Deliverable 6.1, A general systems model architecture

Authors:
Matthew Collinson, David Pym, and Julian Williams
University of Aberdeen
Contributions from Robert Coles, Raminder Ruprai
National Grid
Further information from discussion for section 10 provided by
Woohyun Shim, *UNITN*
Luca Allodi *UNITN*
Fabio Massacci *UNITN*

Document Number	D6.1
Document Title	Deliverable 6.1, A general systems model architecture
Version	0.5
Status	First draft
Work Package	WP 6
Deliverable Type	Report
Contractual Date of Delivery	01.02.2013
Actual Date of Delivery	20.12.2012
Responsible Unit	UNIABDN
Contributors	Authors above
Keyword List	Systems models, game theory, public policy
Dissemination level	PU

SECONOMICS Consortium

SECONOMICS “Socio-Economics meets Security” (Contract No. 285223) is a Collaborative project) within the 7th Framework Programme, theme SEC-2011.6.4-1 SEC-2011.7.5-2 ICT. The consortium members are:

1	UNIVERSITÀ DEGLI STUDI DI TRENTO	Universitï¿œ Degli Studi di Trento (UNITN) 38100 Trento, Italy http://www.unitn.it	Project Manager: Prof. Fabio Massacci Fabio.Massacci@unitn.it
2	DEEPBLUE	DEEP BLUE Srl (DBL) 00193 Roma, Italy http://www.dblue.it	Contact: Alessandra Tedeschi Alessandra.tedeschi@dblue.it
3	Fraunhofer ISST	Fraunhofer Institute for Software and Systems Engineering ISST Emil-Figge-Straï¿œe 91 44227 Dortmund, Germany http://www.isst.fraunhofer.de/en/	Contact: Prof. Jan Jï¿œrjens jan.juerjens@isst.fraunhofer.de
4	Universidad Rey Juan Carlos	UNIVERSIDAD REY JUAN CARLOS, Calle Tulipï¿œn s/n, 28933, Mï¿œstoles (Madrid), Spain. http://www.urjc.es	Contact: Prof. David Rï¿œos Insua david.rios@urjc.es
5	UNIVERSITY OF ABERDEEN	THE UNIVERSITY COURT OF THE UNIVERSITY OF ABERDEEN, a Scottish charity (No. SC013683) whose principal administrative office is at King’s College Regent Walk, AB24 3FX, Aberdeen, United Kingdom http://www.abdn.ac.uk/	Contact: Prof. Julian Williams julian.williams@abdn.ac.uk
6	TMB Transports Metropolitans de Barcelona	FERROCARRIL METROPOLITA DE BARCELONA SA, Carrer 60 Zona Franca, 21-23, 08040, Barcelona, Spain http://www.tmb.cat/ca/home	Contact: Michael Pellot mpellot@tmb.cat
7	Atos	ATOS ORIGIN SOCIEDAD ANONIMA ESPANOLA, Calle Albarracin, 25, 28037, Madrid, Spain http://es.atos.net/es-es/	Contact: Silvia Castellvi Catala silvia.castellvi@atosresearch.eu
8	SECURENOK	SECURE-NOK AS, Professor Olav Hanssensvei, 7A, 4021, Stavanger , Norway Postadress: P.O. Box 8034, 4068, Stavanger, Norway http://www.securenok.com/	Contact: Siv Houmb sivhoumb@securenok.com
9	SOÚ Institute of Sociology AS CR	INSTITUTE OF SOCIOLOGY OF THE ACADEMY OF SCIENCES OF THE CZECH REPUBLIC PUBLIC RESEARCH INSTITUTION, Jilska 1, 11000, Praha 1, Czech Republic http://www.soc.cas.cz/	Contact: Dr. Zdenka Mansfeldova zdenka.mansfeldova@soc.cas.cz
10	nationalgrid THE POWER OF ACTION	NATIONAL GRID ELECTRICITY TRANSMISSION PLC, The Strand, 1-3, WC2N 5EH, London, United Kingdom http://www.nationalgrid.com/uk/	Contact: Dr. Raminder Ruprai Raminder.Ruprai@uk.ngrid.com
11	ANADOLU ÜNİVERSİTESİ	ANADOLU UNIVERSITY, SCHOOL OF CIVIL AVIATION Iki Eylul Kampusu, 26470, Eskisehir, Turkey http://www.anadolu.edu.tr/akademik/yo_svlhvc/	Contact: Nalan Ergun nergun@anadolu.edu.tr

Document Change Record

DATE	PARTICIPANT	ACTIVITY
01/01/2012	ABDN (JW) visit by NGRID (RR)	Content production
23/08/2012	ABDN (JW) visit to UNITN	Content production
12/09/2012	ABDN (DP)	Document skeleton creation.
27/09/2012	ABDN (MC,DP,JW)	Draft document content specification.
08/10/2012	ABDN (DP)	Draft of Section 1 Introduction.
10/10/2012	ABDN (DP)	Draft of Section 2.
10/10/2012	ABDN (DP)	Draft of Section 3.
10/10/2012	ABDN (DP)	Draft of Section 4.
15/10/2012	ABDN (JW)	Draft of Sections 1.1 through to 1.5.3.
21/10/2012	ABDN (JW)	Draft of Section 10 following meeting with NGRID.
24/10/2012	ABDN (JW)	Revisions to Section 1.
26/10/2012	ABDN (MC)	Draft of Section 5.
03/11/2012	ABDN (DP)	Draft of Section 7.
04/11/2012	ABDN (JW)	Draft of Section 8.
04/11/2012	ABDN (JW)	Draft of Section 9.
05/11/2012	ABDN (MC)	Draft of Section 6.
20/12/2012	ABDN (JW)	Revisions and edits to Sections 1,8,9,10.
21/12/2012	ABDN (MC)	Edits to Section 5 and 6, and edits to 2,3,4.
05/01/2013	ABDN (JW)	Draft of Executive Summary
18/01/2013	NGRID	Meeting with Coles (Grid, NY)
23/01/2013	UNITN (Chiarini)	Quality Check
28/01/2013	URJC (Rios)	Scientific Review
28/01/2013	ABDN (JW)	Edits following internal Scientific Review.
24/01/2013	ABDN (JW, MC)	Minor edits following UNITN Quality Check.
25/01/2013	ABDN (JW)	Final Minor Edits.

Deliverable Description

Deliverable 6.1 from workpackage 6, presents the following: *A general systems model architecture.: D6.1 will be written in a suitable modelling language such as Core Gnosis and will the project consortium to build scenarios for testing various policy instruments.*

Contents

Executive summary . . . 5
1. Introduction . . . 9
1.1 Rules-based versus Principles-based Approaches . . . 11
1.2 Modelling Incentives and Principal-Agent Approaches . . . 11
1.3 Principal-Agent Problems in Risk Management . . . 12
1.4 Diminishing Marginal Returns to Security Investment . . . 13
1.5 The Role of Public Policy . . . 14
1.5.1 Policy Agendas . . . 14
1.5.2 Insurance Markets . . . 15
1.5.3 Summary . . . 16
2. A Discipline of Mathematical Systems Modelling . . . 16
2.1 Modelling Methodology . . . 17
2.2 Structure and Process Calculus . . . 18
2.2.1 Processes and Resources . . . 18
2.2.2 Location . . . 19
2.3 Environment . . . 20
3. Reasoning about Process Models . . . 20
3.1 Logic . . . 20
3.2 Model Checking . . . 21
4. The Gnosis Modelling Tool . . . 21
4.1 Adding located resource: secure boats . . . 22
5. Alternative Simulation Approaches . . . 26
5.1 Requirements . . . 26
5.2 Alternative Systems . . . 27
5.2.1 Traditional Applied Mathematical Systems . . . 27
5.2.2 Systems for Concurrent Modelling . . . 28
5.3 Summary . . . 28
6. Mathematical Models to Matlab Models via Gnosis . . . 28

7. An Architectural Methodology for Organizational Security Models 29
7.1 The Basic Concepts of Information Security 30
7.2 An Economic View 31
7.3 Modelling the Security Architecture 33
7.3.1 The Framework Layer 36
7.3.2 The Instantiation Layer 38
8. Real options and pricing risk 41
9. Models of attack and defense with risk averse targets 43
9.1 A Policy Model with Insurance 43
9.2 Attacking Targets and Levels of Defensive Expenditure 44
9.2.1 Self-Protection with Actuarially Fair Insurance 47
9.2.2 What does the equilibrium tell us? 49
9.3 Policy Maker Utility Theory and Loss Function for Vulnerability Management 49
9.4 The Power Utility Family 50
9.5 The Policy-maker's Problem 52
9.6 Decision Support 53
10. Preliminary Models for National Grid 55
10.1 SCADA and Control Systems 55
10.2 Interconnectors 56
10.3 Corporate Network 57
10.4 Smart Metering 57
11. Conclusions 58
BIBLIOGRAPHY 59

Executive Summary

Securing assets and the concept of security proffers an interesting set of challenges for applied modellers. In many circumstances, for example in classical access control, models of security processes need to represent system state (including the system architecture) in a high degree of detail because of the extreme sensitivity of security properties to changes of state. In many other circumstances, for instance in models of the behaviour of populations of human and economic agents subject to security threats or in models of systems with human security factors, this representation may need to include details of the choices and preferences of the agents at work within the system. Now there many security situations in which both types of detail are essential, for instance securing passenger safety in an airport (and this is partly why airport security is not straightforward). Indeed, most of the situations of interest in Seconomics will be of this nature.

However, and here is one of the challenges, it is (with any known technique) almost impossible to incorporate both detailed state and an agent-based economic view of incentives and contracts in a way that allows tractable and robust prediction of future system behaviour. One usually has a choice either to study very abstract models with tractable equilibria, or to use detailed simulations in a more heuristic manner.

The objective of this deliverable is to provide a framework that encompasses a flexible suite of models (and modelling techniques) that range from specific encapsulations of the system architecture to those that abstract away from the specific security architecture, but capture more of the agency, public good and externality issues involved in managing security.

Overview of The SECONOMICS Modelling Framework

The models included in SECONOMICS represent a toolbox of methodological approaches that cover the security case studies evidence reported in the requirements for workpackages 1, 2 and 3 (see D1.3, D2.3 and D3.4). In deliverable D6.1 we will cover the following three basic modelling approaches.

Phrasing of this document

In this document we will refer to ‘security problems’ as specific elements of the case study (a specific security context) work that maybe modelled by a single version of our modelling framework. For instance the security problems relating to SCADA systems for case study 2, referred to in deliverable D2.3 is the problem space.

The models relating to SCADA systems are a set of models referring to the SCADA system and related economic and social policy issues regarding security features of this system. Therefore ‘model’ refers to the set of tractable architectures, policy objectives and strategic interactions for this security problem case.

We refer to architecture usually in terms of the structure of the technological system. Separately although this can be directly overlaid we refer to the ‘mechanism’ as the set of constraints on strategic actions for ‘agents’ (those actors that make decisions relevant to security outcomes within the case problem). Finally we have a series of metrics, that refer

to the security state of a particular system, these maybe measured inter temporally and aggregated to form measures of the overall security status of a system within the security problem context.

Structure of this document

This document is divided into sections based on a concurrent narrative surrounding the SECONOMICS modelling framework. Each section contains specific information regarding the design of models and where possible we have outlined specific examples.

Several sections contain specific information relevant to the modelling of security problems provided in the case study workpackages (WP1, WP2 and WP3). These are not designed to be read by non-specialists, but to provide reference material for the project. Further specific examples of the modelling framework are provided in a series of papers attached as an appendix.

Section 1 introduces the overarching concepts of economic modelling for security policy applications. Here we discuss the motivation for modelling, the types of policy questions that can be answered with appropriate modelling techniques and some preliminary approaches to the case study workpackages.

We also present an overview of the implementation mechanisms of policy approaches in this context. In particular we focus on rules versus principles based approaches to enacting policy requirements. Whilst this is a well known issue in most public policy contexts [1] it has not been studied widely in the area of rule-setting in security. We introduce this concept using some specific examples from critical national infrastructure.

Moving on from policy implementation, we begin to outline the needs for security policy in an economic context by outlining a series of models of security scenarios. In each case we illustrate the need for placing constraints on actions (due to incentive incompatibility) of individual agents to maximize the global level of welfare.

In section 2 we present a detailed systems modelling framework with explicit representations of relevant systems architecture, and logical methods for reasoning about such models.

In section 4 we outline a simulation modelling tool GNOSIS that captures the mathematical structures outlined in 2. The GNOSIS modelling tool currently does not include specific representations of economic theory. Part of the work of deliverable 6.1 is to illustrate how we will use mathematical modelling languages such as Mathworks MatLab to integrate the mathematical structures from 2 and integrate them with notions of utility theory and welfare to better encapsulate the economic interactions inherent in such models.

Following on from this foundational work in section 5 we explain how we can use more general mathematical modelling languages such as MatLab to capture the systems modelling ideas outlined previously and then combine them with economic representations of consumption and welfare. Section 6 then provides a set of references to elements of the Mathworks MatLab programming language to illustrate the existing technologies that can be used in conjunction with the GNOSIS modelling approach.

In section 7 we provide a specific guidance on integrating economic and systems models

[1] see for instance the The Institute of Chartered Accountants in England and Wales summary paper on this topic in accounting http://www.icaew.com/~/media/Files/Technical/Ethics/principles-vs-rules.pdf

in a security problem context. In subsection 7.3 we outline a worked example of an airport security architecture. The systems model in this version of the model has a policy function based on observed metrics that can be used to compare performance of different configurations of the airport security architecture. We derive the core features of the policy function in subsection 9.3.

The objective of this section is to demonstrate a model whereby taking the specific nature of the system architecture in account offers major benefits over standard economic interpretation, whereby the mechanism is generally simplified to allow for more comprehensive comparative statics (elucidation of model properties by changing the structural model parameters).

Section 8 presents a series of results broadly related market based pricing or real options models. These models are useful when you have exogenous risks and allow for the calibration of monetized cost benefit analyses. The models typically measure risk next to a known set of benchmarks with easy to value properties (e.g. using an asset pricing model, such as a Geometric Brownian motion value process or a multi factor asset pricing model such as the famous Capital Asset Pricing Model (CAPM). The objective of these models is to impute discount rates on assets that allow for a) comparison and b) addition of risks. These risks can be converted via a market mechanism to additive costs. This approach is useful when there is at least one easily comparable liquid asset.

In section 9 we develop a series of micro theory models of externalities and incentives (good when the architecture is simple, intractable to formulate otherwise) useful for understanding how to build contracts and incentive structures that improve welfare. This includes principal agent problems, models of externalities and models of public policy, institutional analysis and design. We also present an overview of insurance in this context, again with a view to monetisation/or cardinality of preference of the impact of security policies.

In each of the modelling sections we outline a series of examples in this document, based on the initial requirements from the case studies to illustrate how recombination of this set of tools can be used and some indication of the appropriate balance of techniques given the specific set of security problems envisioned within the project.

1. Introduction

Managers, consultants, and security engineers have responsibility for delivering the security of possibly large, complex systems. Policy-maker and industry/business leaders, on the other hand, have responsibility for ensuring the overall sustainability and resilience of information ecosystems that deliver services, including those in commercial, governmental, intelligence, military, and scientific worlds. Despite these differences in focus and scope, both groups must make policy design decisions that combine a wide range of competing, often contradictory concerns.

Considering this range of stakeholders, we are motivated by the following closely related questions:

- For a given system, with a given set of stakeholders operating in given business and threat environments, how do we determine what is an appropriate (i.e., effective, affordable) security policy? What attributes should be protected, to what extent, in what

circumstances? What impact on business operations is acceptable, and at what financial cost?

- Such an analysis will, if it is to be achievable and robust, be dependent on the provision of rigorous economic and mathematical models of systems and their operations. How are we to express and reason about policies so that their effectiveness against the desired security outcomes and their impact upon the stakeholders and business operations can be understood?

Our aim is to establish a mathematical basis for a systems security modelling technology that is able to handle the structural aspects of systems, the stochastic behaviour of their environments and, specifically, a utility-theoretic representation of security policies and their effectiveness.

Previous work, largely conducted in the context of an industrial research laboratory (HP Labs), has established a methodology of combining elementary utility theory with mathematical systems models [1, 2, 3, 4], grounded in the theory of distributed systems [5], for which an execution engine has been developed (Core Gnosis [6]), to explore the value of security policies and technologies. Between 2008 and 2011, Pym led, with Simon Shiu, a project within HP Labs called 'Security Analytics' [7, 8, 9]. This project, which built on the parallel, externally facing 'Trust Economics' project [10] funded by the Technology Strategy Board, was concerned with applying mathematical systems modelling techniques (developed within Trust Economics) to large-scale security management problems. Security Analytics worked directly with some of HP's largest customers, including the Manhattan-based security team at Citigroup (the world's largest bank). 'Security Analytics' has now transferred into HP's security business [7, 8] (http://www.hpl.hp.com/news/2011/oct-dec/security_analytics.html)

An illustrative example, reported in [11], concerns the use of USB memory sticks by the employees of a bank for the purpose of transporting data to and from client sites. Security managers wish to mitigate the evident risk of data exposure by requiring employees to encrypt data so transported. Implementing such a policy requires several actions on behalf of the security manager: (investment in) training of staff (T), monitoring (M), and technical support (S). Each of these carries a cost to the organization. Other costs are associated with the actions of the employees: they may suffer embarrassment (E) if they are unable to retrieve data when visiting a client, or they may be reprimanded (R) for failure to comply with policy. An executable system model was constructed in which employees are modelled as processes engaging in E and R actions, as well as productivity actions, modelled as successful data transfers (Tr). The frequencies of these actions are determined as a function of the level of T, M, and S investment. This model allows exploration of the variation in levels of confidentiality and availability of data as T, M, and S are varied, and utility theory enables analysis of the value of various security investment policies.

Similar methodology has been applied in a number of other scenarios, including an analysis of patching policies for IT managers [12, 13], and the case of a large organization managing information security policy and technology during a process of divesting itself of subsidiaries [14]. This work employed a modelling tool, Gnosis [6], that captures a mathematical approach to modelling distributed systems based structural concepts of *location*, *resource*, *process*, and, stochastically, *environment* [1]. Gnosis allows these concepts to be deployed at varying scales, and is capable of modelling both protocol-level interactions

and system-scale evolutions, which is our focus here. For example, in [11, 14, 15, 9], as described above.

The systems that we are interested in modelling and understanding — such as National Grid's power distribution and control systems, or the security systems in place at airports — all operate within regulated environments.

An important, and immediately evident, question is the following: How should regulatory policies and procedures be formulated and implemented? There are two leading candidates:

- Regulation by rules and compliance;
- Risk-assessment-principles- and validation-based regulation.

These choices are applicable across the range of domains of interest and application of this project.

1.1 Rules-based versus Principles-based Approaches

Rules are sets of instructions, where each rule is subject to either a dichotomous (adhered to, or not adhered to) or continuous (e.g., 90%, 70%, 50%, etc.) compliance measurement. *Principles* are designed to be general statements that define a goal, or objective, of the entity adhering to the principle. In the cases of information security or cyber-security, the main constituent of a principles-based approach is a risk-based approach. Risk mitigation is therefore built into the principle.

The main advantage of principles- or risk-based approaches to regulation is that they can flexibly encompass a wider range of scenarios than rules-based approaches. However, principles devolve discretion to the entity being regulated and require guidance on the level of conservatism to be applied to their implementation. On the other hand, a rules-based regulatory system ensures that all parties that need to adhere to it are applying the same set of security controls, and may even specify the details of how the controls are to be implemented. This can be seen as a 'double-edged sword' because whilst all parties will have the same level of security, if there is a gap in the regulation — e.g., if a particular aspect of security is missed — this will affect all parties in the same way and the systematic risk will be high. Alternatively, a risk-based esystem, where the individual parties identify the type of security controls that they will implement separately, ensures that the systematic risk is lower.

It is important to note here that the risk-based methodology and framework described below is simply a particular risk-assessment methodology. Both a risk-based and rules-based regulatory framework could require a risk assessment to be completed but the specific requirements around how it is done and applied to the business are likely to be different.

1.2 Modelling Incentives and Principal-Agent Approaches

If we assume that employees need to adhere to certain behavioural constraints to operate towards a firm's specific cost-risk target, and then a natural issue of aligning incentives appears, this is a standard principal-agent problem in economics. As we add up the choices

of all of the firms as collections of principals and agents, we now move to a public policy aspect of economics.

A natural question is where to place the specific constraints on behaviour and what mechanism (regulatory framework) should be used to enforce those constraints. In a principles-based system a set of idealized outcomes is specified. Alternatively, if a public policy maker sets a series of rules then these rules may (a) conflict with the risk targets of the firms, and (b) conflict with the target risk of the agents working in the firm. Setting a penalty structure based on violations of rules does not always result in the correct internalization of externalities at both the level of the firm and the wider economy.

Externalities can arise from the following typologies of economic interactions:

- Principal agent problems within firms and organizations
- Investments in security protection
- Public policy and regulatory approaches (from an economic standpoint)
- The potential role for insurance as a mechanism of regulation
- The potential role of derivatives markets in hedging security risk.

This section is designed to be a general overview and not specifically attached to Critical National Infrastructure (CNI) issues, although examples from this domain are given herein.

1.3 Principal-Agent Problems in Risk Management

We can treat part of CNI as an information processing ecosystem [16], where security leaks have a variety of costs associated with them. Information ecosystems are commonly characterized by services: individuals and organizations (agents) acting on behalf of other user individuals and organizations (principals).

In the CNI industry as with other industries there are two levels to this. The policy-maker or regulator at the very top (principal) communicates with individual organizations' decision maker(s) who in this case are considered agents. The principles or rules previously communicated to each organization's internal decision-maker are then disseminated to its employees. In this step, the internal decision maker becomes the principal and the employees the agents. The issue here is the appropriate communication of policies from the top principal (policy-maker or regulator) to the agents (employees) at the bottom; this is known as the 'Principal-Agent problem'. For example, if the policy-maker defines principles that it requires organizations to follow, these principles need to be ingrained in any controls the internal decision-makers set for their employees.

This is in common with very many other economic activities, such as the mechanism of government and the separation of ownership and management of firms. The heart of principal agent problems stems from the misalignment of risk preferences between principals and agents, and the cost of monitoring agents to the principal. Agents seek to maximize their revenues; they can do this by taking more risk with the capital provided to them by the principals — thus there is an incentive problem. This incentive problem can be mitigated by:

a) placing rules-based restrictions on activities, or b) contractually aligning the incentives of the agents to the principals.

It is difficult to place the CNI Principal-Agent problem without the public policy context and, vice versa, it is inappropriate to define the public policy role of CNI without understanding the atomic, individual Principal-Agent problem within the structure of each firm.

1.4 Diminishing Marginal Returns to Security Investment

A key theme in the above example is the cost of investment in both security provision of the ecosystem and monitoring the individual agents. A key tenet of the security economics literature (see Gordon and Loeb [17] for a summary) is that the level of risk is (on average) decreasing in investment and monitoring (in the case of efficient investment and monitoring strictly decreasing) and that the rate of decrease is again diminishing with extra investment. The term 'monitoring costs' includes all costs associated with aligning incentives (that reduce the need for supervision) and opportunity costs created by engaging in this activity.

A simple two-dimensional model is as follows. Let x and y be the investments in security technology and monitoring respectively. The principal's problem is to minimize the following loss function, by choice of x and y:

$$L(z)S(x, y; z) + x + y.$$

Here $S(.)$ is a risk function that translates investment in technology and monitoring into a residual vulnerability of loss against an amount at risk L from a security breach, given a set of environmental conditions contained in a state vector z. In this case z relates the level of effort in attack and defense to the degree of loss. The value of the variables in the vector z might be derived from an equilibrium relationship, for instance z may represent the Nash versus Stackelberg equilibrium for a continuum of attackers and defenders. For instance, this vector may incorporate a feedback from the size of the loss to the probability of a successful attack, or in the case of externalities, z might contain the deviations of other firms requirements for a global welfare maximising level of investment in monitoring and technology (a technological externality). The salient point is that when a firm computes its optimal stance it only includes costs that are directly relevant to it. If these are the sum of all costs in the economic system then a Pareto efficient outcome is achievable (social welfare is maximized). However, if some costs to other firms by a particular firm's choices are not internalized by that specific firm (so there are externalities), then social welfare cannot be maximized without some form of social coordinator assigning property rights. These property rights then adjust the cost function to account for the externalities (and hence they are internalized).

Consider the risk management case. In an interdependent economy the risk appetite of a firm affects itself and other firms. If the cost of this risk sharing is not distributed across firms in a manner that is appropriately weighted, e.g. assignment of liability claims (the property right), then firms will only cost-in their own risk and not that of other firms in the market. When firm weights are highly-asymmetric, firms are incentivized to dump risks rather than pool them. In a CNI context, using electricity delivery as the example, the electricity distributors and generators may not appropriately secure their own assets connected to electricity transmission systems and assets against cyber threats. This is because they

assume that the transmission operator, National Grid, will undertake the cost of protection (having the higher weighting in the economy).

Nesting this problem within a complex ecosystem linked by z, we can create an rich family of models that capture many of the observed phenomena documented in the practitioner literature.

Atomized models such as those outlined in the previous sections can be added together using conventional utility functions to monetize losses of different types, including cyber risks being materialized. This form of multi-attribute utility theory is commonly used in security policy to assist in the monetization and cross addition of losses from various types of security breach; see, for instance, (Ioannidis, Pym, and Williams [5]).

1.5 The Role of Public Policy

Once a model of the threat environment and the interaction of the dimensions of investment, risk and environment has been mapped, the next step is to understand the interaction of policy in the creation (or erosion) of incentives to effectively manage risk.

In security policy scenarios the models have three classes of actors, which have been mentioned in earlier parts of this section, but are presented again for clarity:

Policy makers or regulators that have objective functions based on broad social welfare targets. In the case of National Grid and the electricity transmission network in the UK, the regulator is the Department of Energy and Climate Change (DECC).

The alignment of incentives between the layers of policy makers in this context proffers an interesting set of economic questions. First, are policy makers operating with identical enforcement practices. For instance, in critical infrastructure the government policy maker, the infrastructure provider management at various levels or in Airport security processes, there are multiple layers to the policy management issue. Not all of the layers of management have the same incentive structure.

1.5.1 Policy Agendas

Policy (in this context) is constrained to mean the following:

- Imposing punishments on revealed antagonists, such as fines for employees committing gross misconduct in particularly sensitive environments such as CNI;
- Requiring particular behaviours of the agents that are exposed to risk (with punishments for non-compliance) — this is a rules-based system where specific requirements are imposed on the agents below;
- Providing global insurance to agents in the event of loss for a particular level of rent.

A substantial tract of Economic analysis focuses on efficient distribution of resources to participants in an economic system. The trade-off is between Pareto efficiency and the Nash equilibria driven by strategic players best response strategies. Under most circumstances these are not the same, i.e. in the absence of a social policy maker the global welfare maximizing choices are not the choices undertaken by the

Pareto efficiency implies that social welfare is maximized via a process where each participant maximizes their own utility function over a set of preferences. There is a gradual convergence towards a maximum social welfare point through the continuous individual optimizations. The optimization is assumed to produce a social welfare optimum in the absence of externalities between agents. Externalities refer to direct and indirect effects on other agents not accounted for by other welfare maximising agent in their own utility functions.

A good set of examples stem from the public goods literature on externalities that are not internalized by individual agents. 'Tragedy of the commons' problems involve public goods for which the sustainability of the public good is often not sufficiently weighted by the group of individuals utilising this good. Grazing rights on public land are a good example of externalities in public goods.

For individual firms within an economic system, regulation is formed from a variety of constraints on behaviour (for instance minimum levels of effort and investment in technological and human security) that have punishments for non-compliance. Policy makers can act as enforcing mechanisms for social coordination problems of information sharing.

A fundamental economic concept is that the presence of externalities creates the need for public policy interventions. This intervention can come in several varieties, for instance a restriction on behaviour to ensure a socially optimal outcome (e.g. forcing individuals with penalties to ensure their computers are updated and secure) or by assigning property rights and liability clauses that distribute costs in a manner that reflects the cost of action on others arising from individual choices.

For information security, the literature has identified potential coordination mechanisms:

- Information sharing and coordination on potential risk vectors. Mechanism: Compulsory reporting of information to an information clearing house that then sets out guidance on risk mitigation (this is the current American approach to cyber-security). This assumes that all costs can be identified and allocated by appropriate information sharing mechanisms. Transfers to mitigate externalities are then isolated as direct transfers (e.g., private litigation or via memberships of associations with credentials);

- Behavioural constraints. Mechanism: Enforcing behaviours via a rules-based system or a set of risk targets evaluated by sets of metrics designed by the policy maker or regulator. This sets out behavioural constraints (either via principles or rules) that have penalties associated with non-compliance. These penalties need to reflect the costs not borne by individual agents (firms or staff) for their own personal actions;

- Insurance markets — these are discussed further below.

1.5.2 Insurance Markets

Mechanism: Compulsory purchasing of insurance from either a monopoly insurer or insurance market. The insurance company then sets behavioural requirements contractually. Two types of insurance market are possible:

- Compulsory insurance markets, all agents (usually at country level) need to purchase insurance, from either a monopoly or competitive market;

- Voluntary insurance, again either from either a monopoly or competitive market. The mechanisms are required in cases whereby an externality exists. Their efficacy is then based on the efficiency (from a global social welfare perspective of the cost of mitigation) in internalizing externalities.

Internalizing externalities is the process by which the cost (or benefit) of an externality is incorporated into an agent's utility function (either via joint-optimization or constraint) and as such the potential externality is internalized. From the previous discussion the position of a critical infrastructure provider results in two potential effects:

- First, they absorb externalities because a cost of security failure is so high that they are willing to bear the costs of other firms and agents (There is a positive effect for the other agents, but a negative effect for the critical infrastructure provider);
- Alternatively, a negative effect for the other agents, but positive effect for the critical infrastructure provider is that the infrastructure provider's security costs are disproportionately distributed to other agents.

More specifically, in the list of potential mechanisms previously discussed, the first approach maximizes social welfare valid if the attack probability $S(x*, y*)$ for optimal choices of x and y is independent of the choices of other agents (employees) in the system. This approach is effective in dealing with externalities, but may not be flexible enough when the problem is extended to a dynamic setting with repeated interactions; that is, the risk generating mechanism changes or the technology of defence renders the imposed constraints irrelevant.

1.5.3 Summary

This section has reviewed potential areas of public economics that could be applied to the regulation of various types of firms, individually and in groups. We have reviewed the various types of mechanisms that can allow risks to develop and the methods commonly used in economics to mitigate or monetize them. We have outlined the advantages and disadvantages of three mechanisms of risk sharing: public policy based approaches with self insurance, insurance markets (monopoly and competitive) and market-based approaches using derivatives contracts.

2. A Discipline of Mathematical Systems Modelling

All of the policy formulation and decision-making problems discussed above are situated in the context of complex (information processing) systems.

Many security modelling situations require a richness of detail in the models, for example, when a decision-maker wishes to understand the behaviour of a system with complex, interacting components and security controls. In these situations, a fine-grained view of system state and evolution is required that is, at the very least, difficult to describe and analyze using classical equational methods. Indeed, a view of state that is formed in precise, logical terms

and that evolves via discrete events can often be more useful. Moreover, preferences regarding choices of security controls may depend crucially upon (fine-grained, logical properties of) the states visited by the system as it evolves, since these may determine, for example, whether a particular exploit is (or can be) realised.

Simulation modelling is a key tool for exploring and reasoning about complex dynamical systems. Many languages and tools for simulation available. In particular, it can be applied in the fine-grained situations alluded to above.

We describe a mathematical framework that supports a modelling idiom based on the core concepts of process, resource, and location, and which also supports stochastic methods for representing environments. In this section, we draw directly upon the content of [2].

2.1 Modelling Methodology

It is often difficult to validate models of complex systems. Indeed, there are important questions about the faithfulness of the representation of the underlying system and, so, about the extent to which models can be reliably/usefully predictive. These concerns suggest that it is appropriate to use an approach based on the disciplined use of small, expressive, languages that have a formal semantics and which are implemented with a high-degree of integrity, employing constructs that naturally support the modelling idiom. Such a language, Core Gnosis, is described in Section 4 below. We intend to use an implementation of this type of mathematical approach, however we intend to implement it in a more easily cross compatible platform, in this case we have chosen the software package MatLab, which is commonly used in industry and academia.

It is useful to argue — see, for example, [4] — that the key structural aspects of systems are the ones discussed below. This point of view is consistent with the classical view of distributed systems, as described, for example, in [5].

Process. Synthetic systems exist in order to deliver services (i.e., processes that execute on the system's architecture).

There are number of familiar aspects of the intuitive notion of process that we might naturally want our model of process to capture. These include sequencing, choice, concurrency, recursion, and others.

Resource. A system's resources, relative to which the system's processes execute, consists of a collection of quantities that may be utilized by the processes in order to achieve their intended purposes.

Recent work on resource semantics (see, for example, [18, 19]) suggests that capturing the idea that resource elements may be combined and compared is sufficient for a great deal of progress to be made.

Location. Generally, system architectures are highly distributed, either logically, physically, or both. System resources are distributed around a collection of locations, and locations have connections between them.

Environment. Systems exist within external environments, from which events are incident upon the system's boundaries. Typically, the environment is insufficiently understood and too complex to be represented in the same, explicit, form as the system itself.

We represent the impact of the environment on the system of interest as the incidence of

random events upon the system's boundary. Some internal components of systems, whose detailed form and operation is unimportant for the model, may be treated as environmental.

2.2 Structure and Process Calculus

We describe a mathematical treatment of the core system components discussed above.

2.2.1 Processes and Resources

We give a brief review of the process calculus SCRP [3] of resources and processes (which builds on and consolidates [20, 21]) and its extension to locations [4].

Our starting points are Milner's synchronous calculus of communicating systems, SCCS [22], perhaps the most basic of process calculi, and the resources semantics of bunched logic [23, 18]. The key components for our purposes are the following:

- A monoid of actions, Act, with a composition ab of elements a and b and unit 1;
- The following grammar of process terms, E, where $a \in \mathsf{Act}$ and X denotes a process variable:
$$E ::= a : E \mid \sum_{i \in I} E_i \mid E \times E \mid X \mid fix_i X.E \mid (\nu R)E.$$

Most of the cases here, such as action prefix, sum, concurrent product, and recursion (in the fix_i case, X and E are tuples, and we take the ith component of the tuple), will be quite familiar to theorists. The term $(\nu R)E$, in which R denotes a resource, is called hiding, is available because we integrate the notions of resource and process. Its meaning is discussed below; it generalizes restriction.

Our mathematical treatment of resource — encompassing natural examples such as space, money, and computer memory — is based on ordered, partial, commutative monoids; for example, the non-negative integers with addition, zero, and less-than-or-equals.

- Each type of resource is based on a basic set of resource elements;
- Resource elements can be combined (and the combination has a unit);
- Resource elements can be compared.

Formally, we take pre-ordered, partial commutative monoids of resources,

$$(\mathbf{R}, \circ, \mathbf{e}, \sqsubseteq),$$

where $\mathbf{R}$ is the carrier set of resource elements, $\circ$ is a partial monoid composition, with unit $\mathbf{e}$, and $\sqsubseteq$ is a pre-order on $\mathbf{R}$.

The basic operational semantics idea is that resources, R, and processes, E, co-evolve,

$$R, E \xrightarrow{a} R', E',$$

according to the specification of a partial 'modification function', $\mu : (a, R) \mapsto R'$, that determines how an action a evolves E to E' and R to R'.

The base case of the operational semantics is given by action prefix:

$$\frac{}{R, a : E \xrightarrow{a} R', E} \qquad \mu(a, R) = R'.$$

Concurrent composition, $\times$, uses the monoid composition (written as $\circ$) on resources,

$$\frac{R, E \xrightarrow{a} R', E' \quad S, F \xrightarrow{b} S', F'}{R \circ S, E \times F \xrightarrow{ab} R' \circ S', E' \times F'}.$$

Modification functions are required to satisfy some simple coherence conditions:

- $\mu(1, R) = R$, where 1 is the unit action, and
- if $\mu(a, R) \circ \mu(b, S)$, and $R \circ S$ are defined, then $\mu(ab, R \circ S) \simeq \mu(a, R) \circ \mu(b, S)$.

Here $\simeq$ is Kleene equality. In some circumstances, additional equalities may be required [3, 4]. The other process constructs are treated similarly.

2.2.2 Location

Just as our treatment of resources begins with some basic observations about some natural and basic properties of resources, our treatment of a useful notion of location starts with the following basic requirements [4, 2, 1]:

- A collection of atomic locations, which generate a structure of locations;
- A notion of (directed) connection between locations, describing the topology of the system;
- A notion of sublocation (which respects connections);
- A notion of substitution (of a location for a sublocation) that respects connections, providing a basis for abstraction and refinement in our system models.

The resulting calculus has transition systems with dynamic behaviour of the following form:

$$L, R, E \xrightarrow{a} L', R', E',$$

where a is an action (in the usual process sense), L, L' are location environments, R, R' are resource environments and E, E' are processes used to control the evolution. Modification functions are extended to include locations, $\mu : (a, L, R) \mapsto (L', R')$, with corresponding versions of the coherence conditions.

The following is the rule for action prefix:

$$\frac{}{L, R, E \xrightarrow{a} L', R', E'} \; Action$$

where $(L', R') = \mu(a, L, R)$.

The following quite general form of the product rule in the presence of locations makes use of a notion of product of locations:

$$\frac{L, R, E \xrightarrow{a} L', R', E' \quad M, S, F \xrightarrow{b} M', S', F'}{L \bullet M, R \circ S, E \times F \xrightarrow{a \cdot b} L' \bullet M', R' \circ S', E' \times F'} \; Product$$

where $\bullet$ is the product of locations. Various simpler forms, such as taking a fixed location, make sense in absence of a product of locations [4, 1]. We can also take a Frame rule (with respect to resources):

$$\frac{L, R, E \xrightarrow{a} L', R', E'}{L, R \circ S, E \xrightarrow{a} L', R' \circ S', E'}$$

provided $\mu(a, L, R \circ S))$ is defined.

This approach stands in contrast to that of Milner and others, in which a single language representing all of the modelling constructs is sought.

2.3 Environment

In our approach to Core Gnosis below, the environment is handled stochastically. In constrast to the work of Hillston et al. [24], our approach — in the spirit of the denotational semantics of programming languages — is to develop our semantic structures in parallel with our modelling language, Core Gnosis. The two are then related by an interpretation of the modelling language in the structures, about which we seek to establish certain properties.

3. Reasoning about Process Models

3.1 Logic

Process calculi such as SCCS, CCS, and the pi-calculus come along with associated modal logics [25, 26, 27]. Similarly, the calculus of Section 2.2 has an associated modal logic, MBI [3, 20, 21]. The basic idea — deriving from Hennessy-Milner logic [25, 27] — is to work with a logical judgement of the form $R, E \models \phi$, which is read as follows: relative to the available resources R, the process E has property ϕ.

The relationship between truth and action is captured by the clauses of the satisfaction relation for the (additive) modalities, given essentially as follows (recall that $R' = \mu(a, R)$):

$$R, E \models \langle a \rangle \phi \quad \text{iff} \quad \text{there exists } E' \text{ such that } R, E \xrightarrow{a} R', E' \text{ and } R', E' \models \phi$$

$$R, E \models [a] \phi \quad \text{iff} \quad \text{for all } E' \text{ such that } R, E \xrightarrow{a} R', E', \; R', E' \models \phi.$$

In this setting, however, the multiplicative conjunction, $*$, that is available in bunched logic [23, 28]provides a characterization of this judgement that is rather finer that which is available in basic Hennessy-Milner logic. Specifically, we obtain the following characterization of the concurrent structure of models:

$$\begin{array}{lll} R, E \models \phi_1 * \phi_2 & \text{iff} & \text{there are } R_1 \text{ and } R_2 \text{ such that} \\ & & R_1 \circ R_2 \sqsubseteq R \text{ and there are } E_1 \text{ and } E_2 \\ & & \text{such that } E_1 \times E_2 \sim E, \text{ and} \\ & & R_1, E_1 \models \phi_1 \text{ and } R_2, E_2 \models \phi_2. \end{array}$$

The truth condition for the multiplicative conjunction requires the combination of resources from the truth conditions for its component formulae. The meaning of the usual, classical/intuitionistic additive connectives — $\top$, $\wedge$, $\rightarrow$, $\bot$, and $\vee$ — is discussed in [3, 4, 2, 1], along with the multiplicative implication corresponding to $*$. All of the process combinators described above obtain logical interpretations.

With locations, a similar logical judgement is available [4]: $L, R, E \models \phi$, where the property ϕ of the process E holds relative to resources R at location L; that is, if a is an action guarding (the rest of) E, then $\mu(a, L, R)$ is defined.

3.2 Model Checking

Model checking is an important technique in computational logic. It addresses the following problem: For a given system of logic, does a given model $\mathcal{M}$ satisfy a given proposition ϕ? That is, for the given logic's satisfaction relation $\models$, does $\mathcal{M} \models \phi$ hold? Here we are assuming that the model $\mathcal{M}$ is a form of transition system, as is the case for the logics we have set up in this work. Model checking provides a valuable component of the logical process approach to systems modelling; that is, the ability to check that the correctness of assertions about the state of a model (albeit for now only quite simple ones). A comprehensive collection of references on the techniques of model checking is available in [29, 30]. A degree of model checking is available for the process calculus of Section 2.2 and the logic MBI has been implemented [1].

4. The Gnosis Modelling Tool

We introduce very briefly the *Core Gnosis* modelling language via a series of examples to illustrate the disciplined approach to modelling discussed above.

Core Gnosis includes constructs for describing processes, resources, and locations that capture many (though, at this stage, not quite all) aspects of the mathematical structures described above.

Birtwistle's [31] provides a classic example: the docking of boats in a harbour with various jetties and tugs, which we extend to include secure docking of boats [2, 1]. Here is a first, simple version in Core Gnosis, taken from [2, 1].

```
-- title : Boats example : time units = hours
-- seed = 426724262

param runTime = (24 * 7)                                                    // 7 days

param numJetty = 2; param numTug = 3

param dockTime    = negexp(2.0)
param undockTime  = negexp(1.5)
param unloadTime  = uniform(1.0, 4.0)

param boatMeanArrival = 10.3
param boatDelay       = negexp(boatMeanArrival)

share jetty numJetty
share tug   numTug

process boat = {
  claim 1 jetty
    claim 2 tug; hold(dockTime); release 2 tug
      hold(unloadTime)
    claim 1 tug; hold(undockTime); release 1 tug
  release 1 jetty
  hold (boatDelay); launch boat
}

launch boat

hold (runTime)
close
```

The model defines a series of constants and distributions (via the **param** keyword), and some shared resource elements, namely jetties and tugs (via the **share** keyword). A single process corresponding to the boat's activities is then defined (using the keyowrd **process**): getting a jetty, getting some tugs (using **claim**s), docking , unloading, (by **hold**ing for a period of time) and so on. An instance of a boat is (immediately) **launch**ed and the simulation then runs for 168 (= 24×7) (simulated) hours before closing. Each of these keywords has a quite intuitive meaning, but more formally the semantics of models is defined by a scheduler [1].

4.1 Adding located resource: secure boats

We illustrate location by adding security properties to the simple boats example.

There are two kinds of dock, Basic and Secure, together with a Guard area to access of control to the Secure docking area. There is a pool of tugs that can be moved between the Basic and Guard locations and from Guard to Secure and and back to Basic. The first part of the model gives the parameters:

```
param runTime = (24 * 7)                                                    // 7 days

param numJetty       = 2; param numTug       = 3
param numSecureJetty = 1; param numSecureTug = 3

param dockTime    =  weibull(2.0, 1.5)
param undockTime  =  weibull(1.5, 1.5)
param unloadTime  =  uniform(1.0, 4.0)

param checkTime   =  weibull (2.0, 3.0)
```

```
param passCheck   =  normal (1, 0.5)
param passLevel   =  0.5

param boatMeanArrival = 10.3
param boatDelay = negexp (boatMeanArrival)

param secureBoatMeanArrival = 18.9
param secureBoatDelay = negexp (secureBoatMeanArrival)

param checkInterval = 3.5
param checkDelay = negexp (checkInterval)
```

With these parameters in place, we can introduce *locations* and *links* between locations. The next part of the model defines the shared, located resources that are needed:

```
location  Basic, Guard, Secure
link      Basic ↔ Guard → Secure → Basic

share jetty@Basic   numJetty
share jetty@Secure  numSecureJetty

share tug@Basic     numTug
share tug@Secure    numSecureTug
```

There are now two kinds of boat, Standard and Secure. Standard (or low-security) boats can only use the Basic jetties whereas secure (or high-security) boats can only use the Secure jetties. Each tug can be used to dock/undock the boats in either dockingarea. Tugs may need to change their rôle/location and move from one to the other as determined by the dock's operational requirements.

Here is the standard boat process:

```
process boat = {
  claim 1 jetty@Basic

    select [claim 2 tug@Basic] {
      hold(dockTime)
      release 2 tug@Basic
    }

    or [claim 2 tug@Guard] {
      move share (2) tug@Guard → tug@Basic
      hold(dockTime)
      release 2 tug@Basic
    }

    or [claim 2 tug@Secure] {
      move share (2) tug@Secure → tug@Basic
      hold(dockTime)
      release 2 tug@Basic
    }

    hold(unloadTime)

    select [claim 1 tug@Basic]  {
      hold(dockTime)
      release 1 tug@Basic
    }

    or [claim 1 tug@Guard] {
      move share (1) tug@Guard → tug@Basic
```

```
      hold (dockTime)
      release 1 tug@Basic
    }

    or [claim 1 tug@Secure] {
      move share (1) tug@Secure → tug@Basic
      hold (dockTime)
      release 1 tug@Basic
    }

  release 1 jetty@Basic

  hold (boatDelay); launch boat
}
```

Notice that tugs are initially claimed from either the Basic, Guard, or Secure pools and, if necessary, moved into the Basic pool. Our version of **move** can only move resources *already* owned by the process (i.e., claimed) from one location to another, and can only do so along a valid link between the two locations. Once a resource is moved to a new destination, it must also be released back to that new location, not to the one from which it was claimed.

Here is the secure boat process:

```
process secureBoat = {
  claim 1 jetty@Secure
    select [claim 2 tug@Secure] {
      hold (dockTime)
      release 2 tug@Secure
    }
    or [claim 2 tug@Guard] {
      move share (2) tug@Guard → tug@Secure
      hold (dockTime)
      release 2 tug@Secure
    }

    hold (unloadTime)

    select [claim 1 tug@Secure] {
      hold (undockTime)
      release 1 tug@Secure
    }
    or [claim 1 tug@Guard] {
      move share (1) tug@Guard → tug@Secure
      hold (undockTime)
      release 1 tug@Secure
    }
  release 1 jetty@Secure

  hold (secureBoatDelay); launch secureBoat
}
```

The next process performs the 'randomized inspection' of tugs — the check process takes either one or two tugs in Basic and 'decides' (via a distribution) whether or not to inspect. The tugs always end up in the Guard area:

```
process check = {
  select [claim 1 tug@Basic] {
    move share (1) tug@Basic → tug@Guard
```

```
    if [passCheck > passLevel] {hold(checkTime)} or else {}
    release 1 tug@Guard
  }

  or [claim 2 tug@Basic] {
    move share (2) tug@Basic → tug@Guard

    if [passCheck > passLevel] {hold(checkTime)} or else {}
    release 1 tug@Guard

    if [passCheck > passLevel] {hold(checkTime)} or else {}
    release 1 tug@Guard
  }

  hold(checkDelay); launch check
}
```

Finally, we launch all three processes, *boat*, *secureBoat*, and *check*, to perform the overall simulation:

```
launch boat
launch secureBoat
hold(checkDelay); launch check

hold (runTime)
close
```

The evolution of the Core Gnosis abstract machine determines the observable change of state recorded by the trace (history).

The language also allows for statements of the form **forget**(l, l') and **recall**(l, l'), where l and l' are simple locations. These statements make the topology of the system dynamic, in that processes may not be able to use the declared links at all points in time. The **forget**(l, l') statement changes the system state by dropping the link from l to l'. Note that **move** statements taking resources from l to l' will block when the link is thus broken. A **recall**(l, l') statement re-connects the link from l to l'. A process which is blocked on a **move** from l to l' will be un-blocked when this link is recalled. For example, one may wish to consider enriching the Secure Boats example, so that the tugs kept at the `Guard` location are, periodically, distrusted. This may be represented by having the link from `Guard` to `Secure` forgotten and recalled periodically.

5. Alternative Simulation Approaches

There are many simulation and modelling tools in existence. In this section we discuss just some of them. We begin with a quick summary of the main requirements for our modelling language. We then discuss the alternatives before concluding this section with a summary.

5.1 Requirements

The precise requirements of any modelling language depend upon its intended uses. Our past experience and the domain examples proposed for Seconomics lead us to the requirements presented in this section.

The purpose of the modelling language is to enable policy-makers to determine their preferences with respect to their possible policy choices. Thus:

> The modelling language must be able to represent policies and to express preferences between policies.

The disciplined representation of policy is an important research topic to be addressed in this project, although past experience has shown that policy certainly can be represented in both Core Gnosis and MATLAB. However, preferences are often discovered after the behavioural response of the wider system to policies is known. We therefore turn to such behavioural aspects next before returning to a more detailed description of preference.

Security is a multi-scale problem: on the one-hand, note that the smallest scale vulnerability in a large system can be transformed into a security threat to the entire system (for example, an exploitation of a code-level vulnerability in an industial control system); on

the other-hand, the properties of interest to decision- and policy-makers are often at the aggregate level (for example, behaviours of large populations of users or markets may determine the view of risk in a cost-benefit analysis). Moreover — and this may not always be appreciated — macro-scale factors (such as the prevalence of one operating system near-monoculture) may affect the micro-scale security factors that matter (since such macro-factors will influence the decisions of attackers to search for particular vulnerabilities). Thus a requirement is:

> The modelling language must be able to represent systems across both micro- and macro-scales.

It is evidently not possible to represent large systems in complete detail. It is also not desirable so to do, for many well-known reasons. Thus, a further requirement is:

> The modelling language must make it possible represent systems at multiple levels of abstraction.

The systems that will be modelled will typically have rich dynamics. Thus:

> The modelling language must be able to express the dynamics of the system, and an executable tool must allow for effective exploration of evolution under such dynamics.

As discussed in earlier sections, the pragmatics of the disciplined construction of systems models requires that the structure of systems (e.g. location and resource) be directly representable, rather than merely encodable. Specifically:

> Appropriate structural features must be representable in the modelling language.

Modelled systems often contain components that evolve or behave in a way that is best described probabilistically; moreover, this is also true of the system environment. Hence:

> The modelling language must enable the expression of stochastic events.

Returning now to the issue of the representation of preference, one of the most common ways that this is done is to evaluate (distributions over) utility values accumulated over the course of model runs for given policy and system configurations. Furthermore, as well as utility expressed over models, models may also contain agents with preferences expressed in terms of utilitites. In a Cournot-Stackelberg (or other game-theoretic) framework, the policy-makers preferences and choices will also be internal to models. Therefore:

> The modelling language must allow for the expression of utility, and its accumulation over intermediate and final system states.

5.2 Alternative Systems

5.2.1 Traditional Applied Mathematical Systems

This class of languages is perhaps the oldest and most-widely used, in high-level form going back at least to FORTRAN, but in low-level form even further. It includes the simulation

capability of MATLAB and also Mathematica. Roughly speaking, these languages encapsulate the dyamics of a system by the use of equations to update a simple notion of state given by variables bearing quantities. Thus they are particularly suited to the mechanical implementation of traditional applied mathematical (and economic) methods. In principle, any of these languages is as powerful as any other. MATLAB offers the advantages of wide and well-documented functionality, high-quality visualization capabilities, but importantly the ability to distribute models as executables.

5.2.2 Systems for Concurrent Modelling

If one cares about the modelling of situations in security where concurrency, synchrony and logical properties of state are paramount (for example, if one needs to model detailed causes and effects on a network during the course of a particular attack) then dedicated concurrency modelling tools should be considered.

Systems for dealing with fine-grained concurrency simulation date back to the development of the SIMULA language. There are now many such simulation packages in existence. MATLAB contais a discrete event simulator with some support for concurrent modelling. This is discussed further in Section 6.

An important offshoot of concurrent modelling languages are those grounded in Theoretical Computer Science, specifically process calculus. These approaches have several principal advantages. The first is compositionality: one can build models of component systems and then combine them in set ways to give larger models. The second advantage is that they have a formal semantics that specifies in a precise, mathematically-tractable way the dynamics of the system. This enables a further capability, namely model-checking, where logical properties of systems can be formally specified and verified with automated tools. PRISM [32] and PEPA [24] both offer the above functionality, including stochastic evolution. Core Gnosis also has its roots in a process calculus approach [1, 2, 4], but adds the representation of structure to the usual dynamics.

5.3 Summary

MATLAB will be the default choice of simulation language for Seconomics WP6. Its power and applicability to the type of models that are likely to be considered make it the most appropriate choice. Although, fine-grained details of concurrency and formality can matter in some models, the economic models that will be the focus in WP6 will not likely be concerned with this abstraction level. Typically, events will be separated by probabilistic timing choices rather than explicitly synchronized. Additional structure of models (e.g. location, resource) can be encoded in MATLAB, as discussed in Section 6.

6. Mathematical Models to Matlab Models via Gnosis

A process modelling language such as Core Gnosis could be implemented in MATLAB. However, this is not necessary for the present purpose, would be extremely time-consuming, difficult to get correct, and will not be done. Instead, only models of a type and at a level of

abstraction suited to MATLAB modelling will be considered. Where appropriate, the MATLAB models will nevertheless be structured in a way that retains some of the central commitments of the Core Gnosis view.

The dynamics of systems will be encoded through updates to state and the logical control operations of the MATLAB language. One way that this can be done is with the Simulink library (http://www.mathworks.co.uk/products/simulink/) which allows for a degree of compositional structure in models using block-diagrams.

Simulink includes a discrete-event simulation language called SimEvents (http://www.mathworks.co.uk/products/simevents/). This allows for concurrently evolving entities, discretely evolving with stochastic events, and shared resources to allow for blocking of entity evolution. The representation of concurrency is through the use of a shared event list to schedule the next event issued by processes, and communication between processes via such events [33].

The treatment of location in Core Gnosis is based on the mathematical notion of graph: a set of nodes connected by edges. It would be easy to program tools to support such uses of graphs directly MATLAB, but graphs are already supported within the Matlab Symbolic Math Toolbox (http://www.mathworks.co.uk/help/symbolic/graph-theory.html).

In Core Gnosis resources are modelled as particular kinds of variable. There are two important things to note about such variables: they collectively form an easily comprehensible part of the system state; they are subject to clear disciplines regarding their use and update. The first point will generally be true of variables in many modelling tools (including MATLAB, but excluding certain process modelling languages). The disciplined variables that Core Gnosis use include ‘shares’, ‘bins’, ‘budgets’ and ‘tallies’. In particular, shares can be sited at locations and move between them. Resources can easily be viewed as located by simply having a separate resource at each location. The mobility of resources is another matter. Should this prove to be needed, it seems that it will have to be done in an ad-hoc way using unstructured MATLAB variables.

7. An Architectural Methodology for Organizational Security Models

In this section, drawing directly upon [34, 35], we introduce a methodology for structuring systems security models. A key idea underlying this methodology is to draw a clear distinction between declarative and operational security concepts. A structure called a *Security Architecture* is introduced that allows one to describe hierarchies of organizational rôles, declarative security objectives for those rôles, and operational components used to achieve those goals. We consider how methods from economics can be used to inform design choices.

The methodology was developed in the context of information security, and the discussion below motivates it in these terms. However, the same methodology can easily be applied to other security modelling situations, as can easily be seen from the example given below concerning airport security.

7.1 The Basic Concepts of Information Security

The fundamental concepts of information security are confidentiality, integrity, and availability (CIA). Alongside these notions sits the concept of privacy. We are not concerned with privacy here.

It is, however, commonplace in the literature to find observations that CIA does not provide an adequate basis for practical, operational, information assurance. Typically, it is suggested to extend CIA with various additional concepts, such as 'authentication', 'non-repudia(tion)(bility)', 'control', or even 'utility'. Perhaps the leading, most developed example is the 'Parkerian Hexad' (as developed in Parker's elegant account of security concepts [36]), in which to

- confidentiality,
- integrity, and
- availability

are added

- possession,
- authenticity, and
- utility.

These concepts, which it has been argued are 'atomic' and 'non-overlapping' are, indeed, all conceptually valuable and pragmatically useful in the understanding and practice of information security.

A similar collection of concepts may be found, for example, in the 'ISO/IEC 7498-2: Information Technology—Open Systems Interconnection—Basic Reference Model—Part 2: Security Architecture', which identifies identification and authentication, access control, data integrity, data confidentiality, data availability, auditability, and non-repudiation.

In the Parkerian Hexad and the ISO/IEC Reference Model, however, as in similar taxonomies, the proposed extensions to CIA constitute, in the Aristotelian sense, category errors. They confuse the (declarative) objectives of information security operations with the (operational) mechanisms deployed in order to achieve those objectives. For one example, access control is an operational notion used, for example, to restrict the availability of a service to a given group of users. For another, 'auditability' , 'authenticity', and 'non-repudiation' are properties — of the kind that might be expressed logically, as discussed below (in 3) — of the underlying systems security architecture. For yet more, possession seems to be a notion that is derivable (at least according to Parker's definition) from confidentiality, integrity, and availability, whilst utility is, evidently, not a security concept, but rather an economic concept that is useful in the security context

This situation is problematic not only from the conceptual point of view — because declarative and operational concepts must be treated differently in order to understand how objectives are delivered (or not) by making (in)appropriate implementation choices — but also from the economic and management points of view — because we are concerned with how the objectives of information security measures trade off against one another.

There are also evident redundancies — as discussed in, for example, [36] — though this issue is not our primary concern here.

7.2 An Economic View

Utility is an economic concept. Utility theory (see, for example, [37]), particularly as developed in the contexts of macroeconomics and financial economics, provides a highly expressive framework for representing the preferences of the managers of a system.

For example (e.g., [38]), in the macroeconomic management of market economies, central banks play a key rôle. The managers of a central bank are given, by their national governments, targets for certain key economic indicators, such as unemployment (u_t) and inflation (π_t) at time t (time can be either discrete or continuous here). Their task is to set a (e.g., monthly) sequence of controls, such as their base (interest) rates (i_t) so that the key indicators are sufficiently close to their targets, $\overline{u_t}$ and $\overline{\pi_t}$, respectively. Typically, using this example, the managers' policy is expressed as a utility function

$$U_t \quad = \quad w_1 f_1(u_t - \overline{u_t}) + w_2 f_2(\pi_t - \overline{\pi_t}) \tag{1}$$

together with system equations, $u_t = s_1(i_t)$ and $\pi_t = s_2(i_t)$, expressing the dependency (among other things) of u and π on interest rates in terms of functions s_1 and s_2 that describe the (macro) dynamics of the economy. Two key components of this set-up are the following:

- The weights w_1 and w_2 (typically, values between 0 and 1) that express the managers' preference between the components of the utility function — that is, which they care about more; and
- The functions f_1 and f_2 that express how utility depends on deviation from target. A simple version of this set-up would take the f_is to be quadratic. Quadratics conveniently express diminishing marginal returns as the indicators approach target, but make utility symmetric around target. More realistically, Linex functions [39, 40, 38], usually expressed in the form $g(z) = (exp(\alpha z) - \alpha z - 1)/\alpha^2$ are used to capture a degree of asymmetry that is parametrized by α.

The managers' task is to set a sequence of interest rates i_t such that the *expected* utility, $E[U_t]$, remains within an acceptable range, as u_t and π_t vary, and trade-off against each other, as the sequence of rates i_t evolves. In general, there can of course be as many components as required in a utility function.

This economic framework can be deployed in the context of information security (see, for example, [11, 12, 13, 41]), where concepts uch as confidentiality, integrity, and availability that lie within competing declarative categories can be seen to trade-off against one another as the relevant controls, such as system configurations or investments in people, process, and technology system configurations, vary:

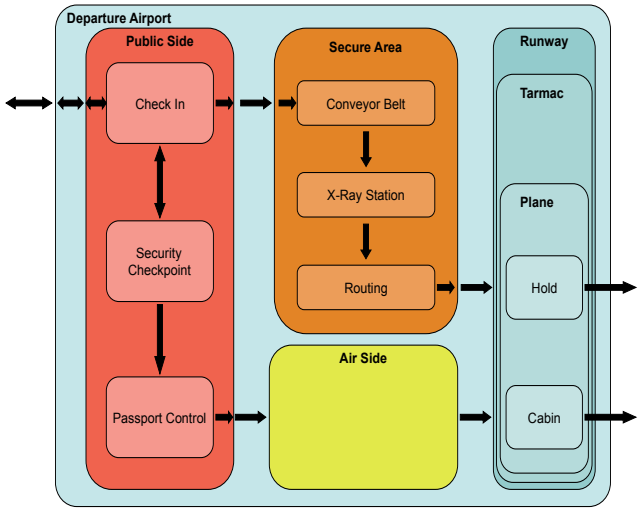
- The organization that deploys information security measures exists in an economic and/or regulatory environment. This environment places constraints upon the systems and security architectures available to the organization's managers.
- The managers formulate a utility function that expresses their policy preferences, which will depend upon the nature of their organization. For example, state intelligence agencies and online retailers will have quite different priorities among confidentiality, integrity, and availability; see, for example, [12].

- In a highly complex situation, such as a security architecture, it will typically not be possible to formulate system equations (in terms of functions s_1 and s_2) in the way that is usually possible in, for example, macroeconomic modelling. Typically, though, the key control variables, such as system interconnectivity or investment in various aspects (people, process, and technology) of security operations, will be identifiable.
- Instead, however, an executable system model, such as Core Gnosis [1] as described in Section 4, using the key control variables, can be used in order to simulate the dynamics of the system and the utility function.

As an example, consider the management of an airport's security process, see Deliverable D2.3 for WP2 requirements on airport security.A key aspect of this is checking passengers and their bags for acceptability to fly. A passenger, with luggage, must navigate from the concourse of an airport's terminal building to a seat on an aircraft; that is, it is an access control process that is predicated on maintaining an *integrity* property of aircraft, and this is achieved by maintaining that property for passengers and their baggage. This integrity property trades-off against other concerns, principally costs, incurred in providing security staff and equipment, and service availability.

A picture of the access control system is given in Figure 1. Note how location and resource are used to structure and control the process, and recall that these were identified in Section 2.1 as key modelling components.

Figure 1: Airport Locations



The passenger is subject to a range of security controls that are intended to ensure a certain integrity property — roughly, that certain dangerous or substances and objects are not present — of the aircraft. The manager of the security process has decided that, in order to access the aircraft, passengers must submit to the security process and must therefore sacrifice their confidentiality — the bags will be searched, there will be body searches. Thus

the manager has given a preference weighting of 0 to passengers' confidentiality. There are, however, non-trivial trade-offs with cost and availability:

- *Cost*. The effectiveness of the integrity might be improved by, for example, introducing more expensive scanning devices that are able to detect more things, more reliably. The efficiency of the integrity check might be improved by introducing more scanning devices and more security staff, thereby facilitating greater parallelism in the security process;
- *Availability*. An important measure of availability[2] of access to the aircraft is the length of time that must be allocated for passengers to navigate the airport's security procedures.

We can, of course, consider very different points of view. From the point of view of a smuggler, the utility function might look rather different. She might submit to the process without deciding to sacrifice her confidentiality — her preference weighting for confidentiality is not 0, but rather is something close to 1 — hoping to conceal her contraband by some means. We will not develop this part of our example here because it is not concerned with the integrity property of the aircraft. Rather, it is concerned with an integrity property of an international boundary (be it outgoing or incoming), and, in practice, may or may not be considered within the security process.

7.3 Modelling the Security Architecture

We introduce our conceptual account of security architectures, its purpose being to give a structured, conceptual description of the components of a security architecture that can naturally be integrated with the natural structure of executable system models.

There are two key layers in our representation of a security system, the *Framework* layer and the *Instantiation* layer. There is a commonality of organization between these layers although they represent conceptually different parts of the model. Both layers are organized into a hierarchy of rôles with each rôle sub-divided into dependencies, priorities, and preferences.

The hierarchy contains all the relevant rôles that make up the organization being modelled. Rôles are ordered by their ability to influence the security architecture of the system. In other words, they are classified by the toolbox that is available to them for modifying *security objects* (that characterize security tasks, defined below). The system accepts multiple and partial orderings. For example, the top level of the model might represent the strategic decision-makers of the organization, such as an airport's security managers or their regulators, while the bottom level might represent an individual employee or user of the organization, such as an airport's check-in staff or a passenger navigating airport security. The rôles represent the possible positions individuals can adopt in the hierarchy. They do not represent any entity themselves. They are instead populated by *actors*, which are another component in system and are described below.

Each hierarchy level contains three sections representing the dependencies, priorities and preferences of that level. For our purposes we define the terms as follows:

[2] Recall that one reasonable definition of availability is along the lines of 'accessibility of service when required' [36].

- *Dependencies* (strong requirement): Externally enforced requirements that actors populating the rôle must meet all of in order to function within the model. Actors occupying this rôle have no choice in whether or not (and possibly even how) to meet these requirements regardless of how resource inefficient they are. Dependencies will often be informed by the environment within which the hierarchy exists;

- *Priorities* (weak requirement): Externally supplied tasks, as many as possible of which should be met by Actors in the associated rôle. Actors have some choice in which priorities to meet and how they are approached. In a limited resource environment, Actors can select the most resource efficient priorities and methods first. Priorities will often be informed by the rôle that the level represents;

- *Preferences*: Actor-generated tasks that the Actor has decided are worth achieving from its own perspective. These can be generated by the Actor's inclusion in other hierarchies.

Dependencies, priorities, and preferences (DPP) and the hierarchy of rôles structure are found in both the framework layer and the instantiation layer.

The form and construction of the security architecture is illustrated in Figure 2.

The key components of this diagram are the following:

- *The hierarchy of rôles* (far left). Rôles capture the relevant security management structure of the organization being modelled. They are ordered by their ability to influence the security architecture of the system;

- *The Framework Layer* (centre left). The Framework Layer is constructed top-down. Dependencies and priorities at a given level in the hierarchy induce dependencies and priorities at lower levels;

- *Security Objects* (trees within Framework Layer). Security Objects represent the security tasks which, if completed, will satisfy the dependencies and priorities with which they are associated;

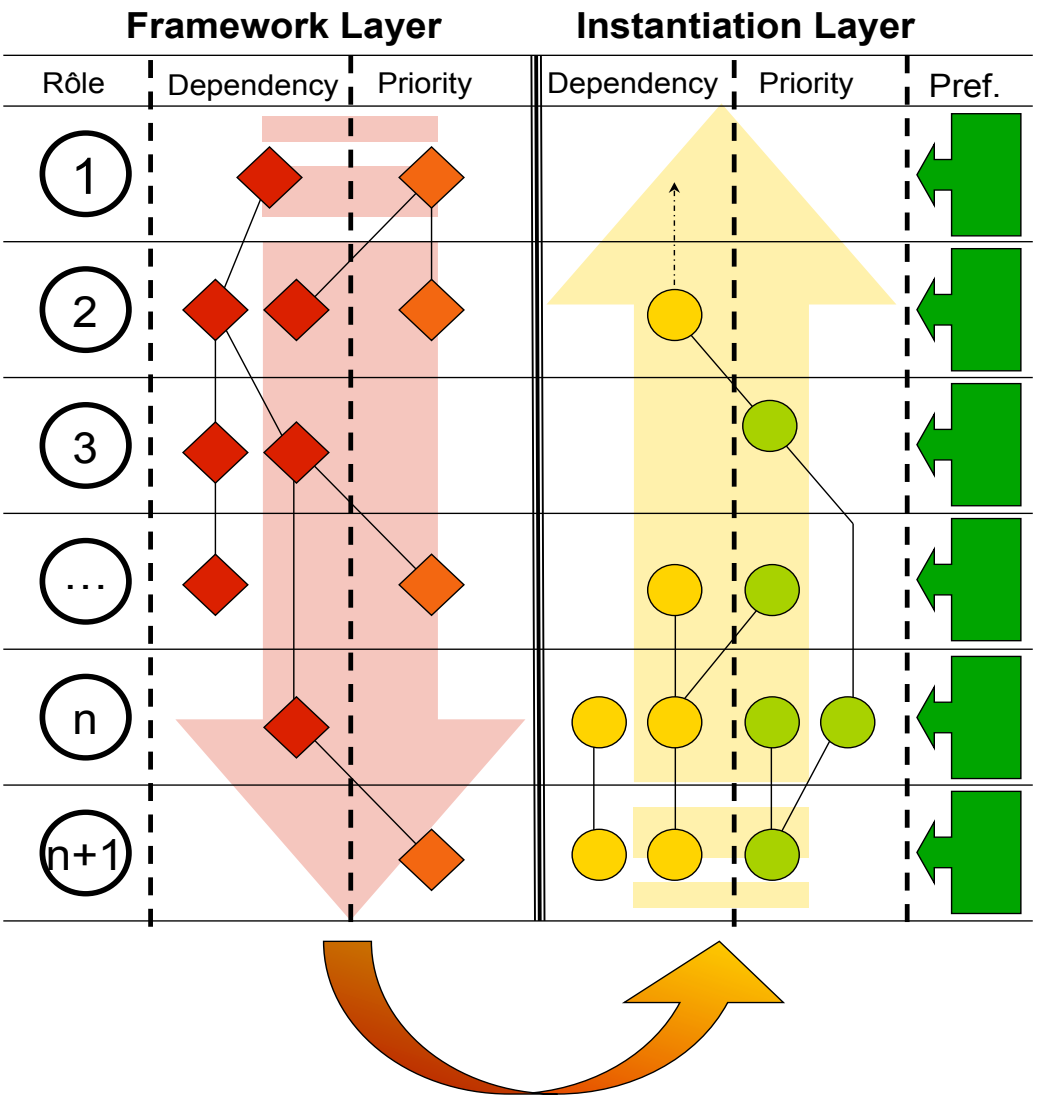
- *The Instantiation Layer* (centre right). The Instantiation Layer is constructed bottom-up, starting where the Framework Layer finishes (see below). The Instantiation Layer is a populated image of the Framework Layer;

- *Security Components* (nodes of trees within Instantiation Layer). Security Components perform the operational checks required in order to deliver Security Objects. They do so by return boolean values up the tree, towards the root. They enter the architecture when the Framework is instantiated;

- *Actors* (far right). Actors occupy rôles. They insert preferences into the hierarchy of rôles at the Instantiation Layer.

A key point here concerns the way in which the dependencies, priorities, and preferences in model are intimately related to the declarative security concepts. For example, given confidentiality, integrity, and availability as the concerns, the dependencies, priorities — in the Framework Layer — express the policies required to implement the managers' preferences

Figure 2: Security Architecture: Framework and Instantiation



as represent in their chosen utility function for the organization. Specifically, given a utility function with definiens of the form

$$w_1 f_1(C - \overline{C}) + w_2 f_2(I - \overline{I}) + w_3 f_3(A - \overline{A}) + w_4 f_4(K - \overline{K})$$

where C, I, and A denote suitable instances and/or measures (see, for example, [12]) of confidentiality. integrity, and availability, respectively, and where K denotes cost or investment, dependencies will have very high weightings, with very little tolerance for deviation from target, and priorities slightly lower weightings. Preferences (see below) have weightings that lower still, and may have high tolerance for deviation from target.

The form and function of the Framework and Instantiation Layers, and the interaction between them, will now be discussed in more detail.

7.3.1 The Framework Layer

The Framework Layer represents the underlying structure of the system. It is static (in the sense that the model does not run on this layer) and declarative but informs the construction of the more operational Instantiation Layer. A completed Framework Layer consists in a hierarchy of rôles (see, for example, [**?**]) with dependencies and priorities assigned to them. As preferences are derived from actors (see below), they do not appear in this layer because actors appear in the Instantiation Layer. The dependencies and priorities will each have a *Security Object* (SO) assigned to them. SOs are a unique component of the Framework Layer and represent the security tasks which, if completed, will satisfy the dependencies and priorities with which they are associated.

For example, in the setting of the running example of airport security that we have begun to introduce, examples of Security Objects include the examining of checked luggage, the checking of hand luggage and passengers — to identify and so remove any prohibited contents — and the tracking of the relationship between passengers and checked luggage. These examples are developed below.

SOs are unconstrained with respect to their location within the hierarchy. SOs can only exist in one hierarchy and never populate multiple hierarchies. This is a key difference between Actors and SOs. One of the aims of this formulation is to improve the communication between different stakeholders and eliminate the duplication caused by the failure to understand the connectedness of security concepts between levels. In practice, that means a typical SO will exist at multiple levels and multiple sections (dependency, priority) in the Framework. It will commonly be the case that an SO created at a higher level will transition through and connect (or create) priorities and dependencies lower in the framework.

For the more mathematically minded reader, there are many choices of formalization of SO. Our working choice for the purposes of this section is, roughly speaking, the following:

- SOs are characterized by (directed) and/or forests[3] [4] (illustrated in Figure 2 by the red/orange trees in the Framework Layer) associated with dependencies and priorities;
- Internal nodes of the trees are labelled with boolean variables, each associated with a dependency or priority, and truth conditions are inherited upwards (towards the root);

[3] A (directed) forest is a disjoint union of (directed) and/or trees.

[4] A forest is required because a given SO may, in general, derive from more than one dependency or priority.

- Leaves are nodes for which a boolean instantiation (all components for conjunctions, one component for disjunctions) can be determined at the next level down in the hierarchy of rôles.

The Framework is populated with dependencies, priorities, and security objects through an iterative process that requires design input from an expert source. The criteria under which security objects terminate and by which the Framework can be said to be complete is the same for all frameworks created in this way.

As indicated above, dependencies and priorities are externally generated. In practice, a hierarchy of rôles will not encompass all possible contributors to the framework and will have been bounded at some sensible level. In our example, we have not represented any rôle higher than the airport security manager in our hierarchy. The creation and bounding of the hierarchy of rôles is the first step in creating a framework layer. To populate a framework, it is necessary to determine the dependencies and/or priorities that the top rôle in the hierarchy will inherit from sources external to the hierarchy. At this stage, we will have a complete hierarchy of rôles that is empty of dependencies and priorities except for the top layer. The next step is to assign security objects to these dependencies and priorities that will allow them to be fulfilled; see Table 1.

Table 1: SOs Stage 1

Rôle	**Dependencies**	**Security Objects**
Airport Security Manager	Ensure no prohibited materials transit the airport	Scan checked luggage Scan hand luggage and passengers Track relationship between passengers and checked luggage

At this point the iterative process begins. The construction of a framework always proceeds from top to bottom. At each iteration the following are checked:

- Are there any dependencies or priorities without an assigned Security Object?
- Are there any unterminated Security Objects?

The construction of a Security Object terminates once it is possible to return a boolean value from its lowest point. If this is not possible, then the Security Object must be extended to the rôle below in the hierarchy, creating any necessary dependencies and priorities as it does so. The dependencies and priorities created will be informed by the rôle creating them; thus, as the SO descends through the hierarchy, it will become more detailed as the scope of the lowers is necessarily more limited. In our example above none of the security objects can return a value and thus need to be extended. Let us extend the ‘scan hand luggage’ SO. The following (Table 2) would be the result of two iterations, one would generate the dependency

Table 2: SOs Stage 2

Rôle	Dependencies	Security Objects
Airport Security Manager	Ensure no prohibited materials transit the airport	Scan checked luggage Scan hand luggage and passengers Track relationship between passengers and checked luggage
Airport Security Staff	Examine all passengers and luggage passing through security checkpoint	Identify contents of hand luggage and verify permitted

from the SO above, the second would find a dependency without an assigned SO and create one:

Note, for example, that the SO 'scan and luggage and passengers' corresponds to a tree (red/orange in Figure 2) in the Framework Layer.

At this point the SO can return a boolean (true/false that the contents of the bag are permitted) and will terminate. The framework is not yet complete, however, as there are still unterminated SOs at the manager layer. Iterating in this fashion would also close those at a suitable point. The final step in a SO is always a compliance step which indicates that at this level and below the rôles in the hierarchy simply comply with the SO and are not involved in its execution. This would add the following line (Table 3) to the framework:

Table 3: SOs Stage 3

Rôle	Dependencies	Security Objects
Passenger	Comply with SO	

Once the framework is complete under the criteria outlined above the SOs will form a Boolean forest with the leaves connecting each dependency and priority in the framework. At this point we can begin to construct the instantiation layer.

7.3.2 The Instantiation Layer

Whereas the framework layer is static and declarative the instantiation layer is dynamic and operational. Two new parts of the architecture are added during instantiation, Security Components (SC) and Actors. Actors will be discussed in more detail below; for now it is sufficient to know that they occupy rôles and insert preferences into the hierarchy of rôles at the Instantiation Layer. Security components combine together to form the operational counterparts of

security objects.

The instantiation layer needs building in the same way that the framework layer did. Again an iterative process is adopted with certain termination criteria. The key difference here is that this layer is built bottom up. SCs lay out the processes and resources needed to perform the boolean checks specified in corresponding SOs. SCs start at the final 'compliance' layer of the SO. Once the processes and resources required at this level are put in place we check to see if they are sufficient to complete the SO. If yes, then the SC terminates. If not then we move up to the rôle above and add additional processes and resources as needed. Again, this process repeats until all SCs are closed. At this point, the Instantiation layer is complete.

A little more formally, corresponding to the slightly more formal view of SOs sketched above, we can describe how SCs are combined to instantiate SOs as follows:

- SCs are combined according to the and/or forest determined by the SO that they instantiate;
- Each SC implements a checking process that applies to Actors at the level below;
- SCs return boolean values that instantiate internal nodes of the corresponding SO.

Working through our example again we start at the passenger level and work upward until we have sufficient processes and resources in place to return a boolean for the statement 'the passenger's possessions and luggage are permitted. The finished security component in this case would be as follows (Table 4):

In Figure 2, the SCs correspond to the green/yellow nodes in the Instantiation Layer

Note that whereas the SO terminated in a 'compliance' level the SC terminates at a 'provision' level when it reaches a rôle that can sufficiently provide the resources required to execute the SC without recourse to a higher rôle.

The final component of the instantiation layer (and the model) are Actors. Actors exist independently from any single security hierarchy. They represent entities that transition between hierarchies, this being the key difference between Actors and SOs. They can interact with any and all hierarchies present, simultaneously if necessary. Actors exist solely as a collection of tags corresponding to their attributes. When an Actor interacts with a hierarchy and seeks to populate one of its rôles, the hierarchy examines the Actor's tags (some or all of which may be unreadable to the hierarchy). It assigns the Actor to a rôle based on its tags and clones a copy of that rôle for the Actor to inhabit for the duration of its lifecycle in that system. This clone inherits the relevant dependencies and priorities of the rôle. Its preferences are obtained by interrogating the Actor's tag-cloud. Some Actors will have no preferences. Such Actors represent items such as inanimate objects (e.g., a passenger's bags in the airport example) or data stores (e.g., a baggage handling label in the airport example) that can be passed between hierarchies (the departure and destination airports will have separate security systems and this will be considered separate hierarchies) but have no intentions of their own. Once a clone is created, the Actor no longer directly interacts with the hierarchy for the duration of its lifecycle.

The Actor clone is now a full part of the hierarchy and can interact with its associated Security Components, updating its dependencies, priorities, and preferences as necessary. Its actions will determine the paths and states of the hierarchy it passes through. These in turn decided the exit point of the actor clone from the hierarchy. Each possible access point

Table 4: SCs

Rôle	Dependencies	Security Components
Airport Security Manager	Ensure no prohibited materials transit the airport	Provide resources (X-ray machine, metal detector, wands) Provide data on prohibited materials for X-ray comparison
Airport Security Staff	Examine all passengers and luggage passing through security checkpoint	Monitor X-ray machine and inspect results for prohibited items Hand-search suspect luggage Hand-scan suspicious passengers
Passenger	Comply with SO	Place luggage on scanner Walk through detector

has a set of tags associated with it that are then passed back to the core Actor to replace the set used to interact with the hierarchy. These may be identical or contain new or changed statuses. For example, a hierarchy designed to check the citizenship of an unknown actor may return tags, which it did not previously possess, identifying it as a national or an illegal alien.

The introduction of Actors adds a third dimension to the framework. Multiple actors can exist on any one layer of the hierarchy (and, typically, the lower levels will have more actors assigned to them). This also means that Security Objects can span multiple actors as well as multiple levels.

8. Real options and pricing risk

Risk based approaches rely on the appropriate pricing of potential risk vectors. Asset pricing models are becoming a more prevalent outside the finance industry for use in all manner of applications. For instance in[17, 42] an extension of an existing quantified loss function with diminishing marginal returns to security investment is redefined into a wait and see approach based on the valuation of a real option to secure assets.

Asset pricing models rely on comparability of risks to well understood benchmarks (usually a broad indices of financial assets). However, for security valuation no broadly traded financial assets are available.

A basic question is: why is it important to have a traded asset whose payoff is linked to a level of security? The basic answer to this question is that the action of trading claims on future levels of security processes all of the currently available information on potential security risks.

For instance in information security, a broad index of physical or information security incidents could be used as a benchmark. Futures contracts could be then traded for delivery at a fixed dollar amount per unit of the index at a pre-specified date. The trading in this contract determines the future expected levels of security risk.

Companies can then benchmark their own security risk relative to this index. This allows for standard risk management models such as the capital asset pricing model to compute the optimal hedging contract (or synthetic hedging contract) to indemnify their security position, this process is called hedging and is similar to insurance. In fact the usefulness of this approach is to allow firms to benchmark the precise value of insurance needed to indemnify their assets and how much this insurance should cost.

One issue with regards to modelling risk with respect to information security is that security shocks are not continuous in nature, but are inherently discrete jumps. Measuring a firms information security assets in terms of time t levels of confidentiality, $C(t)$, availability $A(t)$ and integrity $I(t)$ we can fit a modification of a standard real options model to measure the level of investment $K(t)$ needed to restore levels of risk to some form of optimum determined by a policy maker. In [13] the authors present a model of security risk management for information security using a discontinuous jump model.

For a discrete event risk vector a Poisson process is reasonable way of modelling discontinuous jump arrivals and the log-normal is a single-tailed distribution which captures a random variable that arises as a product positive independent random increments. Our choices represent a simplification of reality, but we believe it is a reasonable one.

Each security incident has a an impact on measurements of confidentiality or availability described by Equation 2 and are encapsulated in the vector y_t. Here $\psi_{1,t}$ and $\psi_{2,t}$ represent the primitive stochastic processes driving the risk generating process. For the most part we will assume that these are Poisson point processes with log-normal jump realizations.

$$\begin{bmatrix} y_{1,t+\Delta t} - y_{1,t} \\ y_{2,t+\Delta t} - y_{2,t} \end{bmatrix} = \begin{bmatrix} \pi_{1,1} & \pi_{1,2} \\ \pi_{2,1} & \pi_{2,2} \end{bmatrix} \begin{bmatrix} \psi_{1,t} \\ \psi_{2,t} \end{bmatrix} \phi_t \tag{2}$$

The parameter vector associated with the system consists of μ_1, μ_2, σ_1^2, σ_2^2, and the correlation coefficient ρ_{12}. The matrix Π, with components, $\pi_{i,j}$, linearly decomposes the signal of the security event arrival to consequences for the availability and confidentiality of the system. The time t states of confidentiality C_t and availability A_t — the discrete-time equivalents of the continuous-time measures $C(t)$ and $A(t)$ — are based on a fully secure system, $[C_0, A_0]^{\mathsf{T}}$. In the presence of security action, confidentiality and availability evolve according to Equation 3:

$$\begin{bmatrix} C_t \\ A_t \end{bmatrix} = \begin{bmatrix} C_0 \\ A_0 \end{bmatrix} - f \begin{bmatrix} y_{1,t} \\ y_{2,t} \end{bmatrix} \tag{3}$$

The function f is a rescaling function to ensure that the security event information, contained in the evaluation of the intensities (ψ_1, ψ_2) at t, matches the appropriate scale of confidentiality and availability.

Consider a policy-maker with two instruments, a long instrument, $x_1(t\,|t_0, E(y_t))$, which is a regular security implementation cycle taken at evenly spaced points in the time interval $[t_0, T]$, and set prior to t_0, and a short instrument, $x_2\,(t)$, a decision to take costly immediate early security action taken within the interval $t \in [t_0, T]$.

At time t, the non-decreasing sequence of confidentiality and availability is as follows (notation: $|_-$ denotes dependency and $\|$ is read as 'or'):

$$t'\,|E(y_t) \quad \in \quad \left[t_0, t_0 + \tfrac{T}{x_1}, t_0 + \tfrac{2T}{x_1}, \ldots, T\right] \tag{4}$$

$$C_{t+\Delta t} = \begin{cases} C_t + \Delta C_t\,|y_t & \text{iff} \quad (t \neq t')\,\|(x_2 = 0) \\ \bar{C} & \text{if} \quad t = t' \\ \bar{C} & \text{if} \quad x_2 > 0 \end{cases} \tag{5}$$

$$A_{t+\Delta t} = \begin{cases} A_t + \Delta A_t\,|y_t & \text{iff} \quad (t \neq t')\,\|(x_2 = 0) \\ \bar{A} & \text{if} \quad t = t' \\ \bar{A} & \text{if} \quad x_2 > 0 \end{cases} \tag{6}$$

In the first cases of Equations 5 and 6, the system is vulnerable because security events have occurred. Each security event maybe mitigated by the utilization of the long nor the short instruments. All other cases denote that the system has been patched.

The long instrument, x_1, is a non-negative integer defining the number of regular patch implementations during the planning period $[t_0, T]$. This process is considered to be the *regular patching cycle* and the associated required increases in information security capital stock are given as

$$\mathcal{P}_1(x_1(t)) = \nu \exp x_1 \tag{7}$$

where ν is the cost of implementing each patch.[5]

Implementation of the short instrument, $x_2(t)$, has two expenditure components: a fixed component and an additional variable reflecting the extra expenditure requirements for patching either side of the regular cycle. If $x_2(t) = 0$, then no additional expenditure is required; that is, $\mathcal{P}_2(x_2(t)) = 0$; otherwise, a convenient representation is given by the following equation:

$$\mathcal{P}_2(x_2(t)) = \upsilon + \alpha' (t'' - x_2)^2 + \beta' (x_2 - t')^2 \tag{8}$$

where t'' is the time of the next regular patch, t' is the timing of the previous regular patch, and α' and β' are structural parameters. In the case $\alpha' = \beta' = 0$, there is no additional penalty for timing the patch outside of the regular cycle. Patching under the short instrument is considered to be patching outside the regular cycle, constituting the *irregular cycle*: practitioners often refer to it as 'out-of-cycle' patching.

$\mathcal{P}_1$ and $\mathcal{P}_2$ are the components of $\mathcal{P}$ that enter the system equation and leading to deviations from the target level of investment $\bar{K}$. Balancing the future evolution of C and A via simulation with investment Kallows the policy maker to determine the optimal allocation of investment.

9. Models of attack and defense with risk averse targets

The preceding model assumes that the contingent claim (investment) can be valued in a way that the implicit option x_1 versus x_2 can be valued in a risk neutral or risk averse framework. A key aspect of this work is in delineating the equilibrium risk structure for these types of valuation models.

In this subsection we will introduce some simplifying assumptions on the systems architecture and then produce predictions on strategic behaviour that generate the resultant security risks, based on the concurrent measures of the security performance of the assets.

The previous approach is extremely helpful in pricing risks. However, if is fundamentally free of any structure in terms of choices of agents in the economic system and the mechanism that generates risks.

9.1 A Policy Model with Insurance

This example provides the basic components for any risk model involving antagonists in a security game. The model is general i.e. it can be calibrated to any number of targets and attackers acting strategically and the equilibrium conditions are easily expressed and analyzed either analytically or via simulation approaches.

[5] For $x_1 = 0$, we define $\mathcal{P}_1(x_1) = 0$.

The Agents

We determine a single social planner/policy maker (we interchange between these terms in this context, for public policy contexts differentiation is non-trivial) who decides constraints on Θ. The 'information security market', is an attack defense game with the following representative agents:

- Attackers — Agents wishing to cause damage to targets.
- Targets (the potentially insured) — Firms in an economic system (the market) assume to be risk averse.
- Insurers - provide loss adjustments to targets in the event of successful attacks.

Different numbers of attackers may attack particular targets. Let n_{Ai} denote the number of attackers which attack target i. We consider the case where attackers can observe the overall level of vulnerability for the population of targets but not the degree of vulnerability of any particular target. In this case, there is no reason for an attacker to attack one target instead of another. Or suppose, for some other reason, that each attacker directs his or her attack at a randomly chosen target. We further approximate a random assignment of attackers to targets by introducing the simplifying assumption that the N_A attackers are spread uniformly over the N_T targets. In this case, $n_{Ai} = n_A = N_A/N_T$ for all i. In this case, $n_{Ai} = N_A/N_T$.

Let $\sigma_i = \sigma_i(x_i, n_{Ai})$ denote the probability that one or more attacks mounted against target i are successful. The probability σ_i depends on both the level of defensive expenditure by target i and the number of attacks mounted against target i. We make the following assumptions on $\sigma_i(x, n)$. First, we posit that $\partial\sigma_i/\partial n > 0$, so that an increase in the number of attackers attacking a target increases the probability that at least one attack is successful for all levels of defensive expenditure x_i and any number of attackers n_{Ai}. Second, we posit that $\partial\sigma_i/\partial x < 0$ so that an increase in the defensive expenditure of a target reduces the probability that at least one attack is successful. Finally, we assume that $\partial^2\sigma_i/\partial x^2 > 0$, at least for large enough values of x. This last assumption implies that the marginal returns to defensive expenditure are decreasing, at least for large enough values of x.

9.2 Attacking Targets and Levels of Defensive Expenditure

The incentives to mount an attack are determined by cost-benefit considerations. Let $R_i(n_{Ai})$ denote the expected monetary reward per attacker obtained by each one of the n_{Ai} attackers who attack target i when one or more of these attacks turns out to be successful.

We assume that $\partial R_i(n)/\partial n \leq 0$ to allow for the possibility that competition among attackers may lower the expected reward per attacker. In order to highlight the effects of competition among attackers, we consider, in most of this paper, a version of the model where the "first winner takes all."In this case, the first attacker who mounts a successful attack against target i receives the reward R_i, where R_i is a positive constant. All other attackers mounting attacks against target i receive nothing.

If each attacker has an equal chance of being the one to make the first successful attack, then the probability that a given attacker is the one to obtain the reward from "success" is

simply $1/n_{Ai}$ and $R_i(n_{Ai}) = R_i/n_{Ai}$. The competition among attackers is particularly sharp in this framework.

We suppose that attackers are risk neutral and wish to maximize their expected net reward from an attack. The expected net reward which an attacker obtains from attacking target i is given by the following expression.

$$\sigma_i(x_i, n_{Ai})\, R_i(n_{Ai}) \;-\; C_A. \tag{9}$$

We normalize the expected net reward which a potential attacker obtains by not mounting attacks to 0.

In equilibrium, therefore, the number of attackers per target, n_A^*, is given by the following equation:

$$\frac{1}{N_T} \sum_{i=1}^{N_T} R_i(n_A^*)\, \sigma_i(x_i, n_A^*) \;=\; C_A. \tag{10}$$

The left-hand side of equation (2.5) denotes the expected reward to an attacker from mounting attacks against the population of targets.

Let V_{0i} denote the value of the assets at risk in a cyber-attack against target i. If one or more successful attacks occur, target i is assumed to incur the monetary loss $L_i < V_{0i}$. For simplicity, we assume that L_i does not depend on the number of successful attacks. This could be the case, for example, if a target is assumed to fix the vulnerability which exposed it to attack once a successful attack occurs.

We suppose that target i is risk averse with attitudes toward risk that can be described by a von Neumann-Morgenstern utility function, $U_i(v)$, where $U_i(v)$ is a twice-differentiable, strictly increasing, weakly concave function.

Target i's preference for different levels of risk and defensive expenditure is described by the expected utility, $EU_i(x_i, n_{Ai})$, given in the following equation:

$$\begin{aligned} EU_i(x_i, n_{Ai}) \;&=\; (1 \,-\, \sigma_i(x_i, n_{Ai}))\; U(V_{0i} - x_i) \\ &\quad + \sigma_i(x_i, n_{Ai})\, U(V_{0i} - x_i - L_i) \end{aligned} \tag{11}$$

where higher levels of expected utility correspond to more preferred outcomes. Target i's expected utility is a function of the target's level of defensive expenditure and the number of attackers that attack target i.

Of course, the expected utility also depends on various other parameters such as V_{0i} and L_i.

As discussed previously, the quantity $\sigma_i(x_i, n_{Ai})$, indicates the probability that one or more successful attacks are mounted against target i. The quantity $V_{0i} - x_i$ represents the value of target i's assets net of defensive expenditure when no successful attack occurs.

Similarly, $V_{0i} - x_i - L_i$ is the net value of target i's assets after a successful attack. In the special case where $U_i(v) = v$ and target i is risk neutral, target i's expected utility is simply the expected net monetary value of its assets. This asserts that a risk neutral target will wish to choose its level of defensive expenditure to minimize the expected loss: $x_i + \sigma_i(x_i, n_{Ai})L_i$.

Suppose that target i chooses the level of defensive expenditure x_i to maximize $EU_i(x_i, n_{Ai})$ holding the number of attackers n_{Ai} fixed. It is convenient to collect the terms involving σ_i on the right-hand side of equation (3.1) and rewrite that equation as follows:

$$EU_i(x_i, n_{Ai}) \;=\; U_i(V_{0i} - x_i) \;-\; \sigma_i(x_i, n_{Ai})\, \Delta U_i \tag{12}$$

where ΔU_i is given by

$$\Delta U_i \;=\; U_i(V_{0i} - x_i) \;-\; U_i(V_{0i} - x_i - L_i). \tag{13}$$

Which indicates that ΔU_i can be interpreted as the loss to target i when a successful attack occurs measured in terms of utility rather than money.

Suppose that the value of x_i which maximizes $EU_i(x_i, n_{Ai})$ is given by the usual first-order condition: $\partial EU_i/\partial x_i = 0$. The first-order condition can be written as follows.

$$\partial[U_i(V_{0i} - x_i)]/\partial x_i + \sigma_i(x_i, n_{Ai})\,\partial[\Delta U_i]/\partial x_i \;=\; -\,\partial[\sigma_i(x_i, n_{Ai})]/\partial x_i \Delta U_i. \tag{14}$$

The left-hand sideis the cost in terms of utility of a marginal increase in defensive expenditure. The right-hand side is the marginal utility gain from a small increase in defensive expenditure due to the reduction in the expected loss from an attack.

This set-up restates the usual rule that defensive expenditure should be increased until the marginal cost of an additional unit is equal to the marginal gain.

We follow common practice in the economic literature and model this outcome as the Nash equilibrium of a noncooperative game. In a Nash equilibrium each player's choice of strategy must be optimal given the player's beliefs about the strategies of the other players.

In addition, the beliefs of each player must be consistent with the actual strategies used by the other players. When all targets and attackers are identical, it is plausible to restrict attention to symmetric equilibria.

In a symmetric Nash equilibrium $\left(x^{NI}, n_A^{NI}\right)$ each target i selects the same level of defensive expenditure x^{NI} so that x^{NI} solves

$$\max_{x_i} EU_i(x_i, n_A^{NI})$$

and the number of attackers per target n_A^{NI} is determined by the free entry condition

$$R(n_A)\,\sigma(x^{NI}, n_A) \;=\; C_A.$$

Note that in equilibrium both attackers and targets correctly forecast the choices of other players.

We suppose that the policy maker's preferences regarding the risks of cyber-attacks are described by a von Neumann-Morgenstern utility function:

$$W \;=\; \sum_{i=1}^{N_T} U_i. \tag{15}$$

This utility function is commonly referred to as a utilitarian social welfare function since it consists of the sum of the utilities of the individual targets.

Order of the policy equilibrium:

- Policy maker observes unrestricted action of attackers and targets.
- Next we that the policy maker moves and all the other actors (targets and attackers) make their choices in a second stage. All the choices in stage two are made simultaneously after observing the choices of the policy maker.

9.2.1 Self-Protection with Actuarially Fair Insurance

Consider the following policy experiment: A target can choose the level of defensive expenditure as well as whether to purchase insurance.

An insurance contract for target i is typically described by the level of coverage, Q_i, which specifies the amount paid to target i in the event of a loss, and the premium, $\Pi_i(Q_i)$, which specifies the amount that target i must pay for the level of coverage Q_i. The premium is paid whether or not a loss occurs.

If target i purchases a general insurance contract specified by the variables Q_i and $\Pi_i(Q_i)$ and chooses a level of defensive expenditure x_i then target i's expected utility is given by the following equation when the number of attackers for target i is n_{Ai}.

$$\begin{aligned} EU_i(x_i, Q_i, \Pi_i(Q_i), n_{Ai}) = \\ (1 - \sigma_i(x_i, n_{Ai}))\, U(V_{0i} - x_i - \Pi_i(Q_i)) + \\ \sigma_i(x_i, n_{Ai})\, U(V_{0i} - x_i - \Pi_i(Q_i) - L_i + Q_i)\ . \end{aligned} \tag{16}$$

As discussed previously, the quantity $\sigma_i(x_i, n_{Ai})$ represents the probability that target i incurs a loss from an attack. The quantity σ_i therefore also represents the probability that an insurer who insures target i for a level of coverage Q_i will pay out that amount.

Hence, the net expected payment that the insurer obtains from target i is: $\Pi_i(Q_i) - \sigma_i(x_i, n_{Ai})Q_i$. The price $\pi_i = \sigma_i(x_i, n_{Ai})$ is the lowest price per unit of coverage that can be charged to target i for which the premium will cover the expected payout in the event of a loss. In the economic literature on insurance, the provision of insurance coverage at this price is commonly referred to as actuarially fair insurance.

The provision of actuarially fair insurance requires that the price per unit of coverage, π_i, and, hence, the premium, Π_i, paid by target i depends on target i's level of defensive expenditure, x_i. For this to be the case, the insurer must be able to observe the value of x_i.

In this paper, we assume that the insurer can costlessly observe the level of x_i. It is sufficient that the level of x_i can be verified after a successful attack has taken place.

In this case, an insurer can set the premium based on target i's statements about x_i and refuse to pay if target i's stated level of defensive expenditure differs from the actual level of expenditure.

In the case of actuarially fair insurance, equation (4.2) reduces to

$$\begin{aligned} EU_i(x_i, Q_i, n_{Ai}) = \\ (1 - \sigma_i(x_i, n_{Ai}))\ U(V_{0i} - x_i - \sigma_i Q_i) \\ + \ \sigma_i(x_i, n_{Ai})\ U(V_{0i} - x_i - \sigma_i Q_i - L_i + Q_i)\ . \end{aligned} \tag{17}$$

Here for simplicity we suppress the dependence of the expected utility on the premium $\sigma_i(x_i, n_{Ai})\, Q_i$, and write the expected utility for this special case as $EU_i(x_i, Q_i, n_{Ai})$.

Suppose that, holding n_{Ai} fixed, target i wishes to choose x_i and Q_i to maximize the expected utility $EU_i(x_i, Q_i, n_{Ai})$. When target i is risk neutral, so that $U_i(v) = v$, it is straightforward to verify that the right-hand side of equation (3.7) reduces to the quantity: $V_{0i} - x_i - \sigma_i(x_i, n_{Ai})L_i$ for all levels of Q_i.

A risk neutral target is indifferent between buying insurance coverage or not even when this coverage is provided at an actuarially fair price. Hence, $Q_i = 0$ is optimal for a risk

neutral target and, as in the case of no insurance discussed previously, the target's choice of defensive expenditure mimimizes the expected monetary loss: $x_i + \sigma_i(x_i, n_{Ai})L_i$.

When $U_i(v)$ is strictly concave, so that target i is strictly risk averse, it is convenient to solve for target i's optimal choice in two steps.

In step 1, we calculate the optimal level of coverage, $Q_i(x_i)$, for each level of defensive expenditure. In step 2, we calculate the optimal level of defensive expenditure x_i when Q_i is set to its optimal level, that is, when $Q_i(x_i)$ is substituted for Q_i.

Proposition 1

A strictly risk averse target i which is offered insurance at an actuarially fair rate will always find it optimal to choose a level of coverage equal to the full loss, that is, $Q_i(x_i) = L_i$ for all values of x_i and n_{Ai}. Moreover, this choice is the unique optimum when $\sigma_i(x_i, n_{Ai}) > 0$.

Proof and Discussion of Proposition 1

Proposition 1 follows directly from Jensen's inequality. The case where $Q_i(x_i) = L_i$ is commonly referred to as the case of full insurance, since target i is fully reimbursed for a loss.

Once target i has chosen to be fully insured, the result implies that target i will choose x_i to maximize the utility $U_i(V_{0i} - x_i - \sigma_i(x_i, n_{ai})L_i)$.

Since $U(v)$is an increasing function, this corresponds to choosing x_i to maximize the expected net value of target i's assets, $V_{0i} - x_i - \sigma_i(x_i, n_{ai})L_i$ or, equivalently, to minimize the expected monetary loss, $x_i + \sigma_i(x_i, n_{ai})\, L_i$■.

Not surprisingly, a strictly risk averse target who is able to offload the entire risk of a loss by the purchase of actuarially fair insurance chooses the same level of defensive expenditure as would be chosen by a risk neutral target.

Proposition 2

Risk averse targets who can buy actuarially fair insurance choose the same level of defensive expenditure as would be chosen by a risk neutral target. That is, each x_i will minimize the expected monetary loss $x_i + \sigma_i(x_i, n_{ai})\, L_i$.

Proof and Discussion of Proposition 2

To characterize equilibrium behavior, we also need to consider attackers' behavior. Importantly, however, the expected reward for attackers does not depend on the targets' levels of insurance coverage but only on the targets' levels of defensive expenditure.

Hence, strategies which are best replies for potential attackers are described in exactly the same way whether or not insurance is available. In particular, the equilibrium relation between the number of attackers per target and the levels of defensive expenditure, $n_A^*(x_1, \ldots, x_{N_T})$, is the same in these two cases.

Thus the equilibrium levels of defensive expenditure x_i^{FI} and the equilibrium number of attackers per target, n_A^{FI}, satisfy the following equations:

$$n_A^{FI} = n_A^*(x_1^{FI}, \ldots, x_{N_T}^{FI}) \text{ and } x_i^{FI} = x_i^{**}(n_A^{FI}) \text{ for all } i\blacksquare \tag{18}$$

9.2.2 What does the equilibrium tell us?

When the social policy maker choices a set of constraints, in the presence of insurance she observes a population of risk neutral targets (on average) under investing in protection. The insurance company profit maximization condition is *incentive incompatible* with the outcomes of the policy maker given the suggested policy objective function.

Monopoly insurer seeks to maintain or even increase risk to maximize its profit condition. Target cost benefit analysis results in average underinvestment in security (with over purchasing of insurance).

Policy maker is still needed to set minimum constraints. This assumes that the insurance company can observe the level of defensive effort, which is generally viewed as the best condition for a perfect market in insurance. In this case strategic interactions with attackers renders the insurance company able to control risks to maximize payoffs. Hence leading to a perverse outcome under these conditions.

9.3 Policy Maker Utility Theory and Loss Function for Vulnerability Management

Our first goal is to orientate the vulnerability management problem in an expected utility-maximization framework. We seek to construct a objective function, whose solution at the maximum is equivalent to the expected utility maximization condition. We state the policy-maker's objective function as

$$\mathbb{E}\left(\mathfrak{U}\left(t,T\right)\right) \triangleq \max_{K(t)} \int_{t}^{T} e^{-\beta t} u\left(x(t);K(t)\right) d\mathbb{P}\left(\omega\left(t\right)\right) \tag{19}$$

where

- T is the terminal time,
- $K(t)$ is a choice of investment function,
- $x(t) = \{x_1, \ldots, x_n\}$ is a n-vector of real-valued system attributes that is stochastic, because of threats, defined over the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ [43],
- $u(x(t); K(t))$ is an instantaneous real-valued twice-differentiable utility function over the system attributes x, with exogenous parameters the investment function, $K(t)$,
- β is a global discount rate, and
- $\omega(t) \in \Omega$ is an experiment in the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ [43].

Here the idea is that we vary the investment function $K(t)$ in order to maximize expected utility at time t by choosing a future investment time $t^* \geq t$.

Equation 19 provides a general characterization of a variety investment problems. As such, it is difficult to derive general analytic solutions and so we reduce the problem space to a polynomial approximation of Equation 19 for which solutions can be found.

In this section, we assume a risk-averse policy-maker. In the case of a risk-neutral policy-maker, our analysis collapses to a polynomial approximation to the real options solution for the investment timing problem [44].

9.4 The Power Utility Family

We explore the general problem described above in the case in which $n = 2$. This is the simplest case that fully illustrates our approach. Examples of this case would include the security attributes confidentiality and availability, and we have explored, in less generality, the way in which these attributes trade off against each other elsewhere [12, 13]. In [12], for example, we exogenously imposed an investment cycle on the representative firm within the model. In contrast, in this section, we demonstrate how an investment cycle arises from investment rigidities.

In economics and finance, the power utility family of functions is the predominant mechanism for defining preferences for inter-temporal decision problems. Whilst for most of our derivation we are agnostic to choice of utility function (our interest is restricted to the ratio of the derivatives), some discussion of the higher level of functional form is relevant and useful for future applied work. The basic power utility construct for a consumption variable $x_i \in \{x_1, x_2\}$, suppressing the control variable K, has a partial utility function defined as

$$u_i(x_i) = \frac{x_i^{1-\gamma_i}}{1-\gamma_i} \tag{20}$$

where γ_i is the coefficient of relative risk aversion $\mathfrak{R}$, for the i^{th} attribute. Combining the partial utility functions with cross power utility would yield and over utility function of

$$u(x_1, x_2) = \frac{1}{1-\gamma_1} x_1^{1-\gamma_1} + \frac{1}{1-\gamma_2} x_2^{1-\gamma_2} + 2\frac{|x_1 x_2|^{1-\gamma_{12}}}{1-\gamma_{12}} \tag{21}$$

Several extensions of the power utility have been proposed in the literature and several of these innovations have useful interpretations for information security problems. From this point onward, for ease of exposition, we shall concentrate on the partial utility functions. Kahneman and Tversky [45] suggest the inclusion of a fixed point, or kink point, k, to discriminate between aversion to risk of loss and aversion to risk of gain. The power utility representation of this approach is

$$u_i(x_i) = \begin{cases} \frac{1}{1-\gamma_i} x_i^{1-\gamma_i} & \forall x_i > k \\ \frac{1}{1-\tilde{\gamma}_i} x_i^{1-\tilde{\gamma}_i} & \forall x_i \leq k \end{cases} \tag{22}$$

where $\tilde{\gamma}_i \neq \gamma_i$. The inclusion of the fixed point adds a significant complication to the type of optimization suggested herein as the derivatives of $u(x)$ are now discontinuous.

An alternative augmentation is to include a utility profile of the consumption of system attributes at some future point in time. This nesting of future utility allows for a substitution

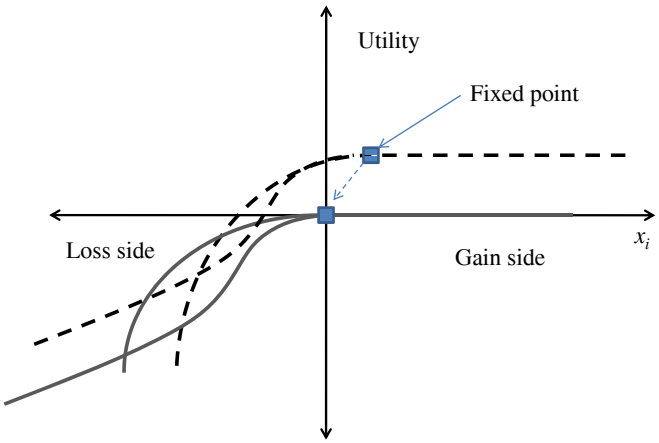


Figure 3: Illustration of the projection of a family of utility functions, $u(x_i)$, for a single attribute. The markers represent the fixed points in the utility problem. The fixed points can be located anywhere within this plane. For example, the dashed line represents a curve with a fixed point at at positive value of x_i. For our purposes, we assume the fixed point is at the origin (the dark grey line) and that deviations from steady state are always to the left of the origin.

between current expected utility and future expected utility and has been used extensively since first being suggested in [46]. The power utility form is compactly presented as

$$u_i\left(x_i(t)\right) = \left(1-\zeta_i\right) x_i^{\frac{1-\gamma_i}{\theta_i}} + \zeta_i \mathbb{E}_t\left(u\left(x_i(t+\Delta t)\right)^{\frac{1}{\theta_i}}\right)^{\frac{\theta_i}{1-\gamma_i}} \tag{23}$$

where θ_i is the anticipated future coefficient of relative risk aversion at $t+\Delta t$, ζ_i is the inter-temporal elasticity of substitution — that is, the substitution between current and future expected utility.

The last type utility function we have considered in our applied work is the 'inside and outside of habit' utility function, suggested by [47]. This sets expected utility as being relative to a peer group represented by an index (of consumption), ξ_i, of the variable x_i. In our notational scheme, the power utility version of this type of utility function is (as usual, suppressing $K(t)$) defined as

$$u_i\left(x_i\right) = \frac{\left(x_i \xi_i^{-1}\right)^{1-\gamma}}{1-\gamma} \tag{24}$$

There are obvious circumstances where each of these definitions of preferences will be appropriate. Augmentations to cater for non-zero cross products (i.e., supermodularity or submodularity) are also relatively trivial. For instance, fixed points are common in many aspects of information security: in particular, on the loss side — essentially, improvements over targets are relatively under rewarded. Figure 3 outlines an example of this structure.

9.5 The Policy-maker's Problem

We begin with a quick review of several key results for properties of utility functions and in particular the structure of risk aversion. Absolute risk aversion ($\mathfrak{A}$) in two-variable (i.e., $n = 2$) multi-attribute decision problems is defined as follows:

$$\mathfrak{A}(x_1, \cdot) = \frac{u''_{x_1}(x_1, \cdot)}{u'_{x_1}(x_1, \cdot)} \qquad \mathfrak{A}(\cdot, x_2) = \frac{u''_{x_2}(\cdot, x_2)}{u'_{x_2}(\cdot, x_2)} \tag{25}$$

where we we suppress in the notation the exogenous parameter, $K(t)$. This is then simply mapped to a relative risk aversion ($\mathfrak{R}$) context

$$\mathfrak{R}(x_1, \cdot) = \frac{-\gamma_{x_1} u''_{x_1}(x_1, \cdot)}{u'_{x_1}(x_1, \cdot)} \qquad \mathfrak{R}(\cdot, x_2) = \frac{-\gamma_{x_2} u''_{x_2}(\cdot, x_2)}{u'_{x_2}(\cdot, x_2)} \tag{26}$$

where γ_{x_1} and γ_{x_2} are the coefficients of relative risk-aversion (i.e., the marginal rate of change in risk-aversion with respect to $u(\cdot)$) for each of the system attributes.

Both $\mathfrak{A}$ and $\mathfrak{R}$ are useful tools in summarizing the properties of specific utility functions: in addition to the risk aversion properties, the cross products for the attributes are useful in elucidating the preference structure. In the bivariate context, there are three main combinations. Consider the following decomposition

$$u(x_1, x_2) = u_1(x_1) + u_2(x_2) + u_{12}(x_1, x_2) \tag{27}$$

where $u(x_1)$ and $u(x_2)$ are the partial utility functions with respect to system attributes x_1 and x_2 and $u_{12}(x_1, x_2)$ is the joint utility adjustment. In the general form of our modelling framework we maintain the general assumption that $u_{12}(x_1, x_2) \neq 0, \forall \{x_1, x_2\} \in \mathbb{R}^2$. For our final analytic solutions, however, we have assumed *separable additivity*; that is,

$$u''_{x_1,x_2}(x_1, \cdot) = 0 \quad \forall x_2 \qquad u''_{x_1,x_2}(\cdot, x_2) = 0 \quad \forall x_1 \tag{28}$$

The contrasting assumptions that maybe made on the shape of the multi-attribute utility function are *supermodularity* whereby

$$u''_{x_1,x_2}(x_1, \cdot) > 0 \quad \forall x_2 \qquad u''_{x_1,x_2}(\cdot, x_2) > 0 \quad \forall x_1 \tag{29}$$

and *submodularity* whereby

$$u''_{x_1,x_2}(x_1, \cdot) < 0 \quad \forall x_2 \qquad u''_{x_1,x_2}(\cdot, x_2) < 0 \quad \forall x_1 \tag{30}$$

Discussion of the appropriate application of these properties is usually driven by game-theoretic models of incentives. For instance, most problems can be treated as separably additive, and as such the attributes rolled in a single linear function. However, in the authors' experience of working with industry and government, compound attacks on multiple system attributes are often more damaging than attacks (of similar component-wise magnitude) that occur at different times. In this case, utility functions incorporating a degree of supermodularity would be most appropriate for describing policy-maker preferences. Cases of submodular preferences are much rarer, although not unheard of. For instance, in a confidentiality, integrity, and availability (CIA) framework, a distributed denial of service (DDOS) attack mixed with a breach of confidentiality could, for certain institutions such as retailers, be understood as being submodular: to some extent, the DDOS mitigates the effectiveness of the confidentiality attack as the system's availability (to the confidentiality attacker) is compromised.

9.6 Decision Support

For simplicity of exposition, we now simplify the decision under uncertainty problem to a policy maker choosing a forward looking investment profile from an initial time t_0; that is, at a point where no existing vulnerabilities are present. The resulting expected timing of investment $t^* > t_0$ is the ex-ante expected amplitude of the investment cycle. Future work will address the 'steady-state' equilibrium investment horizon at time $t > t_0$.

For a given choice of utility function $u : \mathbb{R}^n \to \mathbb{R}$ operating over $n = 2$ system attributes — consumption variables in an economic context — the dynamic representation of the utility function is defined from the terms of the Taylor expansion as

$$\begin{aligned} u\left(x_1(t_0)+\Delta x_1, x_2(t_0)+\Delta x_2\right) &= u\left(x_1(t_0), x_2(t_0)\right) + \\ &\left(u_{x_1}\left(x_1(t_0), x_2(t_0)\right)\Delta x_1 + u_{x_2}\left(x_1(t_0), x_2(t_0)\right)\Delta x_2\right) + \\ &\tfrac{1}{2}u_{x_1,x_1}\left(x_1(t_0), x_2(t_0)\right)\left(\Delta x_1\right)^2 + \\ &\Delta x_1 \Delta x_2\left(u_{x_1,x_2}\left(x_1(t_0), x_2(t_0)\right)\right) + \\ &u_{x_2,x_2}\left(x_1(t_0), x_2(t_0)\right)\left(\Delta x_2\right)^2 \end{aligned} \tag{31}$$

where $x_1(t_0)$ and $x_2(t_0)$ denote initial values, which is a valid approximation as Loistl [48] demonstrates that under fairly mild conditions the remainder converges to zero.

Assuming that the moment generating process is fully described by its first two moments, the following notation applies:

$$\mu_{x_1}(t) = \mathbb{E}_t\left(x_1(t) - \bar{x_1}\right) \tag{32}$$
$$\mu_{x_2}(t) = \mathbb{E}_t\left(x_2(t) - \bar{x_2}\right) \tag{33}$$
$$\sigma_{x_1}(t) = \mathbb{E}_t\left(x_1(t) - \bar{x_1}\right)^2 \tag{34}$$
$$\sigma_{x_2}(t) = \mathbb{E}_t\left(x_2(t) - \bar{x_2}\right)^2 \tag{35}$$
$$\sigma_{x_1,x_2}(t) = \mathbb{E}_t\left(x_2(t) - \bar{x_2}\right)\left(x_1(t) - \bar{x_1}\right) \tag{36}$$

where $\bar{x_1}$ and $\bar{x_2}$ are long-run targets and $\mathbb{E}_t$ is the instantaneous expectation at time t. Substituting these into the utility function above results in the following expected utility function:

$$\begin{aligned} \mathbb{E}\left(u\left(x_1(t), x_2(t)\right)\right) &= u\left(x_1(t_0), x_2(t_0)\right) + \left(u_{x_1}\left(x_1(t_0), x_2(t_0)\right)\mu_{x_1} + u_{x_2}\left(x_1(t_0), x_2(t_0)\right)\mu_{x_2}\right) + \\ &\tfrac{1}{2}u_{x_1,x_1}\left(x_1(t_0), x_2(t_0)\right)\sigma_{x_1}(t) + \sigma_{x_1,x_2}(t)\left(u_{x_1,x_2}\left(x_1(t_0), x_2(t_0)\right)\right) + \\ &u_{x_2,x_2}\left(x_1(t_0), x_2(t_0)\right)\sigma_{x_2}(t) \end{aligned} \tag{37}$$

Assuming the existence of threats that degrade the system, induce utility losses, and continuously compound, and which are such that, for all t, $x_1(t) \geq 0$ and $x_2(t) \geq 0$, then the utility function will obey

$$u\left(\bar{x_1}, \cdot\right) \geq u\left(\bar{x_1} + x_1(t), \cdot\right) \qquad \forall t \tag{38}$$
$$u\left(\cdot, \bar{x_2}\right) \geq u\left(\cdot, \bar{x_2} + x_2(t)\right) \qquad \forall t \tag{39}$$

where $\cdot$ is a placeholder in the function. This results in decreasing marginal utility with

respect to loss:

$$\frac{\partial u\left(\bar{x_1},\cdot\right)}{\partial x_1} \geq \frac{\partial u\left(\bar{x_1}+x_1(t),\cdot\right)}{\partial x_1(t)} \quad \forall t \tag{40}$$

$$\frac{\partial u\left(\cdot,\bar{x_2}\right)}{\partial x_2} \geq \frac{\partial u\left(\cdot,\bar{x_2}+x_2(t)\right)}{\partial x_2} \quad \forall t \tag{41}$$

We define the following policy parameters, as described in Table 5:

$$w_{x_1} = -u_{x_1}\left(x_1(t_0),x_2(t_0)\right) \tag{42}$$

$$w_{x_2} = -u_{x_2}\left(x_1(t_0),x_2(t_0)\right) \tag{43}$$

$$v_{x_1} = -2u_{x_1,x_1}\left(x_1(t_0),x_2(t_0)\right) \tag{44}$$

$$v_{x_1} = -2u_{x_2,x_2}\left(x_1(t_0),x_2(t_0)\right) \tag{45}$$

$$v_{x_1,x_2} = -u_{x_1,x_2}\left(x_1(t_0),x_2(t_0)\right) \tag{46}$$

Each of these has a simple interpretation, as described in Table 5.

Table 5: Policy Parameters

Parameter	Description
w_{x_1}	Policy weighting applied to first system attribute
w_{x_2}	Policy weighting applied to second system attribute
v_{x_1}	Sensitivity (risk aversion) to variance in first system attribute
v_{x_2}	Sensitivity (risk aversion) to variance in second system attribute
v_{x_1,x_2}	Sensitivity to covariance first and second system attributes

From the asymmetric preference structure, the policy-maker's problem can be expressed as maximizing an expected utility function. The expected utility from the state of the system attributes is defined by the following integral that represents the cost of inaction:

$$\mathfrak{U}\left(t_0,T|w_{x_1},w_{x_2},v_{x_1},v_{x_2},v_{x_1,x_2}\right) = \int_{t_0}^{T} e^{-\beta t}\ell\left(t|w_{x_1},w_{x_2},v_{x_1},v_{x_2},v_{x_1,x_2}\right)dt \tag{47}$$

$$= \int_{t_0}^{T} e^{-\beta t}\left(w_{x_1}\mu_{x_1}(t)+w_{x_2}\mu_{x_2}(t)+v_{x_1}\sigma_{x_1}(t)+2v_{x_1,x_2}\sigma_{x_1,x_2}(t)+v_{x_2}\sigma_{x_2}(t)\right)dt$$

where

$$\ell(t|w_{x_1}, w_{x_2}, v_{x_1}, v_{x_2}, v_{x_1,x_2}) = w_{x_1}\mu_{x_1}(t) + w_{x_2}\mu_{x_2}(t) + v_{x_1}\sigma_{x_1}(t) + 2v_{x_1,x_2}\sigma_{x_1,x_2}(t) + v_{x_2}\sigma_{x_2}(t) \quad (48)$$

The additional separable component in the policy-maker's loss function is defined with respect to the additional investment required in the presence of disclosed vulnerabilities. The objective is to find the policy-maker's cycle time to investment; that is, the upper limit of integration, T, which satisfies the equality of utility of action and utility of inaction. We denote this value as t^*.

10. Preliminary Models for National Grid

In this section we delineate the level of abstraction of the system and policy components for various security problems derived from the case study deliverables, D1.3, D2.3 and D3.3. This is the provisional work and is the starting point for SECONOMICS Deliverable 6.2. Part of the work in Deliverable 6.2 is to identify the security problems that may be tractably modeled using our approach and then use these models to inform corporate and public policy using the ideas developed previously in the document.

Critical national infrastructure presents an eclectic set of security scenarios for the purposes of applied modelling. These summaries are short and are designed to link this overview of the various modeling approaches to the case study WPs. For more details see D1.3, D2.3 and D3.3. In Deliverable 2.3 a series of business objects are identified as being critical and within the scope of the project. These are the SCADA and control systems for electricity distribution, the interconnectors that join separate national distribution networks and the corporate network used in administering the business (separate from control of the CNI side).

10.1 SCADA and Control Systems

The Supervisory Control And Data Acquisition (SCADA) system links the network operators to the substations, generators and interconnectors that comprise the nodes of the electricity network. The network requires both load information from the demand side and electricity generation (supply) side to ensure the total amount of electricity in the distribution network is balanced.

Both undersupply and oversupply of electricity in system such as this presents a threat to safety of end users on the grid. This requires modelling of two networks, the electricity transmission network and the data network that runs alongside it.

The data network measures the load on the transmission network at specific intervals. Therefore the two networks interact at certain nodes in their topology. The SCADA system monitors the load at each point in the transmission network and ensures it lies within the operator required range.

Let x be a vector of nodes in the transmission network, y be a vector of nodes supplying information to the SCADA system and z be a vector of controls (for instance generation capacity and routing controls). The network $\mathcal{N}(x, y)$, measures the load and data of the

load for the SCADA system. In a structural form, the model should in equilibrium $N(x,z) - D(y,z) = 0$, where $N(\cdot)$ and $D(\cdot)$ are functions that translate load and data network nodes into equivalent measures.

An abstraction would be setting $N(x,z) - D(y,z) = [C,I,A]'$ where C, I and A are measurements of confidentiality, availability and integrity, respectively. In this context we can use a standard dynamics stabilization model where $L(C-\bar{C}, I-\bar{I}, A-\bar{A})$ is a loss function designed to stabilize the network loads by adjusting control nodes z. The parameters of the loss function are influenced by the target levels of confidentiality, availability and integrity ($\bar{C}$, $\bar{I}$, $\bar{A}$).

This could then be treated as a standard impulse response model, where the system equations $(N(x,z), D(y,z))$ evolve as series of differential equations. Following the approach suggested in [12]. Given that the structure of $D(\cdot)$ includes overlapping wired and wireless data communication, we can add a set of sub functions that delineate this extra complexity to the data channel system.

The structure of $N(\cdot)$ and $D(\cdot)$ is influenced by the specific technologies involved in the design of the system (see SECONOMICS deliverable D2.3 for qualitative description of the network technology and structure).

10.2 Interconnectors

Related to the SCADA system are load demands that can come from outside the geographical location of the transmission grid via the interconnector system. In deliverable D2.3 the use of interconnectors in Europe is discussed at length. Interconnectors bridge different electricity grids of individual countries. The use of this system is in managing volatility of demand and supply across larger geographical, population and industrial areas.

From D2.3: *Making this functionality possible allows countries to limit the amount of reserve capacity it must hold as well as help with any potential shocks in demand or supply such as increases in demand due to large events or unexpected malfunctions at power generation sites.*

Interconnectors add two stochastic properties to the network problem described in §(10.1) stochastic load draw or surplus and information sharing across the interconnector. This information is then connected to the SCADA system. Addition of this new location and resource to the transmission system.

The addition of the SCADA and the ability to control incoming and outgoing load along with the data nests within the previously illustrated SCADA network model. Again we can think of policy modelling for interconnectors at two levels.

First from a network view, a systems model representation allows for comprehensive checking of the security architecture under a variety of foreseeable scenarios. However, assigning likelihoods to these scenarios is somewhat complex. Data driven analysis based on historical trends is often unreliable in forecasting future directions for supply and demand and for certain types of problem, the bounds and moments of the appropriate distribution of outcomes are uncertain and prone to structural change.

In addition to the SCADA and related interconnector security scenarios we will also look at and potentially include the following future state problems (see WP2 Deliverable D2.3 Section 4.2.1 for detailed information).

10.3 Corporate Network

Corporate networks may have developing extensive links between the corporate network and the CNI SCADA systems. This may result in interfaces between the CNI SCADA systems and the internet. This opens the CNI systems and equipment to a vast array of attacker and attack methods which the systems have not been built to defend against. The examples provided in sections 4-9 provide the elements of a model for a corporate network.

10.4 Smart Metering

Smart metering is a new innovation whereby the meters at the end user are networked via the billing and distribution networks. The implications for security scenarios relating to the SCADA system and the corporate networks (in this case the potential security interaction between the various agents in the electricity production and distribution system in the UK).

The level of abstraction in terms of applying the project modelling framework to the smart metering case will be determined over the period M12 to M24 and incorporated into deliverable D6.2 and D8.2.

11. Conclusions

This deliverable has outlined a modelling framework designed to operate on a multi-scale basis for a variety of security modelling problems. We propose to nest a systems model architecture in to an economic setting. For various different problem types we can focus on modelling the specific architecture of the system delineating policy in the context of hierarchies.

As we move to more macro problems, we can retrench and generalize the systems architecture and focus on either risk based analysis using real options or game theory to model strategic interactions between agents in the security system (for instance attackers and targets in a policy game).

We have outlined in detail a set of potential models for airports and critical infrastructure that are relevant for the case studies developed within the SECONOMICS project. Further work will focus on refining the linkages with WP5, D5.2 during the period M12 to M24 and integrate the framework more closely with the cases provided in WP3, D3.3 final requirements.

BIBLIOGRAPHY

[1] M. Collinson, B. Monahan, and D. Pym. *A Discipline of Mathematical Systems Modelling*. College Publications, 2012. ISBN ISBN 978-1-904987-50-5.

[2] Matthew Collinson, Brian Monahan, and David Pym. Semantics for structured systems modelling and simulation. In *Proc. Simutools 2010*. ACM Digital Library, ISBN 78-963-9799-87-5, 2010. ISBN 78-963-9799-87-5.

[3] M. Collinson and D. Pym. Algebra and logic for resource-based systems modelling. *Mathematical Structures in Computer Science*, 19:959–1027, 2009. doi:10.1017/S0960129509990077.

[4] M. Collinson, B. Monahan, and D. Pym. A logical and computational theory of located resource. *Journal of Logic and Computation*, 19(b):1207–1244, 2009.

[5] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems: Concepts and Design*. Addison Wesley; 3rd edition, 2000.

[6] Core Gnosis. http://www.hpl.hp.com/research/systems_security/gnosis.html.

[7] Y. Beres, M. Casassa Mont, J. Griffin, and S. Shiu. Using security metrics coupled with predictive modeling and simulation to assess security processes. In *Proc. Empirical Software Engineering and Measurement (ESEM) 2009*, pages 564–573. IEEE Computer Society, 2009.

[8] David Pym and Simon Shiu. Security analytics: Bringing science to security management. *IISP Pulse*, 4(Summer):12–13, 2010.

[9] Adrian Baldwin, Yolanta Beres, Geoffrey B. Duggan, Marco Casassa Mont, Hilary Johnson, Chris Middup, and Simon Shiu. Economic methods and decision making by security professionals. In Bruce Schneier, editor, *Economics of Information Security and Privacy III*, pages 213–238. Springer, 2012.

[10] D. Pym. *Trust Economics: A Systematic Approach to security decision-making*. Hewlett-Packard, 2011. http://www.hpl.hp.com/news/2011/oct-dec/Final_Report_collated.pdf.

[11] A. Beautement, R. Coles, J. Griffin, C. Ioannidis, B. Monahan, D. Pym (Corresponding Author), A. Sasse, and M. Wonham. Modelling the Human and Technological Costs and Benefits of USB Memory Stick Security. In M. Eric Johnson, editor, *Managing Information Risk and the Economics of Security*, pages 141–163. Springer, 2008.

[12] C. Ioannidis, D. Pym, and J. Williams. Investments and trade-offs in the economics of information security. In Roger Dingledine and Philippe Golle, editors, *Proceedings of Financial Cryptography and Data Security '09*, volume 5628 of *LNCS*, pages 148–166. Springer, 2009. Preprint available at http://www.cs.bath.ac.uk/~pym/IoannidisPymWilliams-FC09.pdf.

[13] Christos Ioannidis, David Pym, and Julian Williams. Information Security Trade-offs and Optimal Patching Policies. *European Journal of Operational Research*, 216(2): 434–444, 2011. doi:10.1016/j.ejor.2011.05.050.

[14] Y. Beres, D. Pym, and S. Shiu. Decision support for systems security investment. In *Proc. Business-driven IT Management (BDIM) 2010*. IEEE Xplore, 2010.

[15] Marco Casassa Mont, Yolanta Beresnevichiene, David Pym, and Simon Shiu. Economics of identity and access management: Providing decision support for investments. In *Proc. Business-driven IT Management (BDIM)*. IEEE Xplore, 2010.

[16]

[17] L.A. Gordon and M.P. Loeb. The Economics of Information Security Investment. *ACM Transactions on Information and Systems Security*, 5(4):438–457, 2002.

[18] D.J. Pym, P.W. O'Hearn, and H. Yang. Possible Worlds and Resources: The Semantics of **BI**. *Theoretical Computer Science*, 315(1):257–305, 2004.

[19] John Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of the Seventeenth Annual IEEE Symposium on Logic in Computer Science, Copenhagen, Denmark, July 22-25, 2002*, pages 55–74. IEEE Computer Society Press, 2002.

[20] David Pym and Chris Tofts. Systems Modelling via Resources and Processes: Philosophy, Calculus, Semantics, and Logic. In L. Cardelli, M. Fiore, and G. Winskel, editors, *Electronic Notes in Theoretical Computer Science (Computation, Meaning, and Logic: Articles dedicated to Gordon Plotkin)*, volume 107, pages 545–587, 2007. Erratum (with Collinson, M.) *Formal Aspects of Computing* (2007) 19: 551–554.

[21] David Pym and Chris Tofts. A calculus and logic of resources and processes. *Formal Aspects of Computing*, 18(4):495–517, 2006. Erratum (with Collinson, M.) *Formal Aspects of Computing* (2007) 19: 551-554.

[22] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25(3): 267–310, 1983.

[23] P.W. O'Hearn and D.J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2):215–244, June 1999.

[24] S. Gilmore and J. Hillston. The PEPA Workbench: A Tool to Support a Process Algebra-based Approach to Performance Modelling. In *Proceedings of the Seventh International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, number 794 in Lecture Notes in Computer Science, pages 352–368. Springer-Verlag, 1994.

[25] M. Hennessy and G. Plotkin. On observing nondeterminism and concurrency. In *Proceedings of the 7th ICALP*, volume 85 of *Lecture Notes in Computer Science*, pages 299–309. Springer-Verlag, 1980.

[26] R. Milner. *Communication and Concurrency*. Prentice Hall, New York, 1989. ISBN 0-13-114984-9 (hardcover) 0-13-115007-3 (paperback).

[27] Colin Stirling. *Modal and Temporal Properties of Processes*. Springer Verlag, 2001.

[28] D.J. Pym. *The Semantics and Proof Theory of the Logic of Bunched Implications*, volume 26 of *Applied Logic Series*. Kluwer Academic Publishers, 2002. Errata and Remarks maintained at publisher's website and at: http://homepages.abdn.ac.uk/d.j.pym/pages/BI-monograph-errata.pdf.

[29] C. Baier and J.-P. Katoen. *Principles of model checking*. MIT Press, 2008.

[30] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 2000.

[31] G. Birtwistle. *Demos — discrete event modelling on Simula*. Macmillan, 1979.

[32] M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic Model Checking for Performance and Reliability Analysis. *ACM SIGMETRICS Performance Evaluation Review*, 36(4):40–45, 2009.

[33] C.G. Cassandras and M.I. Clune and P.J. Mosterman. Hybrid System Simulation with SimEvents. In *Discrete Event Systems, 2006 8th International Workshop on (WODES)*, pages 386–387, 2006.

[34] A. Beautement and D. Pym. Structured systems economics for security management. In T. Moore, editor, *Proc. WEIS 2010, Harvard*, 2010. http://weis2010.econinfosec.org/papers/session6/weis2010_beautement.pdf.

[35] M. Collinson, D. Pym, and B. Taylor. A framework for modelling security architectures in services ecosystems. In *Proc. ESOCC 2012*, volume 7592 of *LNCS*, pages 64–79. Springer, 2012.

[36] Donn B. Parker. *Fighting Computer Crime, a New Framework for Protecting Information*. John Wiley and Sons, 1998.

[37] R.L. Keeney and H. Raiffa. *Decisions with multiple objectives: Preferences and value tradeoffs*. Wiley, 1976. ISBN 0471465100.

[38] Francisco J. Ruge-Murcia. Inflation targeting under asymmetric preferences. *Journal of Money, Credit, and Banking*, 35(5), 2003.

[39] H.R. Varian. A bayesian approach to real estate assessment. In S.E. Feinberg and A. Zellner, editors, *Studies in Bayesian Econometrics in Honor of Leonard J. Savage*, pages 195–208. North-Holland, 1975.

[40] A. Zellner. Bayesian prediction and estimation using asymmetric loss functions. *Journal of the American Statistical Association*, 81:446–451, 1986.

[41] Y. Beres, D. Pym, and S. Shiu. Decision support for systems security investment. In *Proc. Business-driven IT Management (BDIM) 2010*. IEEE Xplore, 2010.

[42] L.A. Gordon and M.P. Loeb. *Managing Cybersecurity Resources: A Cost-Benefit Analysis*. McGraw Hill, 2006.

[43] D. Rogers and L.C.G. Williams. *Diffusions, Markov Processes, and Martingales*. Cambridge Mathematics Library, 2000.

[44] K. Taksumi and M. Goto. Optimal timing of information security investment: A real options approach. In T. Moore, D. Pym, and C. Ioannidis, editors, *Economics of Information Security and Privacy*. Springer, 2010. Proceedings of WEIS 2009, London.

[45] D. Kahneman and A. Tversky. Prospect theory: An analysis of decisions under risk. *Econometrica*, 47:313–327, 1979.

[46] Larry G. Epstein and Stanley E. Zin. Substitution, Risk Aversion, and the Temporal Behavior of Consumption Growth and Asset Returns I: A Theoretical Framework. *Econometrica*, 57(4):937–969, July 1989.

[47] R. Abel. Asset Prices under Habit Formation and Catching up with the Joneses. *The American Economic Review*, 80(2):38–42, 1990.

[48] O. Loistl. The Erroneous Approximation of Expected Utility by Means of Taylor's Series Expansion: Analytic and Computational Results. *American Economic Review*, 66(5): 904–910, 1976.