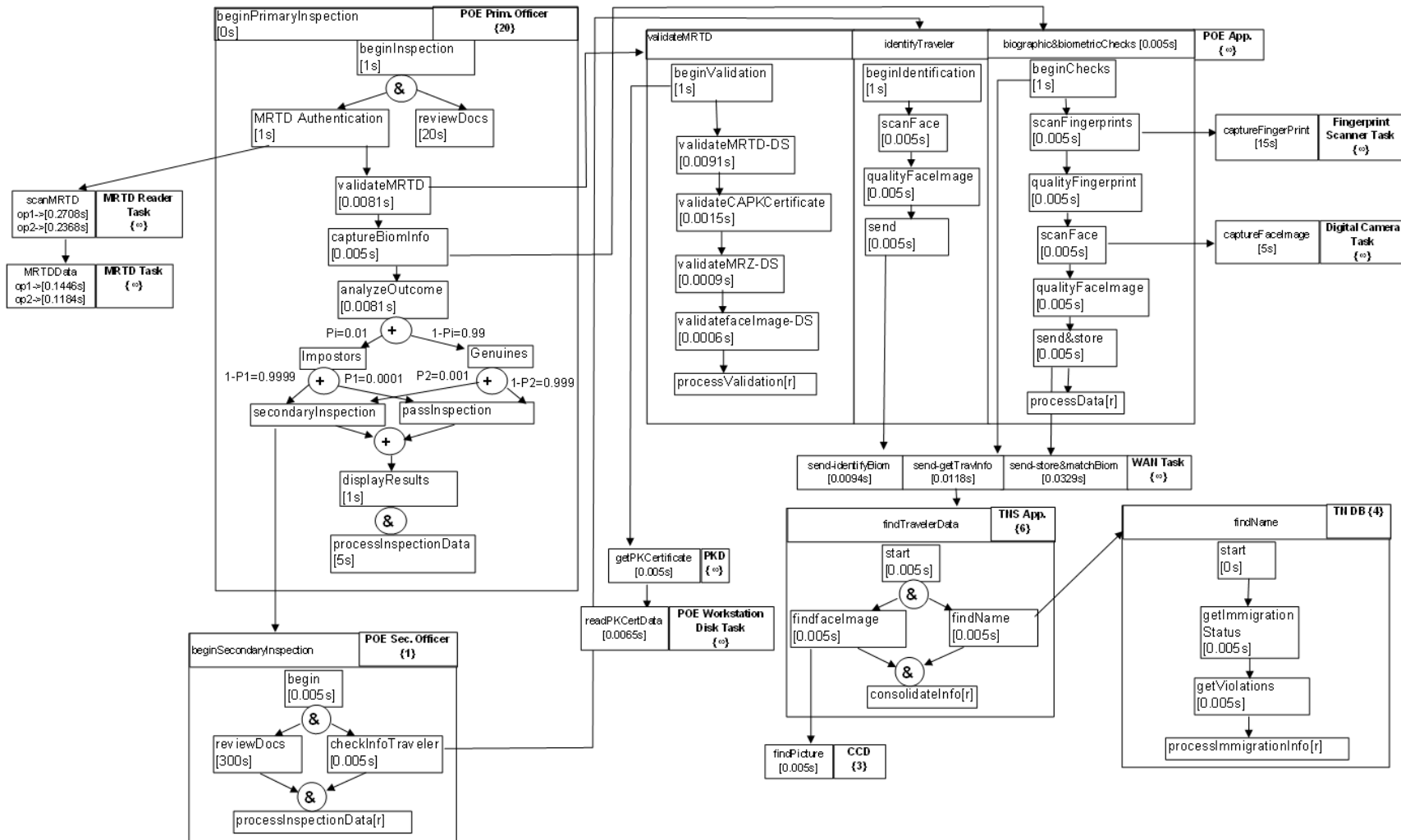


Figure 20 Expanded LQN Model for the primary and secondary inspection options using open workloads

3.4 Experiments and Results

3.4.1 Public Key Certificate

The purpose of this experiment is to examine the waiting time results from accessing public key certificates of authorities issuing MRTDs. The public key certificate may be stored locally in the MRTD, in a dedicated database or in a shared database. The validation of certificates of authorities is required in order to verify the authenticity of travelers' documents at the inspection booths.

The LQN models are solved in order to obtain their analytical solution and establish the most suitable configuration that provides the best performance. As mentioned above, three cases were explored:

1. MRTDs public key certificate is stored inside the traveler document.
2. Dedicated PKD: public key certificate is stored in a database that is access locally from each officer workstation.
3. Shared PKD: public key certificate is stored in a database that is access by a pool of airports in a region. In order to express the intensity of requests on the PKD, four kind of PKI population were studied: 1 airport, 40 airports, 80 airports and 160 airports.

The entries and activities related to configuration 3, basically the inclusion of a PKI system that communicates with the POE workstation, are taken from [4] and depicted in Figure 22 . Specific modifications related with inclusion of impostor population, false match rate, and detailed verification and identification modes on the biometric modules are based on the same description provided by Figure 20. Parameters of the relevant LQN entries are shown in Table 3 for the various scenarios.

Table 3 LQN parameterization for Public Key Certificate Experiment

LQN entries	Option		
	<i>MRTD</i>	<i>Dedicated PKD</i>	<i>Shared PKD</i>
readMRTDdata	0.1446s	0.1184s	0.1184s
scanMRTD	0.2708s	0.2368s	0.2368s
validateMRTD	0.0093s	0.0082s	0.0082s
PKCertificate	0.0065s	0.0065s	0.0065s
verify	0.0044s	0.0044s	0.0044s
send-getPKCert	-	-	0.0017s

Figure 21 presents the result of our model analysis, the average waiting time at the inspection booths undertaken by travelers. The architectural options: MRTD, dedicated PKD and shared PKD with 40 airports present similar average waiting times in the inspection point. The number of travelers that can be inspected every time is limited by the available primary inspection points. They are able to handle population in the ranged 1 to 1600 with acceptable times, with a larger population the system becomes saturated and the waiting time goes to infinite. At those points, the only solution would be to increase the number of available booths, since this is the bottleneck in the system. The shared PKD within 80 and 160 airports become sooner saturated since more airports are requesting services to the shared PKD.

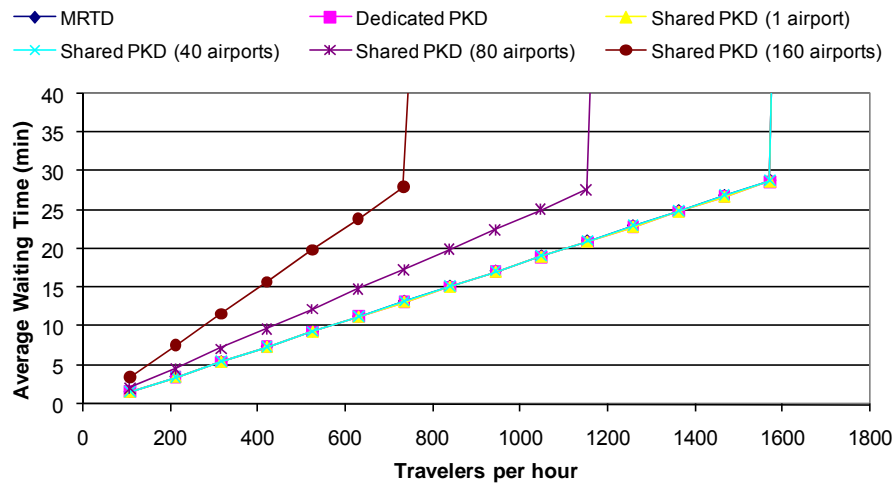


Figure 21 Average waiting time using different architectural options

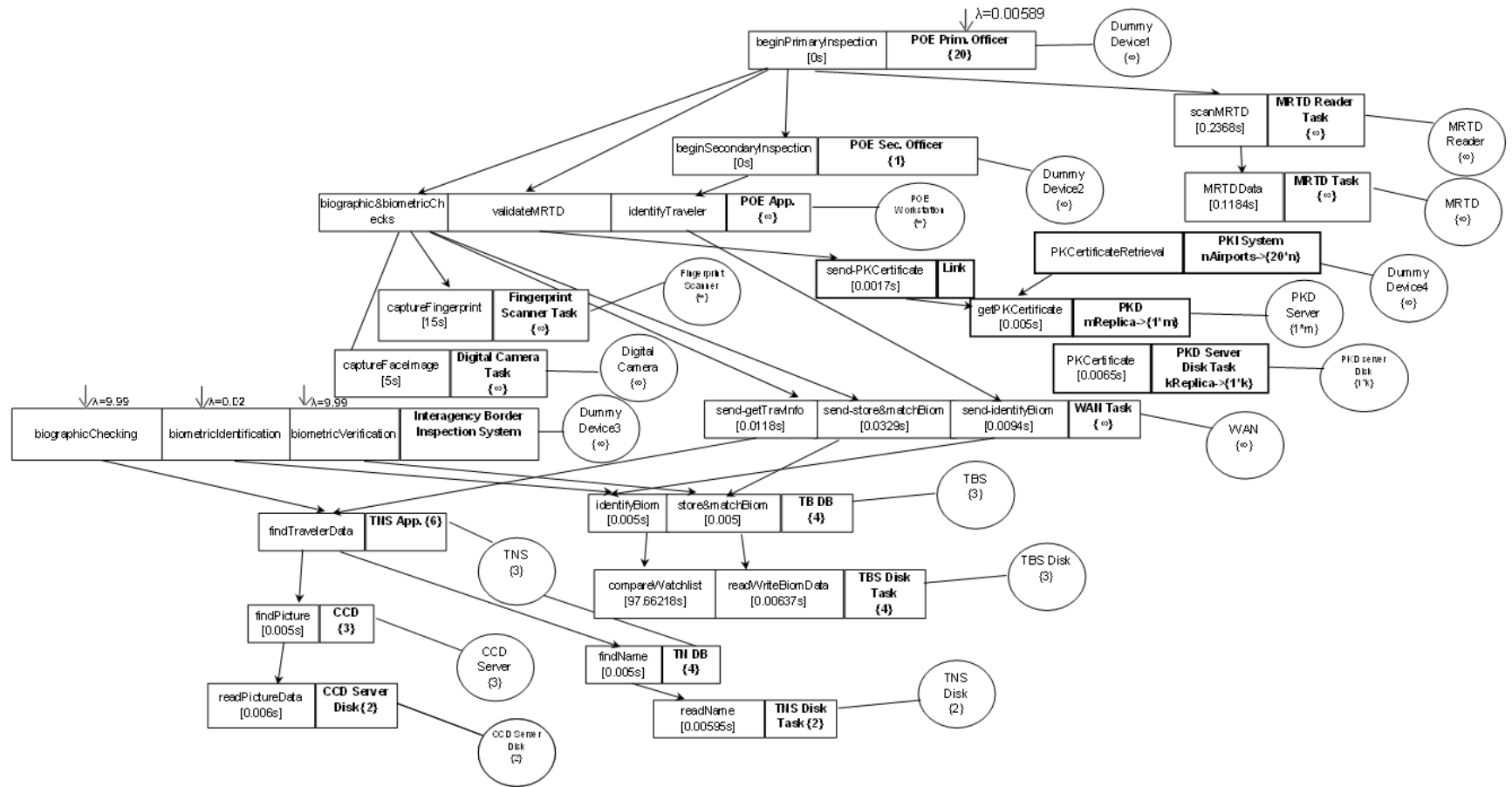


Figure 22 LQN model for the border inspection system using the shared-PKD option

3.4.2 Screening Policies

The purpose of this experiment is to examine the effects of increasing the number of subjects in the watch lists to the inspection and waiting time at the inspection booths. Watch lists are deployed at border crossing to facilitate quick screening of passengers against the list of persons of interest. It is assumed that the watchlists are partitioned and distributed along the TBS disks, in order to increase the system throughput. The considered size values for the watchlist and their corresponding parameterization values in the LQN entries are presented in Table 4.

Figure 23 presents the inspection time experienced by travelers at the booth inspections when the system performs the identification search over a consolidated watchlist. The average inspection time is less than six minutes in the secondary line, when the number of suspects within the list is between 1,000 – 1,000,000, which is comparable with reported results in the literature [25]. On the other hand, the system with the current configuration is unable to work with watchlist sizes equal or larger than to 10,000,000 since it surpasses the acceptable time, thus creating very large queues.

Figure 24 shows how traveler waiting time changes as additional high-risk subjects are included in watchlist. In the baseline scenario, we assume that the watchlist size is fixed to be 1,000,000. We can observe from the plots that increasing the watchlist size directly impacts the waiting time as the number of travelers increases.

Table 4 LQN parameterization for different screening scenarios

Watchlist size (different subjects)	LQN entries (comparison time)
1,000	0.03034 s
10,000	0.25 s
100,000	2.447 s
1,000,000	24.414 s
10,000,000	244.1465 s
50,000,000	1220.709 s
100,000,000	2441.4121 s

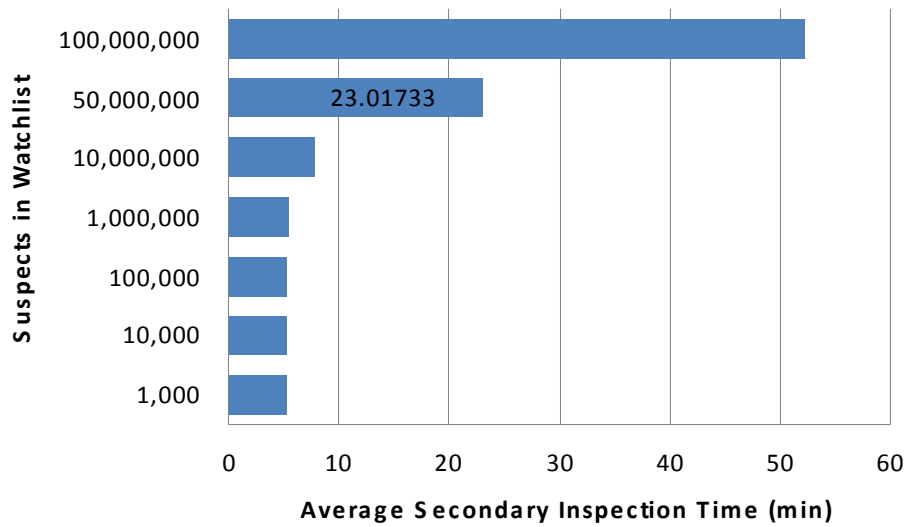


Figure 23 Average secondary inspection time for different screening scenarios

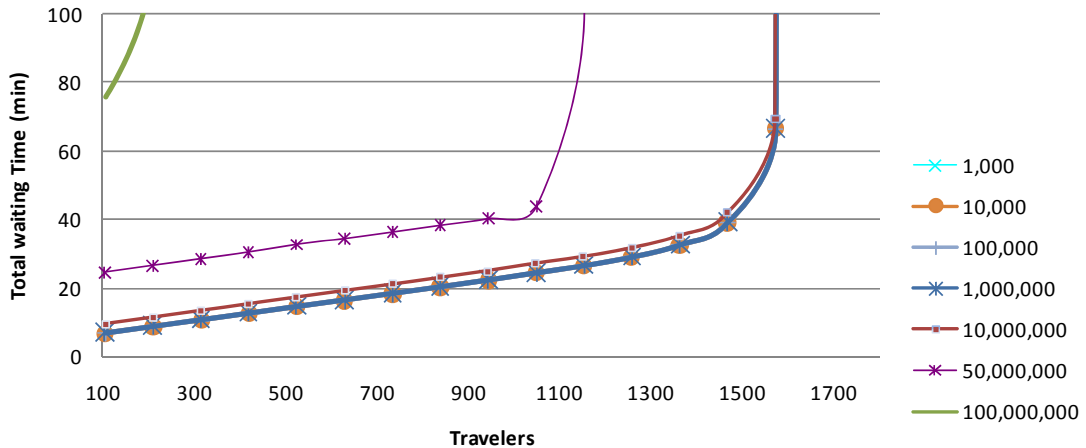


Figure 24 Total waiting time for different screening scenarios

3.4.3 Biometric False Match Rate

False match rate (FMR) is the frequency of the error incurred when deciding that two biometric strings, the input string and the template are from the same subject, while in reality they are from different subjects. It occurs when match scores from impostor are above the system’s threshold. This experiment intends to evaluate how the false match rate in the biometric system affects the performance provided by the airport inspection point. Below, we report the FMR as a p_I in the primary inspection interaction, where p_I

expresses the probability that an impostor traveler is accepted as a genuine traveler; the values considered are shown in Table 5.

Table 5 FMR variation in the LQN model (Biometric False Match Rate Experiment)

FMR	LQN entries
1%	$p_i: 0.01$
0.1%	$p_i: 0.001$
0.01%	$p_i: 0.0001$

The impact in inspection and waiting time associated with variations at the impostors prior probability is also studied, using the fixed values of $FMR=0.01\%$ and $FNMR=0.1\%$. For the sake of clarity, we relate the different levels of security threat with the impostor arrival probabilities. The values considered are shown in Table 6 and they do not correspond with actual values deployed by the Department of Homeland Security at the airports.

Table 6 LQN parameterization (Biometric False Match Rate Experiment)

P(impostor)	Alert Level Status	LQN entries
0%	green	$p_i=0$
0.01%		$p_i=0.0001$
0.1%	blue	$p_i=0.001$
0.5%	yellow	$p_i=0.005$
1%	orange	$p_i=0.01$
10%	red	$p_i=0.1$

Figure 25 represents the total waiting time experienced by travelers when the system FMR changes from 1% to 0.01%. Results of our model analysis did not present significant differences in the waiting time for the various values given to the FMR, using the fixed values of $FNMR=0.1\%$ and $p(impostor)=0.1\%$. The low impostor arrivals probability in combination with the different FMR values allows that the waiting time for secondary inspection between impostors were almost the same. In addition, the system performance is heavily affected by the travelers, so the waiting time increases as the traveler arrival increases. Even more, the system is not saturated as long as the arrival rate is under 1,400 travelers per hour.

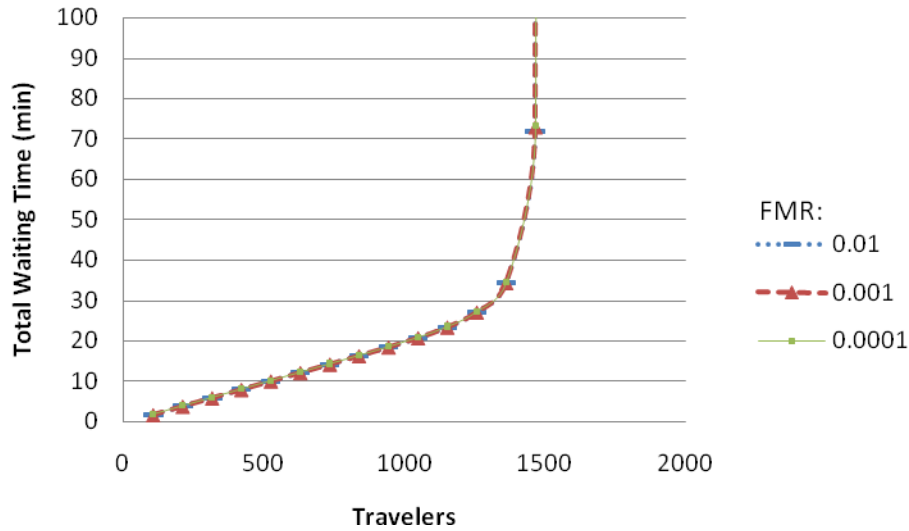


Figure 25 Total waiting time for different FMR scenarios, assuming $P(\text{impostor})=0.001$

Figure 26 presents results for different alert level status at the inspections facilities. For the baseline inspection scenario we assumed a blue alert level or high condition with the impostor prior probability of 0.1%. We perform an analysis on that parameter, assigning it the values 0%, 0.01%, 0.1%, 0.5%, 1%, 10%. These values may correspond to different levels of security threat. A 0% and 0.01% (low condition) correspond to the low risk of experiencing an impostor, so a trusted traveler population (e.g. pilots, government officers, etc) is being processed quickly. A 0.1% (guarded condition), 0.5% (elevated condition), 1% (high condition) and 10% (severe condition) correspond to the increasing risk of experiencing an impostor. From Figure 26 we can conclude that the total waiting time is sensitive to large values of impostor arrival probability. For example, the 10% rate produces saturation at the inspection booths, given that it is more probable that the alarms in the system will be activated, so the officers would have to send more people to a second inspection and the total inspection would take longer, given that a more detailed interviewing is required.

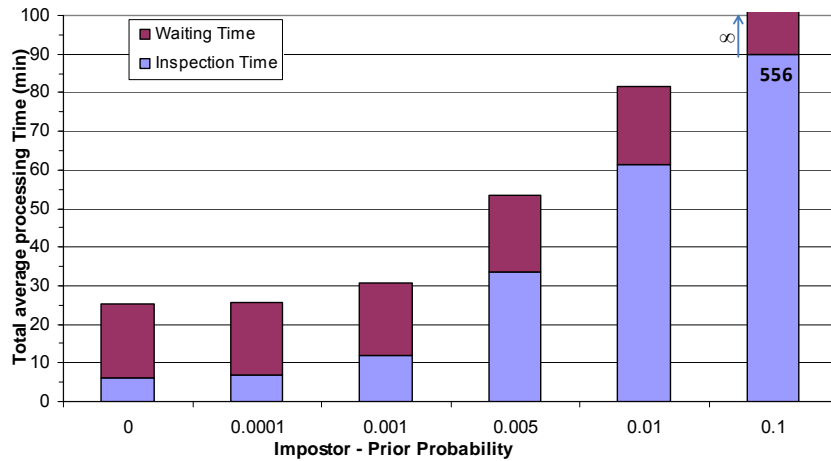


Figure 26 Total Inspection time for different impostor prior probabilities

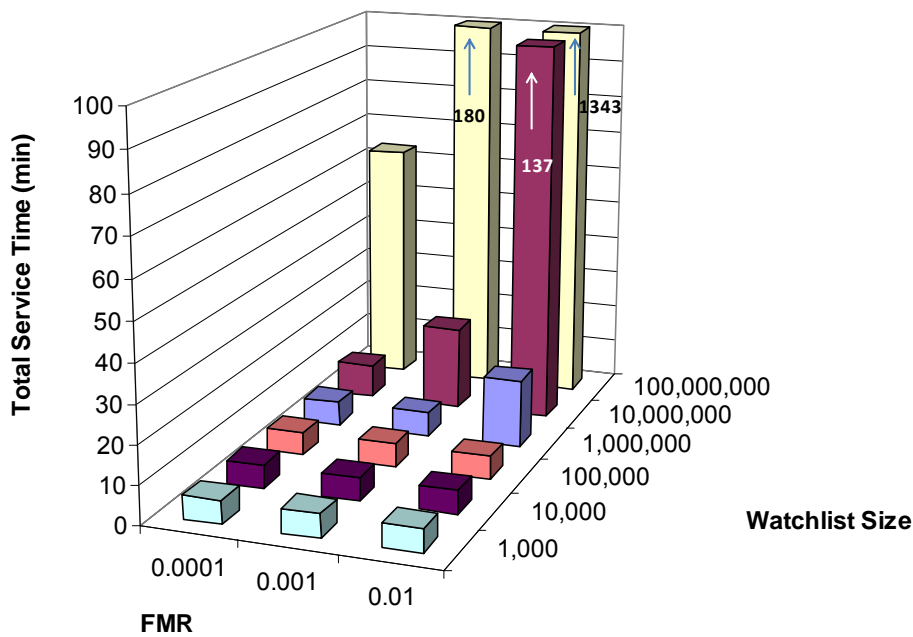


Figure 27 Total average Inspection time at different combinations of FMR and Watchlist size

Figure 27 offers total average inspection time at different combinations of FMR, and watchlist size. For a load of 1049 travelers/hour, and a fixed FNMR=0.001. The system provides the lowest inspection time with the option FMR=0.0001. Nevertheless, going from a watchlist size of 10 million to 100 million increases the total inspection service

time from less than nine minutes to nearly 64 minutes. On the contrary, the effect of watchlist size when below 100 thousand is not significant. Note that the bound imposed by the traveler population is still an issue. In conclusion, more than increasing watchlist size, reduction of the False Match Rate offers significant reductions in passenger screening time.

3.4.4 Replication of PKD

This experiment intends to evaluate the improvement in performance of the system given that we increase the number of PKD available in the shared PKD configuration. We use the LQN model for the shared-PKD option (see Figure 22) augmented with the values corresponding to the number of replicas wanted. We found in the first experiment that options for 80 and 160 airports become saturated very soon. For this reason we replicate some PKDs in order to increase the throughput. In Table 7, for 80 airports, having more than one replica decreases the utilization, thus unsaturating the system. The intermediate point is attained with 3 PKDs because a reasonable waiting time is achieved and the utilization is half providing room for further requests. For 160 airports, such point is attained with 6 PKDs. It must be noted that implementing replicas have an inherent cost because consistency needs to be guaranteed.

Table 7 Utilization

80 airports					
PKD #	Primary Inspection Time (s)	Waiting Time (m)	Utilization		
			PKD DB	PKD Proc.	PKD Disk
1	58.47	25	0.9995	0.4346	0.5650
2	44.49	19	0.8756	0.3807	0.4949
3	44.44	19	0.5856	0.2546	0.3310
4	44.43	19	0.4393	0.1910	0.2483
5	44.43	19	0.3514	0.1528	0.1986
6	45.43	19	0.2929	0.1273	0.1655
160 airports					
PKD #	Primary Inspection Time (s)	Waiting Time	Utilization		
			PKD DB	PKD Proc.	PKD Disk
1	94.02	∞	0.9999	0.4347	0.5652
2	58.34	25	1.0000	0.4348	0.5652
3	47.37	20	0.9999	0.4348	0.5652
4	44.46	19	0.8755	0.3807	0.4949
5	44.44	19	0.7014	0.3049	0.3964
6	44.43	19	0.5846	0.2542	0.3304

3.5 Analysis

Our results from the performance experiments provide insights about total waiting time that a traveler would experience during his/her authentication process at the inspection booths. At the same time, we are able to assess different architectural and security related options, that could present a major impact on the border inspection mission, which is to keep inadmissible travelers (e.g. forged traveler's identity, required by law enforcement agencies) from entering the country and facilitate the entrance of travelers who meet the legal requirements to entry the country, without a detriment in any of the inspection procedures. In all the performance models, we use the arrival rate from the approximated distribution (see section 3.3), which depicts the requests load that the system would experienced.

We start evaluating the average waiting time as a result of different localization of public key certificates of authorities issuing MRTDs. We considered storing the public key certificate inside the MRTD, on a dedicated PKD inside every POE workstation and in a PKD shared by the inspection points within 1 airport, 40 airports, 80 airports and 160 airports. From this experiment, we observe (see Figure 21) that the availability of primary inspection points determines the number of travelers that can be inspected every moment. The configuration that involves sharing the PKD among 80 to 160 airports, presents saturation on traveler's arrival of less than *1,200* and *1,000* per hour, respectively. These results are consistent since many requests are arriving to the single PKD in both of the configurations. On the other hand, the configurations of locally stored (MRTD), dedicated stored (PKD in every POE workstation) and shared PKD among 1 to 40 airports present similar average waiting for the traveler.

Next, we explore the impact of watchlists size on the average secondary inspection time. The average secondary inspection time is about the same in the cases where the number of entries is *1,000,000* or below. However, for watchlists size over *1,000,000* entries (see Figure 23), the average waiting time starts increasing due to a slower response from the Traveler Biometric disks (TBS-disk), which contain the watchlists. This experiment suggest that implementation of novel indexing schemes on TBS-disks may improve its response time that will translate in reduction of the secondary inspection time.

We performed a sensitivity analysis on the probability of impostors and genuine travelers arriving at the inspection booths by evaluating the average inspection time and the total waiting time. We varied the impostor probability from 0% to 10%. Our baseline scenario assumes that 0.1% of the travelers are impostors and there is one secondary inspection. We observed (see Figure 26) that for a relatively large probability of impostors (10%) among travelers, the system bottleneck becomes the secondary inspection given that more travelers are likely referred to it. This experiment suggests that replication of secondary inspections are required when the probability of impostors arriving at the border inspection start reaching 1% or above.

Another important finding is the performance implication as a result of the system's false match rate. The total waiting time in the system is about the same for the FMR's different values 0.1% to 1% (see Figure 25). Those results are consistent given the fact that impostor prior probability and false non-match rate were fixed in all the cases, therefore the secondary inspections queues are slightly influenced by small waiting time contributions from the impostors who were detected by the system.

Next, we explore the effects in the total inspection service time when combining different watchlists sizes with different false match rate values (see Figure 27). In general, the system using a $FMR=0.0001$ provides the lowest inspection time. Although, going from a watchlist size of 10 million to 100 million increases the total inspection service time approximately six times. On the contrary, the effect of watchlist size of below 100 thousand is not significant; the total inspection time is almost the same.

Finally, we analyzed the performance impact from having more than one PKD replica in the configurations that presented saturation (shared PKD among 80 to 160 airports) (see Table 7). The experiment shows reduction in the PKD utilization, up to the point where the having more than three replicas and six replicas in every case respectively did not alter the primary inspection time and waiting time.

In summary, results from the previous experiments provide insights about improvements in the inspection time that travelers would experience in the system when using different types of configurations. Changes in those configurations may result as a set up of security measures that have to be implemented in the system.

Chapter 4 : Risk Analysis

The most important goal of border inspection is to identifying “high-risk” individuals, who can put in danger a country by a terrorist attack. The selection of travelers that must be exhaustively inspected as opposed to a superficial scrutiny without increasing the threat is a key issue. Nevertheless, terrorists may have gone through border inspections without been caught and are still unknown.

One major vulnerability of border inspection system was the lack of an universal mechanism to include the names of all terrorist in watch lists. This vulnerability was exploited by Khalid al-Mihdhar and Nawaf al-Hazmi, two of the hijackers involved in the 9/11, whom even though were identified by the CIA as potential criminals by January 2001, the CIA did not request them to be watch-listed until late August 2001, when they had already being accepted in the US. [14]

Figure 28 and Figure 29 present some statistics inferred from [11] which reveal that officers at ports of entry have prohibited the entrance of thousands of individuals and detected fraudulent travel documents. The officers at ports of entry likely did not apprehend all the inadmissible aliens. How many of the inadmissible aliens and other violators evaded successfully the inspection points is estimated through a program called Compliance Examination (COMPEX) [11].

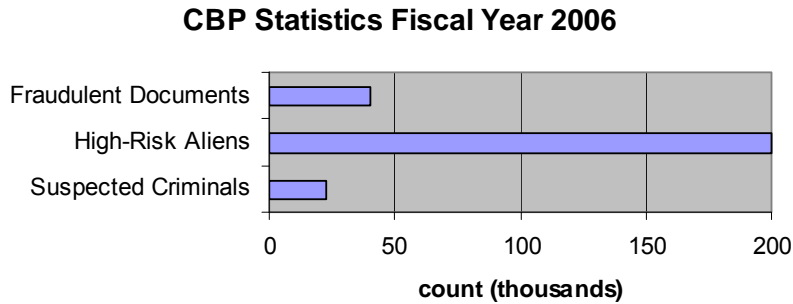


Figure 28 Identification of High-Risk aliens and other violators

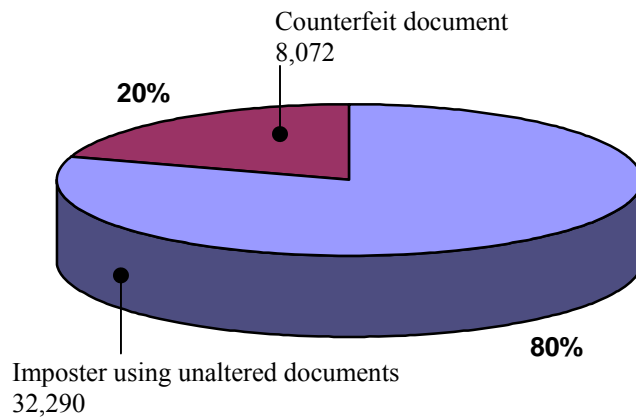


Figure 29 Fraudulent Documents

4.1 Biometric System

Since 2003, the Border Inspection System contains a biometric subsystem, which is able to work in three different modalities: identification, verification and watchlist.

Identification (Who am I?) modality is applied when the identity of a subject is unknown in advance. A biometric pattern is computed from the subject’s biometric features in order to find out the identity of that person. The entire template database is compared against the subject’s probe yielding a match score which has to be within a given threshold in order to provide the top k identities.

Verification (Am I whom I claim I am?) modality is used when the subject provides an alleged identity. The system performs a comparison between the person’s biometric query with the template that is already in the system, a score is produced and checked against a threshold.

In a watchlist, the subject does not claim an identity; therefore the biometric sample of the subject is compared with the samples present in the watchlist to detect if the subject's pattern is in it. When a subject is found to have similarities to one or more entries in the watchlist that are higher than the given threshold, the system activates an alarm and returns the list of identities from the watchlist that triggered the alarm. The system could incur in two types of errors: miss rate and false hit rate. *Miss rate* is when the system pulls out wrong identities without including the right one or a reject and the subject is present in the watchlist. *False hit rate* is when the alarm is activated by a subject that is not present in the watchlist.

Biometric signatures and their corresponding templates are different. Their similarity depends on the acquisition method, user interaction with the acquisition device, the acquisition environment, and the possible variations present in the traits due to physiological changes [30]. Some of the common reasons for biometric signal/representation variations are:

- *Inconsistency in the biometric presentation:* The signal obtained by the sensor from a biometric trait relies on both the intrinsic trait characteristic and the way the trait is offered. For example, the three-dimensional shape of a finger is mapped into the two-dimensional surface of the sensor accordingly to the pressure and contact that the finger put on the sensor surface, given that, the finger is an elastic object and the projection of the finger surface into the sensor surface is not exactly controlled, different impressions of a finger are related to each other by various transformations. Additionally, it is possible that each impression of a finger may represent a different portion of finger's surface introducing additional spurious features.
- *Irreproducible Presentation:* Biometric identifiers are susceptible to wear and tear, malfunctions, injuries and physiological development. For example, the ridge structure of a finger can change either permanently or temporarily by injuries, manual work, and accidents, among others. Hand geometry measurements might not be reproducible when the user is wearing different kinds of jewelry each time. Face recognition is affected by facial hair, makeup, accidents, external accessories that correspond to irreproducible face

representations. A person's voice changes as a result of health problems, for example a common cold, affecting a voice recognition system. These events and more contribute to dramatic variations in the biometric trait signal captured at different acquisition instants.

- *Imperfect Signal/Representational Acquisition:* The acquisition environment in real scenarios is not perfect and produces extraneous variations in the acquired biometric signal. For example, poor-quality fingerprint acquisitions are caused by a non-uniform contact with the sensor and the dryness of the skin, thin/worn-out ridges, skin disease, sweat, dirt, and humidity in the air, which lead to either spurious or missing minutiae. Different illuminations cause prominent differences in the facial appearance. The voice signal is affected by the channel bandwidth characteristics. The use of different image processing operations could perturb feature localization. High inter-class similarity (biometric traits from different subjects are similar) due to either inherent lack of distinctive information in the biometric trait or the representation used for the biometric trait is too restrictive, finally the feature extraction algorithm could introduce some measurement errors.

Variations in the biometric signal/representation lead to error rates in the biometric authentication system, since the matching module has to decide which of the following hypotheses is true:

The *null* hypothesis $H_o \Rightarrow$ the two samples do not match, the input does not come from the same subject as the template.

The *alternate* hypothesis $H_a \Rightarrow$ the two samples match, the input comes from the same subject as the template.

The two errors that a matcher can present are:

- I. False Match (FM): Deciding that the input and the template are from the same individual, while in reality they are from different people, deciding H_a when H_o is true. The frequency with which this occurs is called the False Match Rate (FMR) (3.4.4-1). An impostor can generate a high match score ($s > T$), thus causing an FM.

$$FMR = \int_{s=T}^{\infty} p(s | H_o = true) ds \tag{3.4.4-1}$$

II. False Non-Match (FNM): Deciding that two biometrics, input and template are not from the same identity, while in reality they are from the same identity; deciding H_o when H_a is true. The frequency at which this occurs is called the False Non-Match Rate (FNMR). A genuine subject can generate a low match score ($s < T$), thus causing a FNM.

$$FNMR = \int_{s=-\infty}^T p(s | H_a = true) ds \tag{3.4.4-2}$$

Unfortunately, the non-match score distribution typically overlaps the match score distribution. Subsequently, it is not feasible to choose a threshold for which $FMR=0$ and $FNMR=0$. A threshold must be defined based on the security level that the system must offer. The lower chance of False Match implies a higher security level. This will cause some level of inconvenience for genuine users.

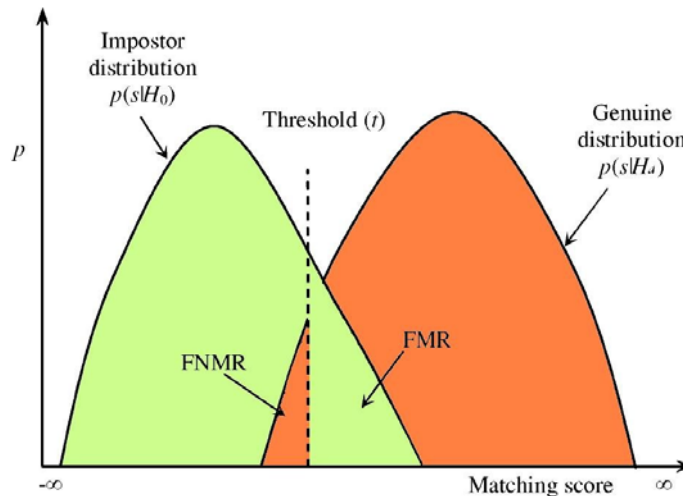


Figure 30 Common Biometric Error rates

In the border inspection system, frequent flyers will usually produce lower error rates since they are more habituated to interact with the system. Furthermore, studies have shown that there are differences in recognizing different persons with respect to their biometric identifiers. Among them, some will produce higher error rates than others

[2][7]. Another important fact is the fraud rate, i.e., the percentage of the population that is attempting to defraud the system.

Finding the most appropriate FNMR and FMR in our models is one of our major goals, since we can evaluate performance scenarios where impostors are not detected during border inspection process, and genuine travelers are inconveniently rejected.

4.1.1 Receiver Operator Characteristic (ROC) Curve

A ROC curve is one method of assessing classification performance in two dimensions of binary class classifiers; the graph plots the FMR against FNMR [8]. The set of points in the curve come from variations in the setting of a classifier's threshold. The ROC curve provides an estimate of the predictive characteristics of a classifier. Figure 31 presents a ROC curve with different application scenarios.

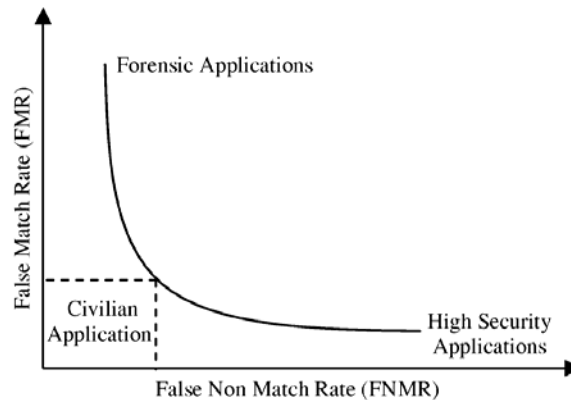


Figure 31 FMR vs. FNMR ROC curve. Source [30]

4.1.2 Vulnerabilities of Biometric Systems

Even though, biometric authentication systems have numerous advantages over traditional authentication systems [17], they are not exempt of weak points that can be exploited by attacks [21].

A biometric system is depicted in Figure 32, where modules may be vulnerable to the following attacks:

1. Fake biometrics are provided to the sensor as input, for example synthetic finger, face makeup, or a face disguise.
2. Resubmitting previously stored digitized biometrics signals: In this mode of attack, a recorded signal is replayed to the system, bypassing the sensor.

Examples include the presentation of an old copy of a fingerprint image or the presentation of a previously recorded audio signal.

3. The feature set obtained at the feature extraction module is chosen by the attacker via a Trojan horse in the module.

4. Fraudulent modification of stored templates: the attacker may modify one or more templates in the database with the intention of accepting a fraudulent subject or rejecting subjects associated with the corrupted template. Biometric systems storing the template in smartcards are susceptible to this kind of attack.

5. Replacement of features obtained from the input signal with a different fraudulent feature set. For example, the communication channel between the feature extraction module and the matcher module can be interfered in order to alter specific packets.

6. Attack to the communication channel between the stored templates and the matcher by intercepting and modifying the information traveling on it, this type of attack can be replayed in other moment for gain access.

7. Alteration of the matcher module in order to emit pre-established match scores.

8. Alteration of the final decision is the most critical attack, given that the system becomes useless, even though its recognition framework possesses excellent performance characteristics.

These attacks can be minimized by incorporating liveness detection of the entity originating the input signal (attack at points 1,2), encrypting the communication channels (attack at points 5,6) , placing the matcher and the database in a secure location (attack at points 4,5,7) using cryptography at the decision module (attack 8), mutual authentication between each pair of modules, hardening of database server, among others [32].

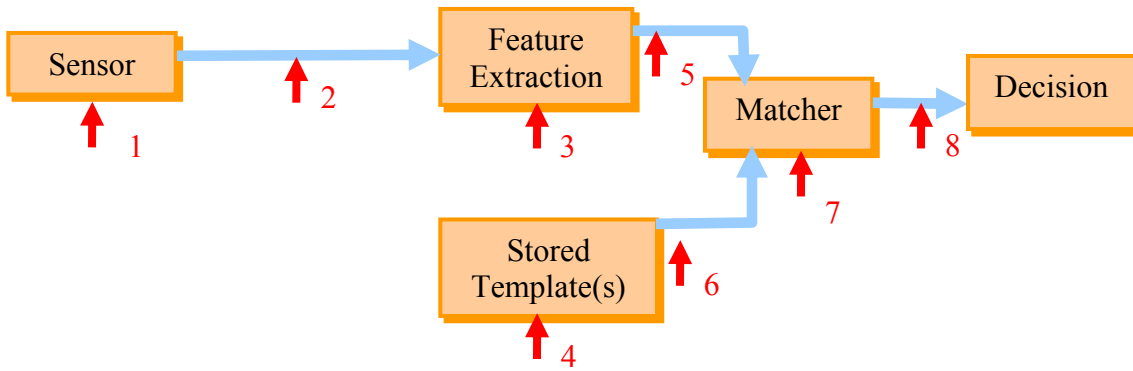


Figure 32 possible Attack points in a biometric authentication system, adapted from [21].

4.2 Risk Model

Predicting the number of high-risk aliens who may bypass the border security inspection system is currently one of the most challenging problems. Some approaches have been developed in order to increase the accuracy in the biometric modules [25]. They formulate the identification as a game theory problem where the Government uses some parameters in order to maximize the detection probability and the impostors want to minimize the detection probability by providing an image of poor quality. Classifying regular travelers as high-risk ones implies the application of several verification procedures, therefore increasing the inspection time which translates in fewer officers available. On the other hand, misclassifying a high-risk alien as a low-risk traveler carries the risk of immigration failure and its cost implications in the society.

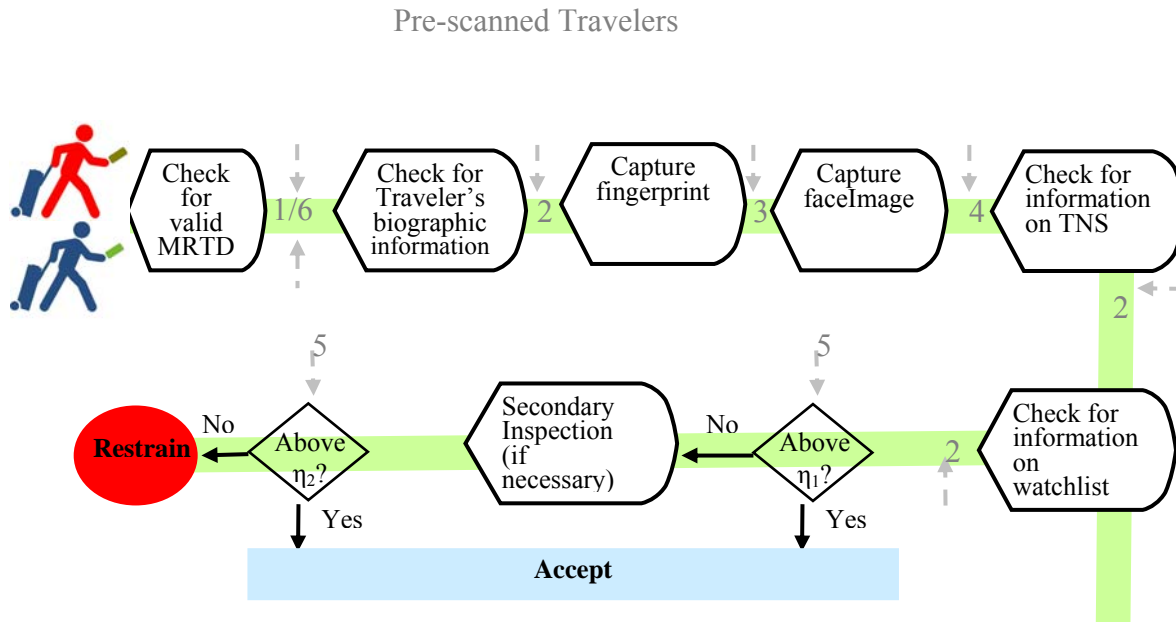
Let us define the set of inspection points as I . I is divided into two disjoint sets, I_1 is the set of primary inspection points which are mandatory for all travelers and set I_2 which is the set of secondary inspection points. If threshold η_1 at I_1 , is not met, an inspector must perform an exhaustive examination so the traveler could be cleared out or detained based on a second threshold η_2 .

We assume that every inspection point in I has a sequence of steps, each intended to check one particular out of the N known possible treats, such as: construction of a fraudulent MRTD using materials from legitimate documents, alteration of MRTD by either substituting the photograph, or altering both the text in the visible zone and machine readable zone (MRZ). Furthermore, there are impersonation attacks, where a

high-risk traveler alters his biometric traits, like face (by using plastic surgery, masks or makeup), fingerprint (by cuts, bruises, among others) in a way that it resembles a low risk traveler.

Figure 33 presents possible technical hazards in the biometric-based border inspection system where each device in the system could introduce some level of inaccuracy in the inspection process. Security attacks related to Trojan horses, software component replacement, and communication channels among others are depicted in Figure 32.

Table 8 presents the severity of each component for the border inspection system that can occur. We are assuming that methods of liveness detection are used in the fingerprint sensors, there is an officer guiding the biometric acquisition, the communication channels are well protected against eavesdropping, replay attacks, man in the middle and brute force attacks.



Technical Threats:

1. MRTD scanner overloading, e.g. interference, power surges, input flooding
2. DB compromise, e.g. DB with modified entries or identity associations changed, furthermore, without recent updates (see Table 9)
3. Fingerprint scanner disrupted service, high failure to acquire.
4. Digital camera spoof by disguise, facial hair and accessories.
5. Threshold bad configuration, e.g. illegitimate traveler is likely to result in a successful match decision.
6. POE workstation is not able to connect to PKI directory, high access time to PKI directory.

Figure 33 Technical threats in the biometric-based border inspection system

Table 8 Severity Analysis for a biometric-based border inspection system

Triggered hazard	Cause	Fault	Criticality
Mismatch between stored template in TNS database and probe in MRTD	System fails to inform officer that the traveler needs a second inspection	Failure to match the impostor traveler, who is a terrorist.	High
Mismatch between stored template in TBS disk and MRTD	System fails to inform officer that the traveler is required by law enforcement	Failure to detain a bad alien	High

Update in TBS disk has not be executed, the watchlist is not the most recently one.	System fails to inform officer that the traveler is required by law enforcement	Failure to detain a bad alien	High
Incorrect search of inconsistencies in biographic data within traveler's documents.	Officers at primary and/or secondary inspection point fail to detect inconsistencies in traveler's documents	Failure to detect fraudulent documents	High
Mismatch between captured facial features and the stored template	System fails to inform officer the real traveler's identity	Failure to identify an impostor alien	High
Data inaccuracies, omitted and inactivated fields, duplicate records in watchlists [13]	System fails to inform the officer the current suspects in government watchlists	Failure to detain a traveler required by law	High
Poor quality in fingerprints due to genetics, hard labor or deliberately done. 5% of the general public 10% of those on the watchlist [25].	System is not able to collect fingerprints and perform verification and identification.	Failure to authenticate a traveler in the system.	High
Mismatch between stored template in TNS database and probe in MRTD	System flags to the officer that the traveler needs a second inspection, when he is who claims to be.	Failure to no match the genuine traveler	Low

Table 9 presents the approximate update time among the databases used in real border inspection systems [18]. The maximum lapse of time for executing updates is monthly, which could create a security hole in the system.

Table 9 Databases at port of entry and their approximate update time [18]

Database	Update time
CCD	Days / weeks
TBS (IDENT)	Days/weeks/months
TNS (APIS)	Hours
PKD	Undefined

Our risk model considers misclassifications due to the limitations of biometric identification system. For that reason, we evaluate different levels of security related to misclassification costs over the classification algorithms that may be deployed at the border inspection and how their results would impact the waiting time in the system.

4.2.1 Cost Curves

Cost Curves present graphically the expected cost of misclassification of classifiers along the range of their operating points. Additionally, they can show confidence intervals and statistical significance when comparing their error rates, which cannot be done easily by using ROC curves [8].

The coordinate axis in Cost curves are represented by probability cost function $PC(+)$ in x obtained as in (4.2.1-1) and by the normalized expected misclassification cost in y obtained as in (4.2.1-2). Since we are dealing with misclassification, let us denote class “+” as impostor and class “-” as genuine. The cost of misclassifying a genuine user as an impostor is denoted by $C(+|-)$ and $C(-|+)$ denotes the cost of incorrectly classifying an impostor as a genuine. Probabilities of an user being an impostor or a genuine user, at the deployment of the system, are represented by $p(+)$ and $p(-)$ respectively.

$$PC(+) = \frac{p(+)*C(-|+)}{p(+)*C(-|+) + p(-)*C(+|-)} \quad (4.2.1-1)$$

$$Norm(E[Cost]) = (FMR - FNMR) * PC(+) + FNMR \quad (4.2.1-2)$$

An example of a cost curve including possible regions is depicted in Figure 34. Trivial classifiers such as ones that either classify all the users as genuine or classify all the users as impostors are characterized by line connecting (0,0) to (1,1) and line

connecting (1,0) to (0,1) respectively. Points above these lines correspond to cases where the performance of a classifier is worse than the trivial classifiers. Finally, best and worst cases are represented by line (0,0) to (0,1) and line (0,1) to (1,1) respectively. Recall that the best case occurs when the classifier correctly classifies the users and the worst case occurs when the classifier misclassifies all the users.

We are interested in the region where classifiers perform better than trivial ones, then the range of normalized expected cost that suffices our model evaluation is (0,0.5) along the probability cost $PC(+)$ in (0,1).

Cost curves are constructed by drawing lines between points (0, FNMR) and (1, FMR) for all possible values obtained by moving the threshold in the classifier. All the intersection points from left to right are connected so the lower envelope of the cost curve is created. Every line in the cost curve corresponds to a point (FNMR, FMR) in the ROC curve given that they have a bidirectional point/line duality.

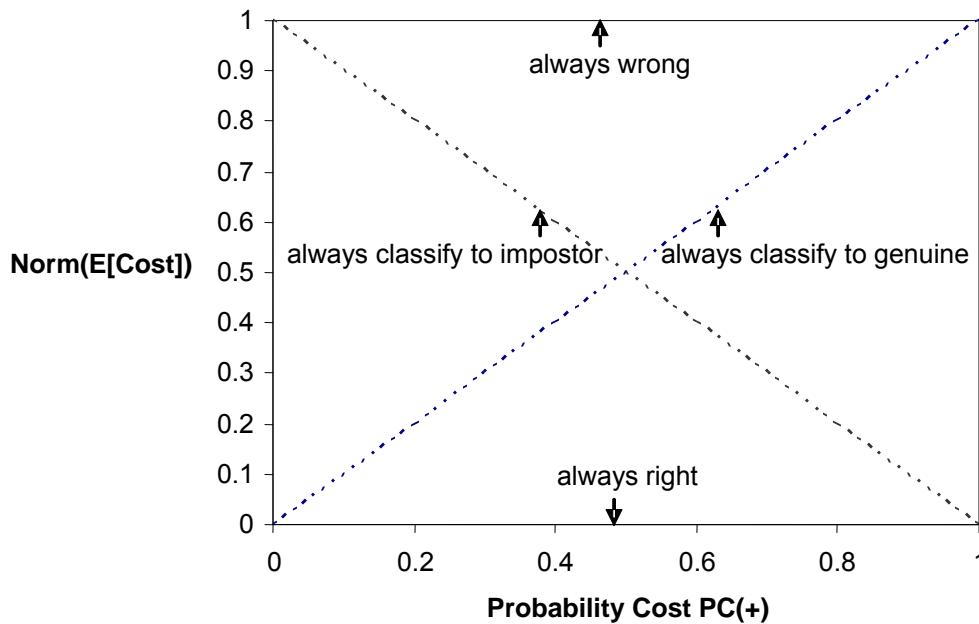


Figure 34 Possible regions in a cost curve

A cost curve analysis provides the foundations for selecting the model under which the overall misclassification cost is minimum, taking into account the misclassification cost ratio defined as (4.2.1-3). The probability cost function $PC(+)$ can be rewritten as

(4.2.1-4). Misclassification costs can influence the tendency for deciding on models and model parameters considered the most appropriate. This type of analysis is incorporated in our risk model, so we can use the performance of fingerprint and face matchers publicly available.

$$\mu = \frac{C(+|-)}{C(-|+)} \quad (4.2.1-3)$$

$$PC(+)= \frac{p(+)}{p(+)+p(-)*\mu} \quad (4.2.1-4)$$

Martonosi et al [50] studied the effectiveness of terrorist detection by prescreening systems at the airports before travelers get into the airplane. Their system may be viewed as an analogy to our border inspection system. Following their ideas, probabilities about the flow of impostors in the border inspection system are defined in Table 10.

Table 10 Definition of Probabilities associated with impostors at the primary and/or secondary inspection points

$P(hr,+)$	Prior probability that an actual impostor is classified as high risk (hr) visitor.
$P(select 2^{nd} Insp,lr)$	Fraction of low risk (lr) visitors that are selected for secondary inspection
$P(detain 1^{st} Insp,+)$	Conditional probability that an impostor is detected in the primary inspection
$P(detain 2^{nd} Insp,+)$	Conditional probability that an impostor is detected in the secondary inspection.

We combine our cost curve analysis with the corresponding risk resulting from the product of the severity level with posterior probability related to impostors and genuine travelers:

$$P(lr \cap unselect 2^{nd} Insp) = (1 - P(hr,+)) \cdot (1 - P(select 2^{nd} Insp | lr)) \quad (4.2.1-5)$$

$$= P(lr,+)\cdot(1 - P(select 2^{nd} Insp | lr))$$

$$P(+ \cap check 2^{nd} Insp) = P(hr,+)+P(lr,+)\cdot P(select 2^{nd} Insp | lr) \quad (4.2.1-6)$$

$$Risk = \left\{ [1 - P(detain | 1^{st} Insp,+)] \cdot P(lr \cap unselect 2^{nd} Insp) + [1 - P(detain | 2^{nd} Insp,+)] \cdot P(+ \cap check 2^{nd} Insp) \right\} \cdot Severity \quad (4.2.1-7)$$

Given that $P(\text{pass} | 1^{\text{st}} \text{ Insp},+) = 1 - P(\text{detain} | 1^{\text{st}} \text{ Insp},+)$ and, similarly, $P(\text{pass} | 2^{\text{nd}} \text{ Insp},+) = 1 - P(\text{detain} | 2^{\text{nd}} \text{ Insp},+)$, equation (4.2.1-7) can be simplified as:

$$\text{Risk} = \left\{ P(\text{pass} | 1^{\text{st}} \text{ Insp},+) \cdot P(\text{lr} \cap \text{unselect} | 2^{\text{nd}} \text{ Insp}) + P(\text{pass} | 2^{\text{nd}} \text{ Insp},+) \cdot P(+ \cap \text{check} | 2^{\text{nd}} \text{ Insp}) \right\} \cdot \text{Severity} \quad (4.2.1-8)$$

The cost of misclassifying a genuine user as an impostor can be rewritten as

$$C(+|-) = \left\{ P(\text{detain} | 1^{\text{st}} \text{ Insp},-) \cdot P(\text{lr}|- \cap \text{unselect} | 2^{\text{nd}} \text{ Insp}) + P(\text{detain} | 2^{\text{nd}} \text{ Insp},-) \cdot P(- \cap \text{check} | 2^{\text{nd}} \text{ Insp}) \right\} \cdot 1 \quad (4.2.1-9)$$

Also, the cost of incorrectly classifying an impostor as a genuine, $C(-|+)$, is:

$$C(-|+) = \left\{ P(\text{pass} | 1^{\text{st}} \text{ Insp},+) \cdot P(\text{lr} \cap \text{unselect} | 2^{\text{nd}} \text{ Insp}) + P(\text{pass} | 2^{\text{nd}} \text{ Insp},+) \cdot P(+ \cap \text{check} | 2^{\text{nd}} \text{ Insp}) \right\} \cdot 100 \quad (4.2.1-10)$$

We considered the methodology shown in Table 11 for determining the overall system performance according to the required level of risk of the system. In section 4.2.2 we analyzed the cost curves at the light of the performance results obtained from section 3.4.

Table 11 Proposed Methodology combining Risk and Performance for a Software System

<p>Construct the LQN model from the Use case diagram, deployment diagram, sequence diagram and performance annotations.</p> <p>For each scenario</p> <p style="padding-left: 40px;">For each value of the system threshold</p> <p style="padding-left: 80px;">Calculate FNMR and FMR</p> <p style="padding-left: 40px;">End</p> <p style="padding-left: 40px;">Construct the Cost curve</p> <p style="padding-left: 40px;">Calculate cost ratio based on equations (4.2.1-9)(4.2.1-10)</p> <p style="padding-left: 40px;">For each misclassification cost ratio</p> <p style="padding-left: 80px;">Calculate Probability Cost and Expected Cost</p>

<p>End</p> <p>End</p> <p>Find the best scenario where Expected Cost is the lowest</p> <p>Select the corresponding FNMR and FMR from the previous step</p> <p>Map those thresholds in the performance models</p> <p>Calculate overall system performance</p>

4.2.2 Cost Curves for Face Recognition and Fingerprint Recognition

Classification of travelers as impostors (high-risk aliens) or genuine (low-risk aliens), during the authentication process in the border inspection system as a result of fixed system's thresholds, have different types of security risk costs. It is highly undesirable to misclassify a high-risk traveler as low-risk since high-risk individuals are getting into the country defeating the main goal of the border security system. On the other hand, it is undesirable to misclassify a low-risk traveler as a high-risk one since we are overloading the system with comprehensive traveler checks that may cause the detriment of the system responsiveness. Our research goal is to demonstrate that varying the system thresholds induce by low risk costs at the border inspections. Under a given range of probability cost and cost ratios, we are able to assess system responsiveness as it relates to waiting time and inspection time.

Following our methodology (see Table 11), we utilized the publicly available performance matching rates of face and fingerprint recognition algorithms reported in Face Recognition Vendor Test 2006 [9] and Fingerprint Vendor Test 2003 [10], respectively. We created the cost curves associated with every classification algorithm in each modality (face and fingerprint) by drawing lines between the points (0, FNMR) and (1, FMR) for all the available pairs (FMR, FNMR) in the Receiver Operator Characteristic (ROC) curve of each algorithm. All the intersection points from left to right are connected so the lower envelope of the cost curve is created. Additionally, we defined a set of probabilities and misclassification costs for every scenario that considers the different alert level status in the system (see Table 12). Finally, we cross-relate the information from the cost curves (e.g. FMR, FNMR, normalized expected cost) with our performance models in order to estimate the impact to the traveler's waiting time.

Figure 35 shows the recognition performance results from state-of-the-art computer algorithms and human face recognition using a large-scale experiment setup in the Face Recognition Vendor Test (FRVT) 2006 [9]. The dataset consisted of a pool of single frontal facial faces across illumination changes, where half of these pairs were match pairs (images from the same person) and half were non-match pairs (images from different people). The experiment matched face images taken under controlled illumination against face images under uncontrolled illumination. The seven computer algorithms used were: Viisage (V-norm), Tsinghua U. (Ts2-norm), SAIT (ST-norm), Neven Vision (NV1-norm), Identix (Idx1-norm), Cognitec (Cog1-1to1), and Sagem (SG1-1to1). Among these algorithms, the Tsinghua U. (Ts2-norm) algorithm performed better than humans, Viisage (V-norm) and SAIT (ST-norm) algorithms had a similar performance.

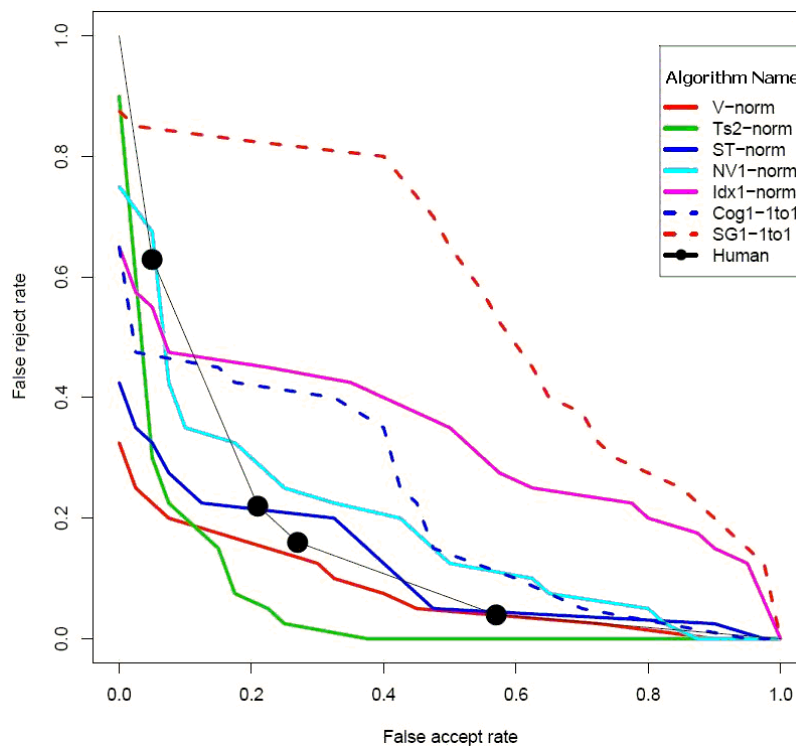


Figure 35 ROC curve for performance matching of 7 algorithms for Face images. Source [9]

Figure 36 shows the recognition performance results from single-finger flat and slap in a medium scale test in the Fingerprint Vendor Technology Evaluation (FpVTE) 2003 [10]. Images used in the experiments were obtained from livescan devices. Comparison of a single dataset against itself was performed; the dataset was formed from 10,000

images, where 5,000 images were single-finger flats and the remaining images were single-finger segmented slaps. Among the eighteen algorithms used in the experiment, Nec and Cogent algorithms had the best recognition performance.

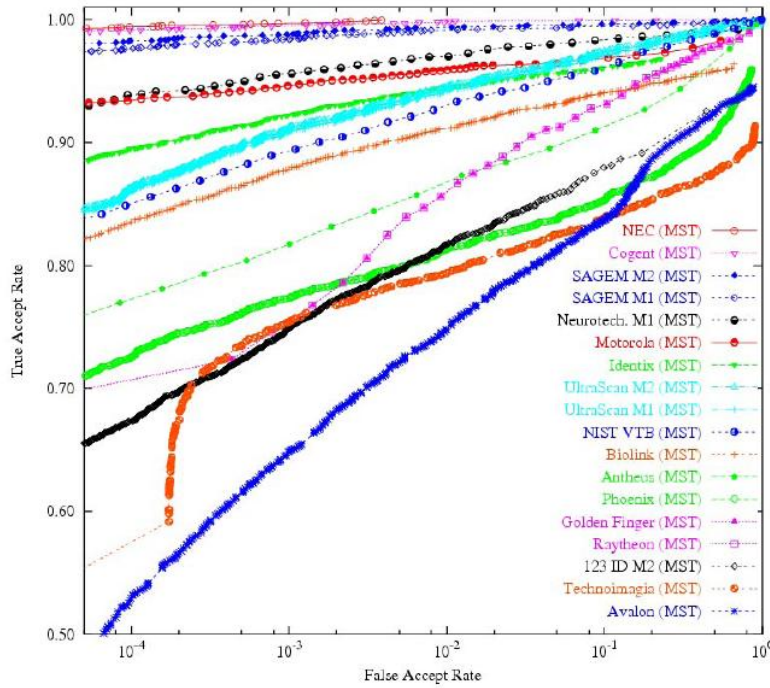


Figure 36 ROC curve for performance matching of 18 algorithms for fingerprints. Source [10]

In Table 12, we defined for every alert level status in the system (severe, guarded and low) class distributions, $p(+)$ and $p(-)$, that may be deployed. We estimate the probability cost (PC) values for the four cost ratios μ (1:10, 1:100, 1:1000, and 1:10000).

Table 12 Assumed probabilities for the risk scenarios

Probabilities	$\mu = \frac{C(+ -)}{C(- +)}$	$PC = \frac{p(+)}{p(+)+p(-)*\mu}$	Alert Level Status
$p(+)=0.01$ $p(-)=0.99$	0.1	0.091743	Severe condition
	0.01	0.502513	
	0.001	0.909918	
	0.0001	0.990197	
$p(+)=0.001$ $p(-)=0.999$	0.1	0.009911	Guarded condition
	0.01	0.090992	
	0.001	0.500250	
	0.0001	0.909174	
$p(+)=0.0001$	0.1	0.000999	Low condition
	0.01	0.009902	

$p(-) = 0.9999$	0.001	0.090917	
	0.0001	0.500025	

Figure 38 to Figure 40 present the corresponding cost curves for the face modality for every alert level status that a border inspection could experience by a given probability of the impostor population. When the ratio cost is 1:10 and with the assumed impostors probabilities (0.01, 0.001, and 0.0001), the Probability Cost approaches zero no matter which alert level is deployed. Furthermore, the Probability Cost decreases (moves towards the right direction) as long as the impostor prior probability decreases.

Table 13 presents the analysis of cost curves for face matching algorithms, within the range values of 0.01, 0.001, and 0.0001 impostor probability. It is observed that face modality does not perform satisfactory in the severe and guarded conditions at ratio cost 1:1,000. The total waiting time experienced by the travelers at the border inspection system is high (∞), making it infeasible to perform recognition since there is saturation at the second inspection. This scenario may not be adequate for the border inspection system.

Table 13 Cross-relation between cost curves for Face modality and the performance models

$p(+)$	$\mu = \frac{C(+ -)}{C(- +)}$	$PC = \frac{p(+)}{p(+)+p(-)*\mu}$	FNMR	FMR	$Norm(E[Cost])$	Total waiting time (min)
0.01	0.01	0.502513	0.175	0.073	0.123743674	∞
	0.001	0.909918	0.367	0.003	0.035789848	∞
0.001	0.01	0.090992	0.00152	0.322	0.030681116	29.81656
	0.001	0.500250	0.175	0.073	0.1239745	∞
	0.0001	0.909174	0.367	0.003	0.036060664	∞
0.0001	0.001	0.090917	0.00152	0.322	0.03065708	28.08652
	0.0001	0.500025	0.175	0.073	0.12399745	∞

Recalling the performance experiments $p(+)=0.01$ and $FMR=0.0001$ and $FNMR=0.001$ indicates a waiting time near twenty minutes. This is shown below in Figure 37. On the other hand, with the misclassification cost ratio of 1:100 and 1:1000 and an arrival rate of 1049 travelers/hour, we found that the combination of all these parameters in the system produce an infinite waiting time since the secondary inspection is unable to attend the high load created by genuine travelers incorrectly classified.

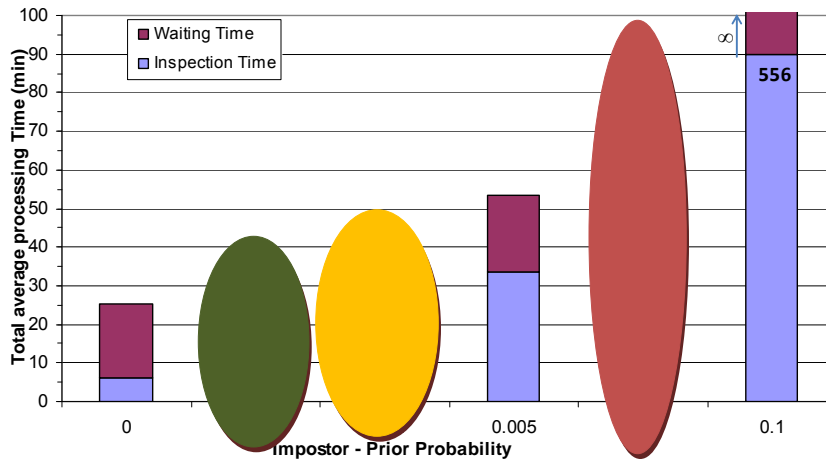


Figure 37 Total Inspection time for different impostor prior probabilities using FMR=0.0001 and FNMR=0.001

From Figure 38 to Figure 40 we find that V-norm and Ts2-norm are the two face recognition algorithms that offer the best performance. We are able to distinguish the range of Probability Cost where the algorithm V-norm outperforms Ts2-norm. The operating range for the lower envelope of Ts2-norm algorithm is $0.45 < PC(+) \leq 0.99$ and it does not include the lowest extreme cost ratio (1:10) in any of the three configurations. V-norm has the lower expected cost with 1:10 cost ratio for all values of PC(+). The maximum difference in expected cost for this cost ratio is about 58% between the Ts2-norm and V-norm. On the other hand, the operating range for V-norm algorithm is slightly narrower than the one in Ts2-norm, $0.00099 \leq PC(+) < 0.45$ and it does not include the highest cost ratio (1:10000) in any of the three configurations because the Ts2-norm has the lower expected cost with that cost ratio for all values of PC(+). The maximum difference in expected cost for that cost ratio is about 56% between the V-norm and Ts2-norm. In conclusion, when the system has to be deployed in an extreme risky environment it is recommended to use the V-norm while in a safer environment the Ts2-norm algorithm would lead the lowest expected misclassification cost.

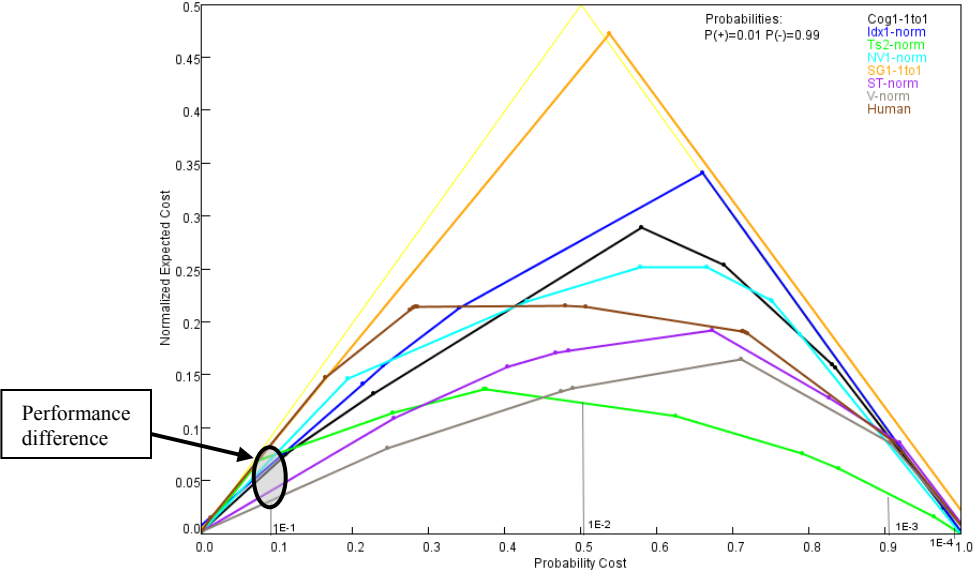


Figure 38 Cost curves for face matching algorithms – Severe condition.

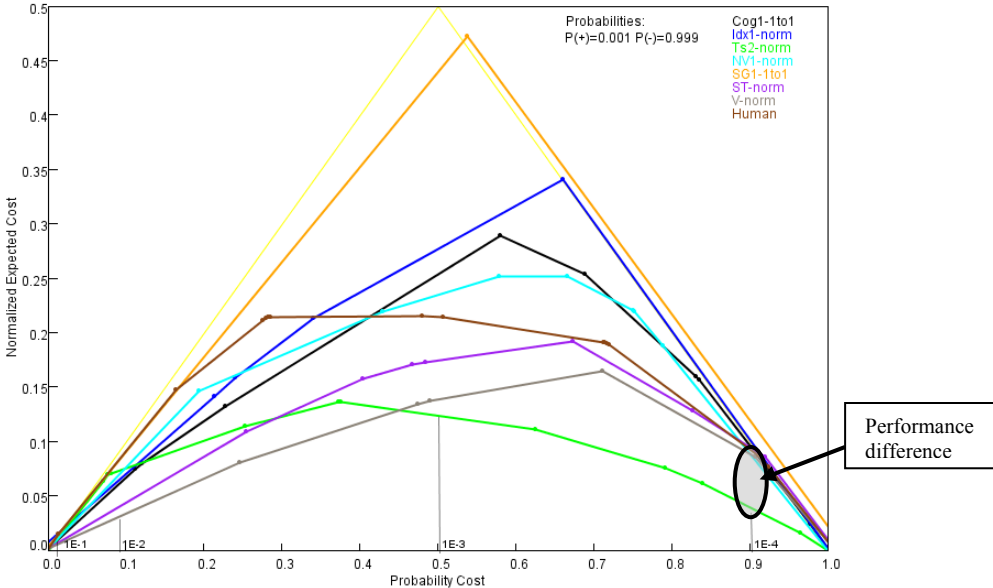


Figure 39 Cost curves for face matching algorithms - Guarded condition

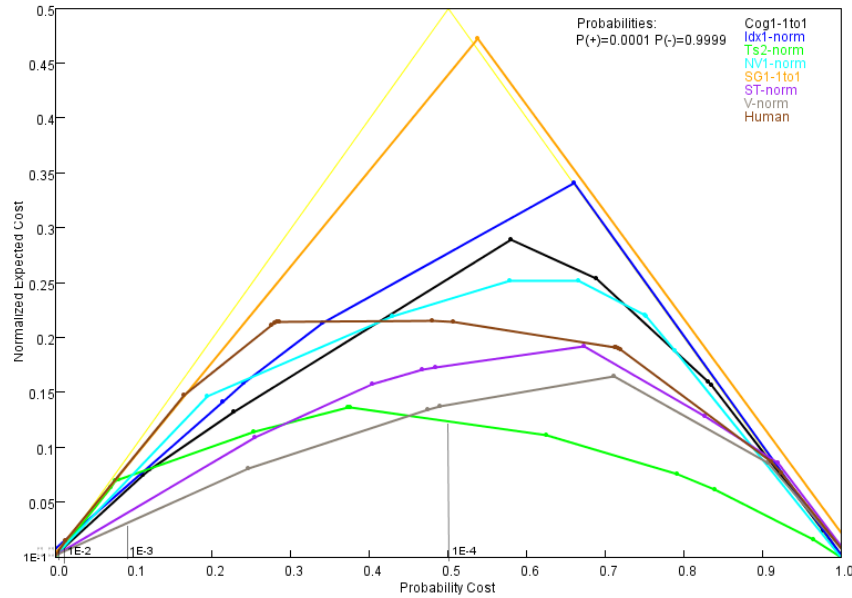


Figure 40 Cost curves for face matching algorithms - Low condition

Figure 41 to Figure 43 show the corresponding cost curves for the fingerprint modality for every alert level status that a border inspection could experience by a given probability of the impostor population. NEC algorithm offers the smallest normalized expected cost among the algorithms 123IDM2, Avalon, Biolink, Cogent, Identix and Motorola. Cogent shows similar recognition performance in the ROC curve (see Figure 36), but in the cost space is well separated from the NEC algorithm. The maximum difference between them is about 76% (0.0009 compared to 0.0037) which occurs when PC(+) is about 0.7913.

Table 14 shows normalized expected costs and their FMR and FNMR for the fingerprint modality. Contrary to what was revealed with the face modality, performance of fingerprint is highly satisfactory since the total waiting time is in the range of 20 to 32 minutes. The normalized expected cost is two orders of magnitude lower than in the case of face recognition algorithms and the FMR – FNMR ratio corresponds to acceptable configurations desired in the border inspection system.

The NEC algorithm includes all the cost ratios, with the highest normalized expected cost at 0.0013 in the operating range $0.3625 \leq PC(+) \leq 0.5501$.

Table 14 Cross-relation between cost curves for Fingerprint modality and the performance models

$p(+)$	$\mu = \frac{C(+ -)}{C(- +)}$	$PC = \frac{p(+)}{p(+)+p(-)*\mu}$	$FNMR$	FMR	$Norm(E[Cost])$	Total waiting time (min)
0.01	0.01	0.502513	0.001276	0.0013	0.00128806	28.0293
	0.001	0.909918	0.002026	0.0006	0.000728457	32.32856
0.001	0.01	0.090992	0.0001834	0.0031	0.000448787	20.26925
	0.001	0.500250	0.001276	0.0013	0.001288006	26.62272
	0.0001	0.909174	0.002026	0.0006	0.000729518	30.98755
0.0001	0.001	0.090917	0.001276	0.0013	0.001278182	20.11963
	0.0001	0.500025	0.002026	0.0006	0.001312964	26.48694

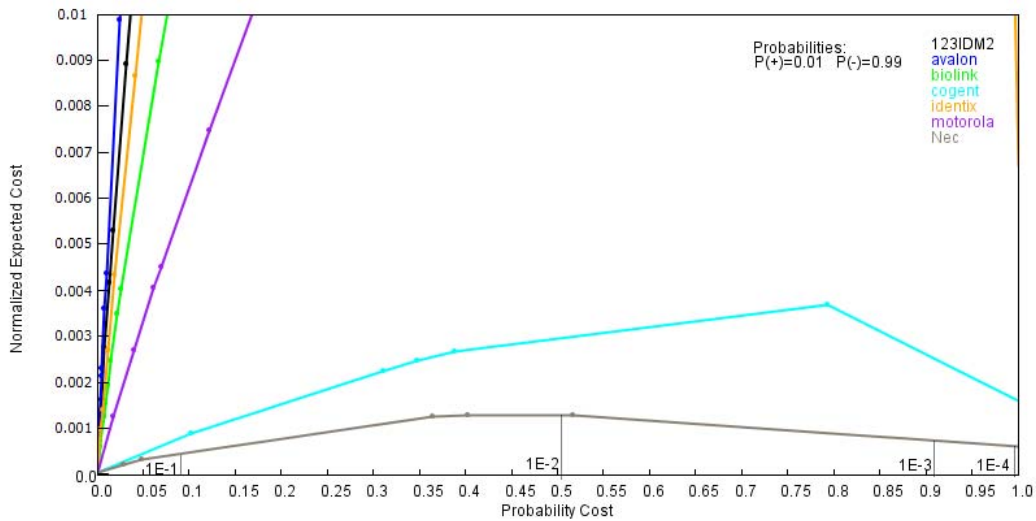


Figure 41 Cost curves for fingerprint matching algorithms – Severe condition

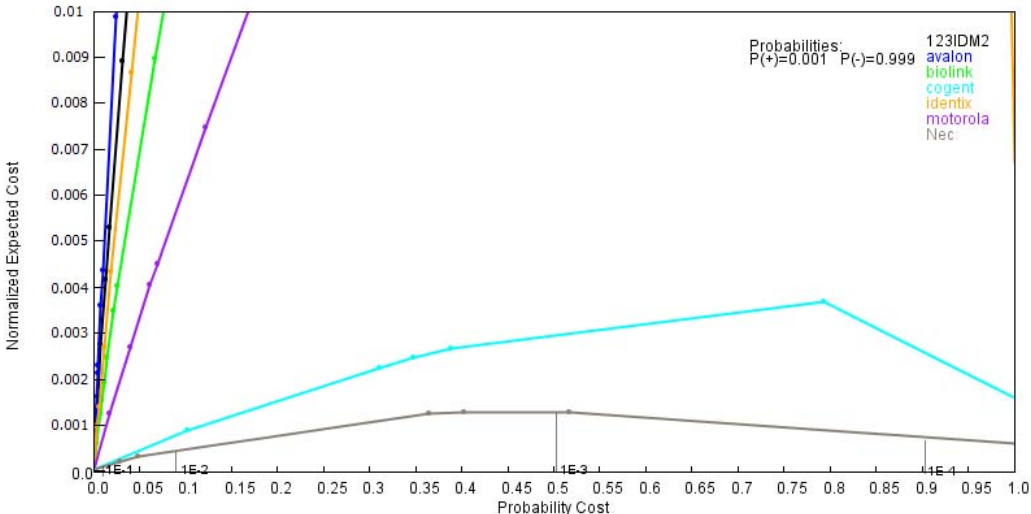


Figure 42 Cost curves for fingerprint matching algorithms – Guarded condition

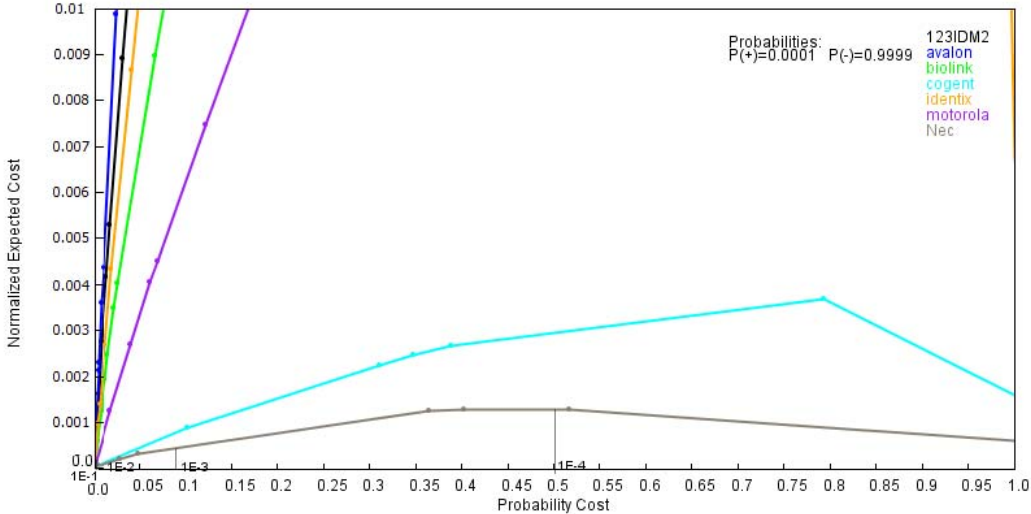


Figure 43 Cost curves for fingerprint matching algorithms – Low condition

Chapter 5 : Conclusions

5.1 Research Summary

As biometric-based border inspection systems increase in size and complexity, the need to evaluate their performance and security risk becomes critical. The use of analytic models provided an opportunity for exploring design and configuration alternatives. The benefit of analytical model is that their solution times are significantly faster than what can be expected from simulations.

This thesis uses the analytic modeling tool called LQNS (Layered Queueing Network Solver) to solve open model for a biometric-based border inspection system and Cost curves to evaluate risk.

First, we constructed a performance model of a biometric-based border inspection system by using available traces corresponding to average traveler arrivals of similar deployed systems [5] and information from technical reports [27][28][29]. We used this information as the traveler's workload intensity, where the range of workload intensities goes from low to peak. We estimated the software resource requirements using the information from available literature, technical reports, and educated guesses. We created several performance experiments in order to evaluate different type of system configurations related to screening policies, to configuration of system's thresholds, to localization of public key certificates database, and finally to replication of public key directories when using a centralized database.

Second, we established a security risk model of a biometric-based border inspection by creating cost curves of the biometric algorithms' recognition performance that may be deployed. We considered our system risk criticality as is either high or low, given that a traveler may be an impostor (high-risk) or a genuine (low-risk). We considered three

different security scenarios, such as severe level, guarded level and low level. For each level, we studied different misclassification cost ratios which range from the least costly to the most costly. Additionally, we estimated the normalized expected cost at those misclassification ratios.

Third, we analyzed a combination of performance and security risk in the border inspection system in order to find the most suitable biometric thresholds. Using the cost curves approach and the LQN models, we were able to explore the interplay between the loads imposed by the traveler arrivals, and the system security.

The biometric-based border inspection system performance is dramatically affected by increasing the size of watchlists and by changing the false match rate of the system. Among the configuration options and the level of security, it is clear that the system saturates when the size of the watchlist is larger than *1,000,000* entries. On the other hand, watchlist sizes between 1 thousand and 1 million entries at a fixed FMR and FNMR of (0.0001 , 0.001) causes marginal changes in the average inspection time. Variations at the FMR with a fixed watchlist size creates saturation in the system sooner than the system with different watchlist sizes. Our experiments suggest that FMR has a great impact in the overall system performance and security.

We have analyzed a performance model of a biometric-based border inspection system under a risk/security trade-off view with the aim to find the most suitable operating set-ups (FMR, FNMR) in both fields. We found that having a low normalized expected cost from the deployed recognition algorithms does not guarantee low waiting times that may be experienced by travelers in the border inspection system. The travelers' load imposes a very strong restriction over the results for any specific risk level. Therefore, the correlation between the cost curves and the performance models indeed guide the findings of optimal operating conditions.

5.2 Future Work

Our plans for the future work include the definition of confidence intervals for every alert condition scenario in the cost curves in order to find the bounds within which the risk (normalized cost) is expected to vary for every operating condition (FMR, FNMR). Additionally, we plan to incorporate the assessment of statistical significance in operating

ranges (misclassification costs and class probabilities) where cost curves from different classification algorithms are taken into account.

Another important line of the research could be to analyze the performance and risk of the border security systems by modeling the fusion of face and fingerprint matching algorithms at the score-level and decision-level. This would create opportunities to further analyze the operating points that would emerge as the most adequate compromise from the risk analysis and performance models.

Bibliography

- [1] Bedford T., and Cooke R., “Probabilistic Risk Analysis”. Cambridge University Press, pp. 3-11, 2001.
- [2] Bolle R., Pankanti S., and Ratha N., “Evaluation techniques for biometric-based authentication systems (FRR)”. IBM Computer Science Research Report RC 21759, 2000.
- [3] Bracchi P., Cukic B., and Cortellesa V., “Modeling the performance of Border Inspections with electronic Travel Documents”, IEEE 17th International Symposium on Software Reliability Engineering, 2006.
- [4] Bracchi P., “A Methodology for Software Performance Modeling and its application to a Border Inspection System”. Master Thesis. WVU, 2006.
- [5] http://www.cbp.gov/xp/cgov/about/mission/cbp_is.xml
- [6] Cortellesa V., Goseva-Popstojanova K., Appukkutty K., Guedem A., Hassan A., Elnaggar R., Abdelmoez W., Ammar H., “Model-based Performance Risk Analysis”, IEEE Transactions on Software Engineering, Vol. 31, No. 1, January 2005, pp. 3-20.
- [7] Doddington G., Liggett W., Martin A., Przybocki M., and Reynolds D., “Sheeps, goats, lambs and wolves: a statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation”. Proceedings of ICSLD ‘98, Sydney, Australia, November 1998.
- [8] Drummond C., and Holte R., “Cost curves: An improved method for visualizing classifier performance”, Machine Learning, Springer, pp. 95-130, 2006.
- [9] Phillips P. J., Scruggs W. T., O’Toole A. J., Flynn P. J., Bowyer K. W., Schott C. L., and Sharpe M., “FRVT 2006 and ICE 2006 Large-Scale Results”, NIST, March 2007.
- [10] Wilson C., Hicklin R. A., Korves H., Ulery B., Zoepfl M., Bone M., Grother P., Micheals R., Otto S., and Watson C. “Fingerprint Vendor Technology Evaluation 2003 Analysis Report”, NIST, 2003.
- [11] GAO-08-219, Border Security “Despite Progress, weaknesses in Traveler Inspections Exist at Our Nation’s Ports of Entry”. Report to Congressional Requesters, 2007.
- [12] Goseva-Popstojanova K., Hassan A., Guedem A., Abdelmoez W., Nassar D., Ammar H. And Mili A., “Architectural-Level Risk Analysis Using UML”,

- IEEE Transactions on Software Engineering, Vol. 29, No. 10, October 2003, pp. 946-959
- [13] Krouse W., Elias B., “Terrorist Watchlist checks and Air Passenger Prescreening”, CRS Report for Congress, 2006.
 - [14] Krouse W., “Terrorist Identification, Screening, and Tracking Under Homeland Security Presidential directive 6”, CRS Report for Congress, RL32366, 2004.
 - [15] Leveson N., “SAFWARE System Safety and Computers”. Addison Wesley, 3rd Edition, 1999.
 - [16] Object Management Group, “UML Profile for Schedulability, Performance, and Time Specification”. Version 1.1, 2005
 - [17] O’Gorman L., “Comparing Passwords, Tokens, and Biometrics for user authentication”, Proceedings of the IEEE, Vol. 91, No. 12, December 2003.
 - [18] Ortiz D., Pfleeger S., Balakrishnan A., and Miceli M., “Revisiting US-VISIT U.S. Immigration Processes, Concerns, and Consequences”, RAND Corporation, 2006.
 - [19] Petriu D.C., Zhang J., Gu G., and Shen H., “Performance Analysis based on the UML SPT Profile”. Chapter 14. Springer LNCS, 2005
 - [20] Raj J., “The Art of Computer Systems Performance Analysis”. John Wiley & Sons, 1991.
 - [21] Ratha N.K., Connell J.H., and Bolle R.M., “Enhancing security and privacy in biometrics-based authentication systems”, IBM Systems Journal, Vol. 40, No. 3, 2001, pp. 614-634.
 - [22] Schroeder B., Wierman A., and Harchol-Balter M., “Open Versus Closed: A Cautionary Tale”. NSDI’06: 3rd Symposium on Networked Systems Design & Implementation, USENIX, 2006.
 - [23] Smith C.U., and Williams L., “Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software”. Addison Wesley, 2002.
 - [24] Stoneburner G., Goguen A., and Feringa A., “Risk Management Guide for Information Technology Systems”, NIST Special Publication 800-30, 2002.
 - [25] Wein L., and Baveja M., “Using fingerprint image quality to improve the identification performance of the U.S. Visitor and Immigrant Status Indicator Technology Program”, Proceedings of the National Academy of Sciences, 2005.
 - [26] Woodside M., Franks G., and Petriu D.C., “The future of Software Performance Engineering”. IEEE, FOSE ‘07, 2007.
 - [27] “PKI digital signatures for machine readable travel documents”, version 4, technical report, ICAO, 2003.
 - [28] “ICAO requirements for epassports interoperability”, annex k, version 1, Technical Report, ICAO, 2004.
 - [29] “PKI for machine readable travel documents offering ICC read-only access, version 1. Technical Report, ICAO, 2004
 - [30] Jain A., Flynn Patrick, and Ross A., “Handbook of Biometrics”, Springer, 2008.
 - [31] Maltoni D., Maio D., Jain A., and Prabhakar S., “Handbook of Fingerprint Recognition”, Springer, 2003.

- [32] InterNational Committee for Information Technology Standards (INCITS). Study report on biometrics in e-authentication. Technical Report INCITS M1-06-0693, 2006.
- [33] Petriu D., and Shen H., "Applying the UML Performance Profile: Graph Grammar based derivation of LQN models from UML specifications", in Computer Performance Evaluation: Modelling Techniques and Tools, LNCS 2324, pp.159-177, Springer, 2002.
- [34] Gu G., and Petriu D., "XSLT Transformation from UML Models to LQN Performance Models", Proc. of 3rd Int. Workshop on Software and Performance WOSP'2002, pp.227-234, Rome, Italy, 2002.
- [35] Xu J., Woodside M., and Petriu D., "Performance Analysis of a Software Design using the UML Profile for Schedulability, Performance and Time," in Proc. 13th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS 03), Urbana, USA, 2003.
- [36] Balsamo S., Di Marco A., Inverardi P., and Simeoni M., "Model-based performance prediction in software development: A survey. IEEE Transactions on Software Engineering, Vol. 30, No. 5, pp. 295-310, 2004.
- [37] Israr T., Lau D., Franks G., and Woodside M., "Automatic Generation of Layered Queuing Software performance models from commonly available traces", ACM WOSP'05, pp. 147-158, 2005.
- [38] Singh H., Cortellessa V., Cukic B., Gunel E., and Bharadwaj V., "A Bayesian approach to reliability prediction and assessment of component based systems", IEEE ISSRE 2001, pp. 12-21, 2001.
- [39] Cukic B., "The virtues of assessing software reliability early", IEEE Computer Society, pp. 50-53, 2005.
- [40] Cortellessa V., Singh H., and Cukic B., "Early reliability assessment of UML based software models", ACM WOSP'02, pp. 302-309, 2002.
- [41] Wilson D., and Weber L., "Surveillance, risk and preemption on the Australian border", <http://www.surveillance-and-society.org>, pp.124-141.
- [42] Ceyhan A., "Technologization of security: management of uncertainty and risk in the age of biometrics", <http://www.surveillance-and-society.org>, pp.102-123.
- [43] Popic P., Desovski D., Abdelmoez W., and Cukic B., "Error propagation in the reliability analysis of component based systems", IEEE ISSRE 2005.
- [44] Yacoub S., Cukic B., and Ammar H., "A scenario-based reliability analysis approach for component-based software", IEEE Transactions on Reliability, Vol. 53, No. 4, pp. 465-480, 2004.
- [45] Franks G., Maly P., Woodside M., Petriu D., and Hubbard A., "Layered Queuing Network Solver and Simulator User Manual". Carleton University, Ottawa, Canada, 2005.
- [46] Petriu D., Franks G., and Hubbard A., "SRVN input file format". Carleton University, Ottawa, Canada, 1998.
- [47] Woodside M., and Franks G., "Tutorial Introduction to Layered Modeling of Software Performance". Carleton University, Ottawa, Canada, 2005.

- [48] Babu V., Batta R., and Lin li, "Passenger grouping under constant threat probability in an airport security system", Elsevier European Journal of operational research 168, pp. 633-644, 2006.
- [49] Nie X., Batta R., Drury C., and Lin li, "Passenger grouping with risk levels in an airport security system", Elsevier, 2008.
- [50] Martonosi S., and Barnett A., "How effective is security screening of airline passengers?", Interfaces, 2006.
- [51] Jiang Y., Cukic B., and Menzies T., "Cost curve evaluation of fault prediction models", IEEE ISSRE 2008, pp. 197-206.
- [52] Woodside C., Neilson J., Petriu D., and Majumdar S., "The Stochastic Rendezvous Network Model for Performance of Client-Server-Like Distributed Software." IEEE Transactions on Computers, Vol. 44, 1995.
- [53] Franks G., "Layered Queueing Network Solver Software Design" Department of Systems and Computer Engineering, Carleton University, 1994.
- [54] Rolia J., and Sevcik K., "The Method of Layers", IEEE Transactions on Software Engineering, Vol. 21, 1995.
- [55] Edmunds T., Sholl P., Yao Y., Gansemer J., Cantwell E., Prosnitz D., Rosenberg P., and Norton G., "Simulation Analysis of Inspections of International Travelers at Los Angeles International Airport for US-VISIT", Technical Report, Lawrence Livermore National Laboratory, 2004.
- [56] Sacanamboy M., and Cukic B., "Workload Representation in the Modeling of Border inspection points". The 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Fast abstract, Anchorage, Alaska, 2008.
- [57] Goseva-Popstojanova K., "Software performance engineering class notes", <http://www.csee.wvu.edu/~katerina/Teaching/CS-736-Spring-2009/CS-736-DataCollectionEarlyCycle.pdf>.
- [58] "Machine Readable Travel Documents (MRTDs): History, Interoperability, and Implementation", ICAO, TAG-MRTD/17-WP/16, 2007.

Appendix A: Detailed Uses Cases and Sequence Diagrams

In this appendix, we present in detail the assumed border inspection system configuration by means of use cases (see Tables 15-18) and their corresponding sequence diagrams (see Figures 40-44). The actors that interact with the system are: travelers and the interagency border inspection system. Further, the critical processes that we studied are: traveler examination, biographic checking, biometric verification and biometric identification.

Table 15 Traveler Examination expanded use case

Use Case	Traveler Examination	
Actors	Traveler	
Purpose	In this use case the traveler's documents and biometrics traits are verified against information of the traveler stored in system databases in order to grant access to the country to rightful travelers who does not present a risk for the country.	
Cross-reference	Biographic checking, biometric verification	
	1. The traveler presents his/her MRTD (machine readable traveler document) to the primary officer in the booth.	2. The primary officer scans the MRTD through the MRTD reader. 3. The data on the MRTD is retrieved and its corresponding digital signature is confirmed by employing the Public Key Certificate stored in the Public Key Directory or in the MRTD itself. 4. Verification of the Public Key Certificate

	<p>Authority, MRZ and face image is performed.</p> <p>5. The system initiates the biographic checking use case.</p> <p>6. The system requests the live inkless acquisition of traveler’s fingerprints and face image, see biometric verification use case.</p> <p>7. The primary officer grants the traveler access to the country.</p>
Alternate section	
<p>Line 2. If the MRTD is either damaged or its reader is not working properly, the officer manually enters the information into the system and continues to perform the biographic and biometric inspection.</p> <p>Line 4. If any of the verifications fail, the primary officer stops the traveler’s primary inspection process and sends the traveler to the second inspection.</p> <p>Line 7. The primary officer is not convinced with the system’s results, so he sends the traveler to further processing in the secondary inspection.</p>	

Table 16 Biographic Checking expanded use case

Use Case	Biographic Checking
Actors	Interagency Border Inspection System
Purpose	In this use case the traveler’s information is checked against existing information of the traveler in the CCD system database and the TNS in order to permit the entry of individuals who are not suspect of having a threat to the country, who have abided the terms of their admission to the country and who are not required for criminal acts.
Cross-reference	Traveler Examination
	<p>1. The basic traveler’s information such as: name, last-name, birth-date and MRTD number is sent to both the Traveler’s Name Server (TNS) and the</p>

	CCD.
	<p>2. Traveler's information from current immigration status and criminal violations are retrieved from the TNS.</p> <p>Picture and consular information related to visa applications, approvals, refusals and the biometric identifiers captured during the process of MRTD issuance are retrieved from the CCD.</p>
	3. A consolidate information about the traveler is displayed.

Table 17 Biometric Verification expanded use case

Use Case	Biometric Verification
Actors	Traveler, Border Inspection System
Purpose	In this use case the traveler's biometrics traits are verified against information of the traveler stored in system databases.
Cross-reference	Traveler Examination
	<ol style="list-style-type: none"> 1. The traveler's live fingerprints (right slap, left slap, and thumbs slap) are captured by the fingerprint scanner and a face image is taken by a digital camera. 2. The system checks the quality of the collected biometric traits. 3. The collected biometric traits are matched against the traveler's biometric templates in the Traveler Biometric System (TBS). 4. Matching scores for fingerprint and face are generated. 5. The system answers that the traveler is who claims to be.

Alternate section
Line 2. If the quality of the collected biometric traits is poor, the system will request again the acquisition of the biometric traits.
Line 5. When the system decides that the traveler is not who claims to be, the primary officer sends him/her to the secondary inspection booth.

Table 18 Biometric Identification expanded use case

Use Case	Biometric Identification
Actors	Border Inspection System
Purpose	In this use case the traveler's face image is compared against a consolidated watchlist stored in system databases.
Cross-reference	Traveler Examination
	<ol style="list-style-type: none"> 1. The traveler's face image taken by a digital camera in the primary inspection booth is sent to the TB DB by an inspector in the secondary inspection. 2. The TB DB generates match scores after comparing the traveler's face image against the entire consolidated watchlist. 3. The TB DB ranks the match scores and retrieves the top 50 identities.

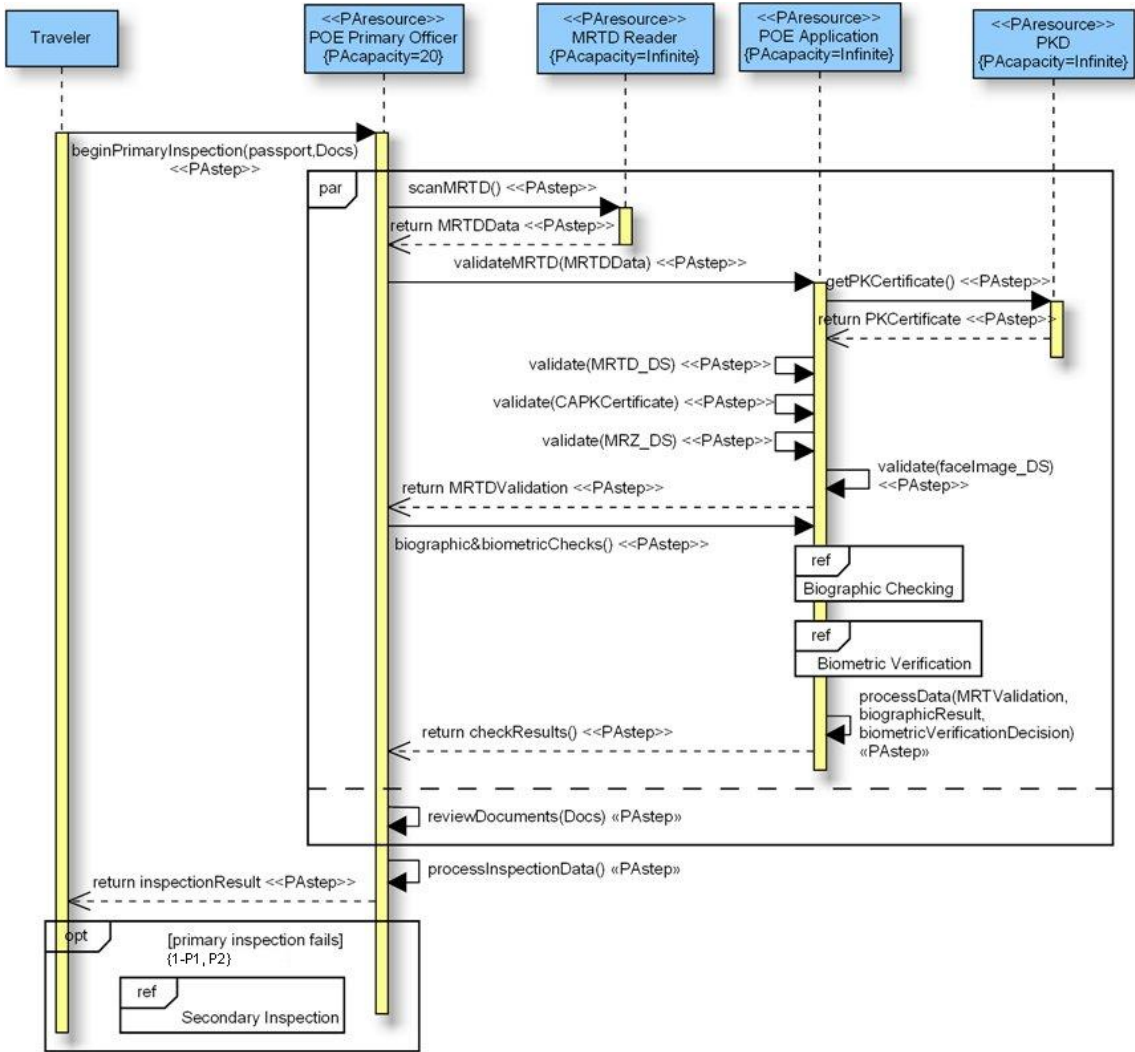


Figure 44 Sequence Diagram for the traveler examination use case. Adapted from [4]

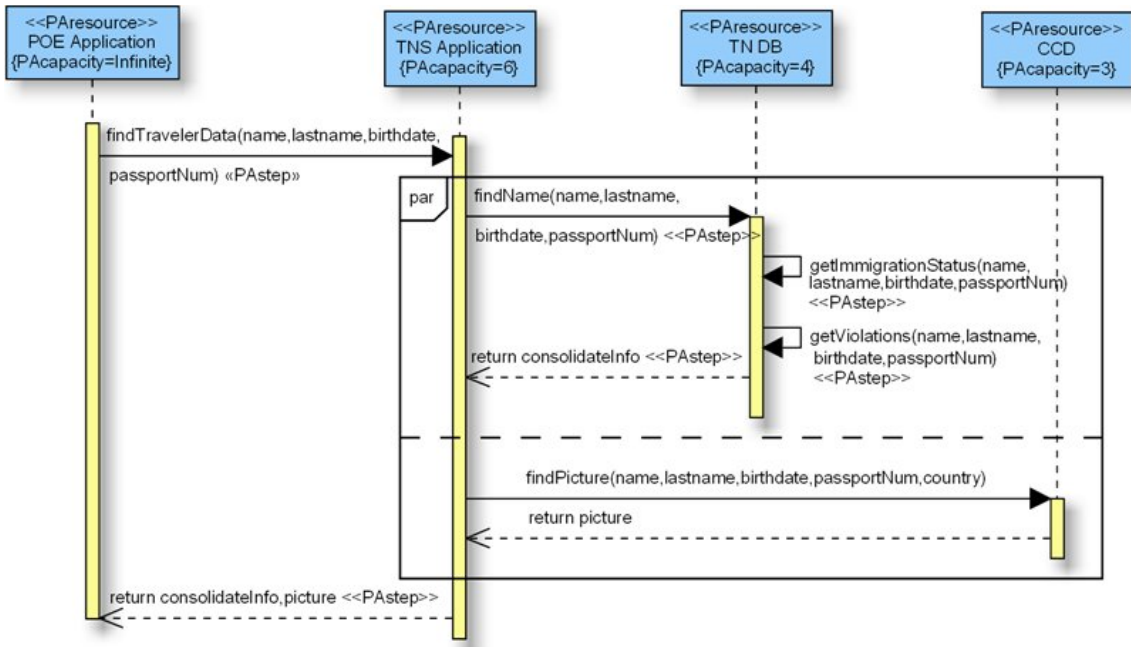


Figure 45 Sequence Diagram for the Biographic Checking use case

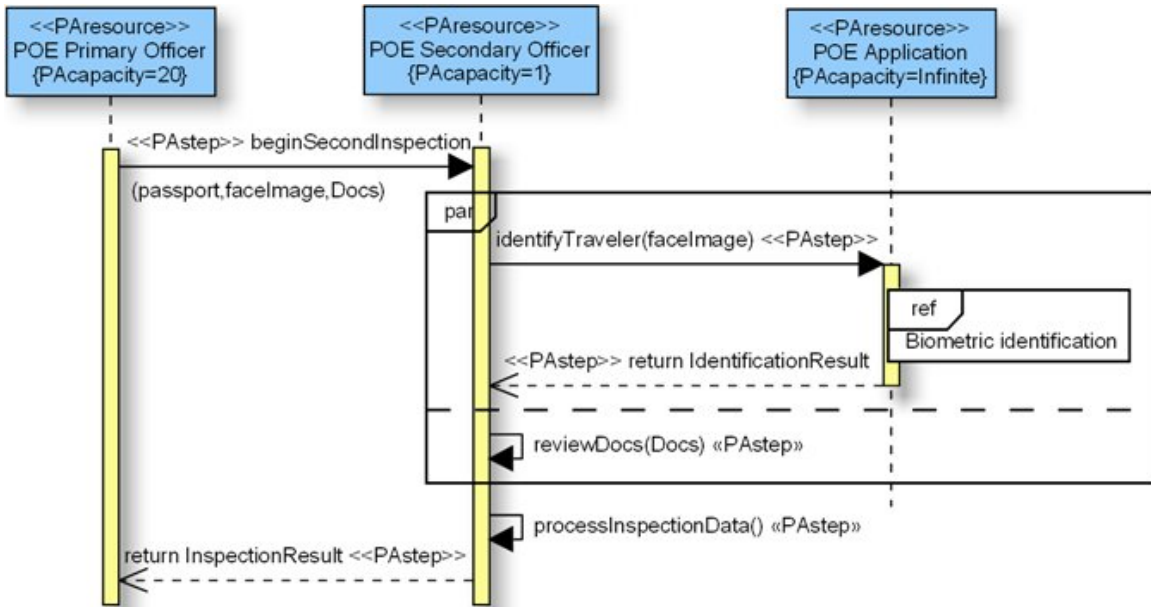


Figure 46 Sequence Diagram for the Secondary Inspection in the traveler examination use case. Adapted from [4]

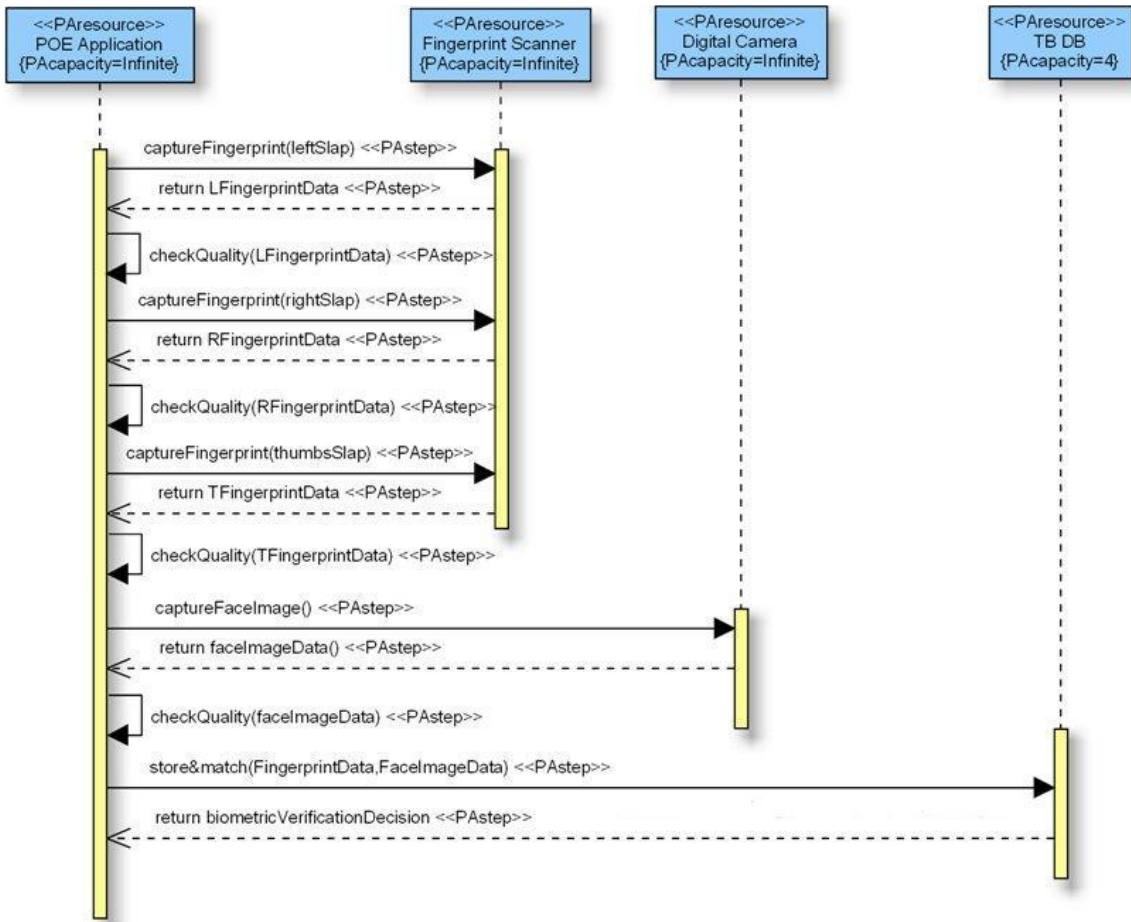


Figure 47 Sequence Diagram for the Biometric Verification Use Case



Figure 48 Sequence Diagram for the Biometric Identification Use Case

Appendix B: Performance Parameters

The assumed execution environment of the border inspection system is depicted here. Resource demands within each step performance scenario that is translated to a parameter in the LQN model are also presented. Table 19 and Table 20, present data assumptions that are extended from technical reports of analogous systems [4][27][28][29][58]. Table 21 to Table 24, present the estimated time of each <<PAstep>> for the sequence diagrams based on the data assumptions and calculated values.

Table 19 Hardware Platform Devices and Transfer Rates Specification. Adapted from [4][58]

	<i>POE Workstation</i>	<i>Traveler Name Server (TNS)</i>	<i>Traveler Biometric Server (TBS)</i>	<i>Consular Consolidate Database (CCD) Server</i>	<i>MRTD Reader</i>	<i>MRTD Card</i>	<i>LAN</i>	<i>WAN</i>
CPU	Pentium 2.6 GHz							
Memory	2 GB							
Disk Delay	6 ms	5.93 ms	5.93 ms	5.93 ms				
Transfer Time	130 MB/s	200 MB/s	200 MB/s	200 MB/s	424 kb/s	848 kb/s	100 Mbits/s	16.6 Mbits/s

Table 20 Estimated Data Size. Adapted from [27][58]

	Fingerprint Scans (10 fingerprints)	Face Image	Watchlist (10 ⁶ Face images)	Traveler Name Record	Consolidated Consular Record	Picture inside MRTD	MRTD digital signature	MRTD Public Certificate	Machine Readable Zone
Type	KB	KB	KB	KB	KB	bytes	bytes	KB	bytes
Size	50	20	2*10 ⁷	5	20	12704	20	1.8	88

Table 21 Scenario Steps with Performance Annotations for the Sequence Diagram of the Traveler Examination. Adapted from [4]

Scenario Step	Meaning
beginPrimaryInspection {PAdemand=('asm'd', 'mean', (0, 's'))}	Assumed time to generate a request for primary inspection.
reviewDocuments {PAdemand=('asm'd', 'mean', (20, 's'))}	Assumed time used for interviewing/reviewing- documents a traveler in the primary inspection.
processInspectionData {PAdemand=('asm'd', 'mean', (5, 's'))}	Assumed time spent by the primary inspection officer to decide whether the traveler can enter the country based on the results from checks.
return inspectionResult {PAdemand=('asm'd', 'mean', (5, 's'))}	Assumed time to communicate the result of the inspection process in the primary inspection.
biographic&biometricChecks {PAdemand=('asm'd', 'mean', (0.005, 's'))}	
captureFingerprint {PAdemand=('asm'd', 'mean', (0.005, 's'))}	Assumed time used for setting the fingerprint scanner up.
return fingerprintData {PAdemand=('pred', 'mean', (15, 's'))}	Calculated time used for capturing left slap, right slap and thumbs slap of the traveler.
checkQuality {PAdemand=('asm'd', 'mean', (0.005, 's'))}	Assumed time used to check the quality of the biometric trait.
capture faceImage {PAdemand=('asm'd', 'mean', (0.005, 's'))}	Assumed time used for setting the digital camera up.
return faceImageData {PAdemand=('asm'd', 'mean', (5, 's'))}	Assumed time used for taking a traveler's picture using a digital camera
send-store&matchBiometrics {PAdemand=('asm'd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.0329, 's'))}	Calculated time used for transmitting the biometric data collected from the traveler (fingerprint scans + face image) in the POE Workstation to the TBS, through the WAN that connect them. 0.0329 = 70KB/16.6 Megabits/s
read-writeBiometricData {PAdemand=('asm'd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.00637, 's'))}	Calculated time used for writing the biometric data collected from the traveler (fingerprint scans + face image) to the TBS disk, and to read a previously stored face image file (20KB) from it. 0.00637= 5.93 ms + (90 KB/ 200MB/s)
return biometricMatchResult {PAdemand=('asm'd', 'mean', (0.005, 's'))}	
processData(MRTDValidation, biographicResult, biometricVerificationResult) {PAdemand=('asm'd', 'mean', (0.005, 's'))}	Assumed time used for consolidating the results from the MRTD validation, the biographic information and the biometric verification result.
return checksResult {PAdemand=('asm'd', 'mean', (0.005, 's'))}	
scanMRTD {PAdemand=('asm'd', 'mean', (1, 's'))}	Assumed time for swiping the MRTD through the MRTD reader by the primary inspection officer.

Scenario Step	Meaning
return MRTDData {PAdemand=('pred', 'mean', (X, 's')) PAextOp=('pred', 'mean', (Y, 's'))}	Calculate time used by the MRTD Reader to read the data stored in the MRTD. The actual duration of the operation depends on the size of the MRTD data, which leads to the following alternatives: $X = 15695.2 \text{ bytes} / 424 \text{ kilobits/s} = 0.2708\text{s}$ $X = 12852 \text{ bytes} / 424 \text{ kilobits/s} = 0.2368\text{s}$ Time used by the MRTD to transfer its data to the MRTD reader. The actual duration of the operation depends on the size of the MRTD data, which leads to the following alternatives: $Y = 15695.2 \text{ bytes} / 848 \text{ kilobits/s} = 0.1446\text{s}$ $Y = 12852 \text{ bytes} / 848 \text{ kilobits/s} = 0.1184 \text{ s}$
validateMRTD {PAdemand=('pred', 'mean', (X, 's'))}	Time used to transfer the MRTD data from the MRTD reader to the POE Workstation through a 12 Mbits/s USB link. The duration of the operation depends on the size of the MRTD data: $X = 15695.2 \text{ bytes} / 12 \text{ MB/s} = 0.0093\text{s}$ $X = 12852 \text{ bytes} / 12 \text{ MB/s} = 0.0082\text{s}$
getPKCertificate {PAdemand=('pred', 'mean', (0.005, 's'))}	
return PKCertificate {PAdemand=('pred', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (Y, 's'))}	Time used to read Public Key Certificates from the Disk of the POE Workstation. The reading time depends on the number of certificates to be read (one or two), which leads to the next alternatives: $Y = 6.5 \text{ ms} + (1.8\text{KB} / 130\text{MB/s}) = 0.006513\text{s}$ $Y = 6.5 \text{ ms} + (3.6\text{KB} / 130\text{MB/s}) = 0.006517\text{s}$
validate(MRTD_DS) {PAdemand=('pred', 'mean', (X, 's'))}	Time used to validate the authenticity of the digital signature on the MRTD by computing a hash function (SHA-1) of the MRTD data, and by verifying the authenticity of the digital signature through the RSA algorithm using the Public Key of the MRTD signer (2048 bits) [27]. The next alternatives consider different MRTD sizes: $X = t[\text{SHA}_1(14695.2\text{bytes})] + t[\text{RSA}(2048\text{bits})\text{-verify}(20\text{bytes})] = 0.0091\text{s}$ $X = t[\text{SHA}_1(12852\text{bytes})] + t[\text{RSA}(2048\text{bits})\text{-verify}(20\text{bytes})] = 0.0091\text{s}$
validate(CAPKCertificate) {PAdemand=('pred', 'mean', (0.0015, 's'))}	Time used to validate the authenticity of the digital signature on the Public Key Certificate of the MRTD issuer by computing a hash function (SHA-1) of the certificate data itself, and by verifying the authenticity of its digital signature through the RSA algorithm using the Public Key of the Country CA (3072 bits) [27]. $0.0015 = t[\text{SHA}_1(1.8\text{KB})] + t[\text{RSA}(3072\text{bits})\text{-verify}(20\text{bytes})] = 0.0015\text{s}$
validate(MRZ_DS) {PAdemand=('pred', 'mean', (0.0009, 's'))}	Time used to validate the authenticity of the MRZ within the MRTD by computing a hash function (SHA-1) of the MRZ data, and by verifying the authenticity of the digital signature on the MRZ through the RSA with the Public Key of the document signer (2048 bits) [27]. $0.0009 = t[\text{SHA}_1(88\text{bytes})] + t[\text{RSA}(2048\text{bits})\text{-verify}(20\text{bytes})] = 0.0009\text{s}$

Scenario Step	Meaning
validate(faceImage_DS) {PAdemand=('pred', 'mean', (0.0006, 's'))}	Time used to validate the authenticity of the face image within the MRTD by computing a hash function (SHA-1) of the image data, and by verifying the authenticity of the digital signature on the face image through the RSA with the Public Key of the document signer (2048 bits) [27]. $0.0006 = t[\text{SHA}_1(12704\text{bytes})] + t[\text{RSA}(2048\text{bits})\text{-verify}(20\text{bytes})] = 0.0006\text{s}$
return MRTDValidation {PAdemand=('asmd', 'mean', (0.005, 's'))}	
findTravelerData {PAdemand=('asmd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.0118, 's'))}	Time used to exchange the TNS traveler's biographic and lookup information (5 KB) and his picture (20 KB). $25\text{KB} / 16.6 \text{ Megabits/s} = 0.0118\text{s}$
findName {PAdemand=('asmd', 'mean', (0.005, 's'))}	
getImmigrationStatus {PAdemand=('asmd', 'mean', (0.005, 's'))}	
getViolations {PAdemand=('asmd', 'mean', (0.005, 's'))}	
return consolidateInfo {PAdemand=('asmd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.00595, 's'))}	Time used to retrieve from the TNS disk the traveler's biographic and lookup data (5 KB): $5.93 \text{ ms} + (5 \text{ KB} / 200\text{MB/s}) = 0.00595\text{s}$
findPicture {PAdemand=('asmd', 'mean', (0.005, 's'))}	
return picture {PAdemand=('asmd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.006, 's'))}	Time used to retrieve from the CCD server disk the traveler's picture: $5.93 \text{ ms} + (20\text{KB} / 200\text{MB/s}) = 0.006\text{s}$
return consolidateInfo, picture {PAdemand=('asmd', 'mean', (0.005, 's'))}	
beginSecondaryInspection {PAdemand=('asmd', 'mean', (0, 's'))}	Time used by the secondary inspection officer to start an identification process.
identifyTraveler {PAdemand=('asmd', 'mean', (5, 's')) PAextOp=('pred', 'mean', (0.0094, 's'))}	Time used to send a face image of the traveler to the TBS. $20\text{KB} / 16.6 \text{ Megabits/s} = 0.0094\text{s}$
compareWatchlist {PAdemand=('asmd', 'mean', (5, 's')) PAextOp=('pred', 'mean', (97.66218, 's'))}	5 s is the time used by the TBS to match the traveler's face image with the set of 10^6 face images in the biometric watchlist. Time used by the TBS disk to read the set of 10^6 face image templates in the biometric watchlist. The total time required to perform the computation depends on the size of the watchlist. $5.93 \text{ ms} + (2 \times 10^7 \text{ KB} / 200\text{MB/s}) = 97.66218\text{s}$
rankIdentities {PAdemand=('asmd', 'mean', (0.005, 's'))}	
Return identities {PAdemand=('asmd', 'mean', (0.005, 's'))}	
reviewDocs {PAdemand=('asmd', 'mean', (300, 's'))}	Time used by the secondary inspection officer to thoroughly review the traveler's documents and personal effects.
processInspectionData {PAdemand=('asmd', 'mean', (5, 's'))}	Time used by the secondary inspection officer to decide if authorizing the traveler to enter the country based on the results from checks.

Scenario Step	Meaning
return inspectionResult {PAdemand=('asmd', 'mean', (2, 's'))}	Time used by the secondary inspection officer to communicate to the traveler the result of the inspection process.

Table 22 Scenario Steps with Performance Annotations for the Sequence Diagram of the Biographic Checking

findTravelerData	{PAdemand=('asmd', 'mean', (0, 's'))}
findName	{PAdemand=('asmd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.006, 's'))}
getImmigrationStatus	{PAdemand=('asmd', 'mean', (0.005, 's'))}
getViolations	{PAdemand=('asmd', 'mean', (0.005, 's'))}
return consolidateInfo	{PAdemand=('asmd', 'mean', (0.005, 's'))}
findPicture	{PAdemand=('asmd', 'mean', (0.005, 's')) PAextOp=('pred', 'mean', (0.0062, 's'))}
return picture	{PAdemand=('asmd', 'mean', (0.005, 's'))}
return consolidateInfo, picture	{PAdemand=('asmd', 'mean', (0.005, 's'))}

Table 23 Scenario Steps with Performance Annotations for the Sequence Diagram of the Biometric Verification

captureFingerprint	{PAdemand=('asmd', 'mean', (0.005, 's'))}
return fingerprintData	{PAdemand=('pred', 'mean', (15, 's'))}
checkQuality	{PAdemand=('asmd', 'mean', (0.005, 's'))}
capture faceImage	{PAdemand=('asmd', 'mean', (0.005, 's'))}
return faceImageData	{PAdemand=('asmd', 'mean', (5, 's'))}
store&matchBiometrics	{PAdemand=('asmd', 'mean', (0, 's'))}
MatchFingerPrint	{PAdemand=('asmd', 'mean', (0.005, 's'))}
MatchFace	{PAdemand=('asmd', 'mean', (0.05, 's'))}
Verification	{PAdemand=('asmd', 'mean', (0.5, 's'))}
return biometricVerificationDecision	{PAdemand=('asmd', 'mean', (0.005, 's'))}

Table 24 Scenario Steps with Performance Annotations for the Sequence Diagram of the Biometric Identification

identifyBiometrics	{PAdemand=('asmd', 'mean', (0, 's'))}
identifyTraveler	{PAdemand=('asmd', 'mean', (5, 's')) PAextOp=('pred', 'mean', (0.0094, 's'))}
compareWatchlist	{PAdemand=('asmd', 'mean', (5, 's')) PAextOp=('pred', 'mean', (97.66218, 's'))}
rankIdentities	{PAdemand=('asmd', 'mean', (0.005, 's'))}
return identities	{PAdemand=('asmd', 'mean', (0.005, 's'))}