

# The Role of Dynamic Security Policy in Military Scenarios

Helge Janicke<sup>1</sup>, Linda Finch<sup>2,3</sup>

<sup>1</sup>STRL, De Montfort University, Leicester, UK

[heljanic@dmu.ac.uk](mailto:heljanic@dmu.ac.uk)

<sup>2</sup>General Dynamics United Kingdom Limited, Newbridge, UK

<sup>3</sup>Communications Research Centre, Cardiff University, Cardiff, UK

[linda.finch@generaldynamics.uk.com](mailto:linda.finch@generaldynamics.uk.com)

**Abstract:** *The military is moving towards Network Enabled Capability (NEC) where the emphasis is on resource sharing within national contingents and on a coalition basis, facilitated by the Network. Future capability is predicated on the core attribute of agility. NEC is expected to enable the dynamic formation of communities of interest and the rapid reorganisation of resources as required by military commanders. Through the application of a suitable security policy framework to a small-scale case study, this paper tests the assertion that the ability to express, verify and implement flexible security policy is essential to achieve the agility required.*

**Keywords:** *NEC, DBSy, Dynamic Policy, Information Security*

## Introduction

This journal paper is an elaboration of the version presented at the European Conference on Information Warfare (ECIW) held in July 2007 (Janicke and Finch, 2007). It presents ongoing research grounded in the Network Enabled military organisation of the future where operational agility is a fundamental strategy in dealing with the new type of adversary. Agility, autonomy and a more holistic approach to information sharing are key requirements in supporting an infrastructure capable of adapting to reflect the intent of the commander as situational conditions dictate. Clearly this raises some unique challenges to security since it calls for a flexible architecture and infrastructure underpinned by dynamic policy. Furthermore, it is necessary to provide assurance that a more flexible approach does not result in less security. This may be achieved in part, through verifiable security policy rules constructed using a formal language.

Traditionally, emphasis has been placed in strong separation between security classes. In turn this has resulted in the development of ‘stove piped’ systems unable to respond to the requirements of agile, interoperable and collaborative environments where information sharing is a prerequisite. Sharing introduces a number of challenges, not least of which is security. In their study, Phillips *et al.* (2002) observed that members of a coalition hold different security requirements and apply different security labels to items thus creating problems for translation between systems and members. In our research it is proposed that a radical review of the security challenge is required to support the desired level of flexibility necessary in future military operations,. This includes adopting the approach of ‘fit for purpose’ security where, “in order to be trustworthy a system just has to be sufficiently secure to be fit for purpose” (Kuhlmann and Ghering, 2003) a view at odds with traditional approaches.

This suggestion is not made lightly: it is recognised that in a changing environment short-term expediency must be balanced against the longer-term impact on information and resources.

Furthermore, it is appreciated that excessive rigidity in security may impede the task and result in system workarounds; the risks of which may not be captured or mitigated. Military systems operate with limited resources in hostile environments therefore a security solution must introduce minimal additional impact to the system both in terms of resource footprint and management overhead. The authors believe that if systems can be designed and constructed which are capable of dynamically accommodating and implementing changing security requirements in accordance with pre-defined conditions, then the human user can be released to focus on their core activities rather than system management tasks. The net result is a more flexible operating environment with the potential to adapt to changing circumstances with minimal operator intervention; an obvious benefit in hostile environments. It is proposed that there is an opportunity to realise 'fit for purpose' security through the synthesis of policy, enforcement mechanism and architecture. This research explores the first of these components, policy, and demonstrates how dynamic requirements relevant to the military organisation can be captured, formalised and validated.

This process commences with the capture and expression of high-level separation and sharing requirements in a format that is easily comprehensible to all stakeholders. The Domain Based Security Architecture (DBSy)<sup>®</sup> approach and notation from QinetiQ (Hughes, 2002) was developed as an intuitive but non-formal model specifically for this purpose. DBSy does not, however, accommodate changing requirements. As a part of some related research potential additions to the DBSy notation are being explored. These modifications will enable policy developers to capture changing and conditional behaviour which results in non-persistent connectivity between communities. For the purposes of this paper standard DBSy notation is used, without any proposed additions, to provide a visual representation of the case study.

High-level requirements are subsequently refined into rules that can be implemented and read by individual system components to enforce desired behaviour. Refinement is an iterative process and allows for the identification and adjustment of ambiguities in the requirements. The process requires a framework and language and, crucially for this case study, it must also convey the temporal and dynamic nature of the rules. For this paper the SANTA security policy framework (Janicke 2007, Janicke 2006, Siewe 2005) was used to model the requirements. SANTA has been developed to capture the dynamic changes in security requirements over time and on the occurrence of events.

The SANTA security policy framework comprises three main components. Firstly, the policy model allows for the expression of security requirements such as:

- *Authorisation*: Defines access to resources such as the communication infrastructure.
- *Delegation*: Defines which stakeholders can pass on their privileges to others.
- *Obligation*: Defines conditions under which activities must be performed.

The model has a formal underpinning in Interval Temporal Logic (ITL) Cau *et al.* (2007) which is suitable for expressing temporal dependencies and the dynamic change of these requirements in a compositional manner. As has been noted by others (e.g. Becker, 2006 and Abadi, 2003), a mathematical approach is essential to achieve a level of assurance in the policy. This is particularly so where policies are employed in (security) critical applications, where non-conformance to stated security requirements endangers the mission's objective or can lead to loss of life. Secondly, linguistic support is provided to abstract from the underlying mathematical description and appeal to a more business-oriented audience. The semantics of the language is, however, unambiguous as it directly translates into the

underlying model. Thirdly, tool-support for the specification and analysis of policies is provided as part of the SANTA framework. Currently this includes the Security Policy Analysis Tool (SPAT) that allows for the early prototyping and validation of policies and thus aids policy developers to translate high-level requirements (for example those expressed in DBSy) into concrete policies. Finally it is possible to develop, based on the theoretical work presented in (Janicke, 2007), dedicated system components that enforce a given policy.

The formal foundation and the expressiveness of the policy model with respect to dynamic change and temporal dependencies of security requirements is used to show how high-level DBSy specifications can be represented by policies. Classes of requirements that cannot be adequately represented using the current DBSy notation are identified and modelled. The approach described in this paper exploits the advantages of both methodologies: the intuitive and high-level modelling capabilities of DBSy with the expressive and formal language of SANTA. This complementary approach should appeal to a wider policy development community and ultimately increase the opportunities for developing 'fit for purpose' security.

## **Related Work**

Security models and frameworks have been the subject of academic and commercial interest for over three decades where much of the early work originates from studies conducted specifically for the military sector. One of the most influential models to emerge at the time was the Bell and LaPadula model (Bell and LaPadula, 1975), which is primarily concerned with expressing and supporting multiple levels of security on a single system. Objects are assigned a protective marking such as *Top Secret*, *Secret* and *Restricted* and may additionally be assigned a codeword relevant to the particular activity. The composition of protective marking and codeword forms a lattice structure of object dominance that is underpinned through mathematical theory: an area that has been studied by other researchers such as (Denning, 1976).

A significant issue with the Bell and LaPadula model arises from the complexity of implementation. Furthermore the model is principally concerned with preserving confidentiality. Whilst this is critical to the military community, research from (Clark and Wilson, 1997) concluded that in the commercial sector preserving integrity was of paramount importance. They developed a different model based on two central concepts: separation of duty and well-formed transactions. When implemented, this model also suffered from the complexity generated by numerous rules to control the multitude of system transactions.

The Role Based Access Control (RBAC) model aims to reduce this complexity and overhead by associating permissions with roles that in turn represent the job functions within an organisation. Permissions are therefore not individually associated with users. A user occupies a single role relevant to their function, which is clearly in the spirit of the Clark and Wilson 'separation of duty' principle. RBAC recognises inheritance where a 'parent' role may contain the capabilities of a subordinate role in addition to its own specific permissions. Although the potential benefit of using roles for permission assignment had been active throughout the late 1980s and early 1990s, the work was fragmented and application-specific. A general-purpose framework and formal analysis of systems utilising roles was missing; a requirement fulfilled by (Ferraiolo and Kuhn, 1992) who integrated and augmented existing principles and developed a generalised RBAC model. Research has continued: for example Sandhu *et al.* (1996) introduced a framework of four conceptual models that captured different levels of defined RBAC functionality and as a result can offer a choice of

complexity and capability to customers and system developers alike. New models based on the original RBAC standards target specific organisational requirements and continue to emerge.

Earlier models were limited in their ability to capture the dynamic aspects of security policy. The Bell and LaPadula model in particular assumed a static machine state where access did not change. In most environments this security policy is unrealistic and therefore encouraged alternative approaches to be developed. Thomas and Sandhu (1997) for example, suggested that workflow systems required security policy where "... permissions are constantly monitored and activated and deactivated in accordance with emerging context associated with progressing tasks..." and constructed the Task Based Access Control (TBAC) model. This is designed to support 'active security' models based on the underlying principle of 'just-in-time' and 'need to do' permissions; an outcome that upholds the philosophy and motivation for this paper.

Unlike other frameworks, SANTA allows the expression of composite policies that capture temporal properties using a formal language based on a sound logical framework. Flexibility must be balanced against assurance therefore the impact of a policy change must be assessed to provide a level of confidence that underlying security has not been compromised. Mitigating action can then be taken where necessary. The Security Policy Analysis Tool (SPAT) allows a policy developer to analyse the modelled policy at runtime and refine the model before implementation.

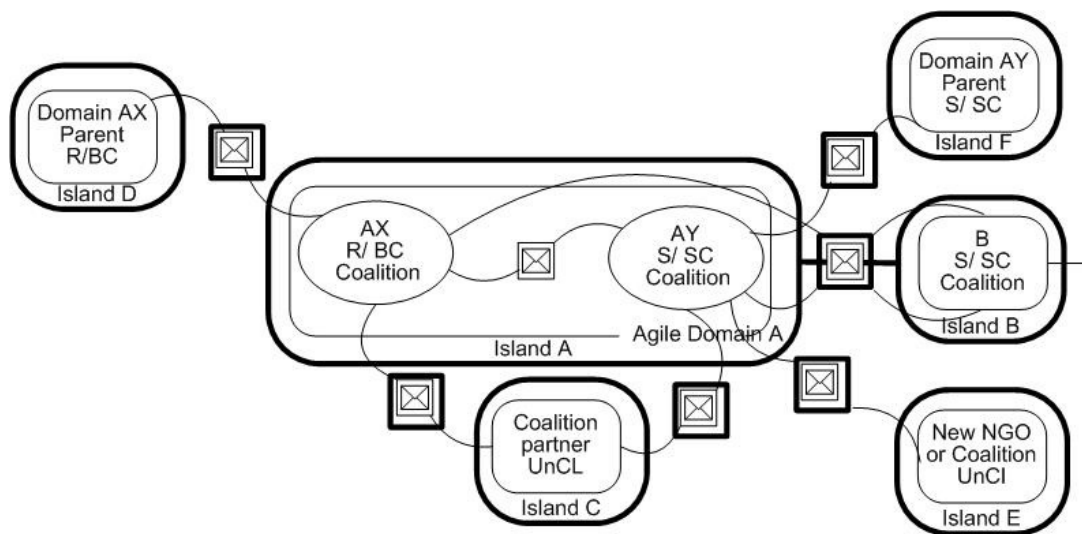
## Case Study

This investigation is based on a small-scale scenario representative of a military requirement, which may be summarised as follows:

**Military Need:** On an expeditionary operation, the deployed order of battle (ORBAT) comprises groups of military assets that exist within pre-defined security domains. NEC envisages that commanders may create any desired subset as an agile group for a limited period, to undertake a specific task. The elements of the agile group will need to intercommunicate and communicate with their functional parent organisations. On completion of the task, either the agile group will dissolve or its composition will be adjusted for another task. The commander's ability to task organise an agile group should not be constrained by security considerations, in terms of what can be task organised, the free passage of information between agile group elements or the time taken to establish the group.

## Case Study Model

The military need described above has been modelled using standard DBSy notation. The output, an Information Security (InfoSec) Architecture model, which is the composite of InfoSec Business and Infrastructure models, is shown in Figure 1. For brevity, a limited description of the DBSy notation used in Figure 1 is provided; additional information may be found in other publications such as (Robinson, 2001). It is quite credible that on an expeditionary operation, military assets will be drawn from different security domains. The model therefore attempts to illustrate this possibility; however, in order to maintain simplicity we have deliberately limited the number of classes that might normally be expected in a coalition deployment.



**Figure 1:** InfoSec Architecture model of case study

Domains, shown in Figure 1 as an ellipse or rounded rectangle, contain users with similar information requirements and security constraints. Within a domain information may be exchanged freely, however, inter-domain communication is subject to restrictions. In Figure 1 domains AX and AY are members of the agile mission group. In accordance with military need, the agile group has internal communications capability (messaging) and external communication with their respective functional parents. Furthermore they are permitted to communicate with other groups and we may visualise domain B as a Headquarters. Coalition partners may be military or non-military non-government organisations such as humanitarian groups called upon at different times during the operation.

In DBSy terminology, domains are hosted on Islands of infrastructure shown by the emboldened rectangles. These illustrate that strong separation is required between different communities although controlled sharing via the Causeway (emboldened square) is permitted between any two domains directly connected by a line or permitted business connection. Each connection is annotated and in this case the envelope construct indicates that messaging is permitted using this link. Security enforcement mechanisms are hosted on the Causeway.

An important omission from Figure 1 is the concept of temporal capability. Although the permitted business connection indicates that communication is authorised between domains AX and B, in reality this is conditional; specifically when AY is unavailable to AX. This is

referred to by the authors as a non-persistent connection (NPC) and in related work it is proposed that the temporary and conditional nature of the connection should be identified within the DBSy framework and notation. Due to space limitations the policy statements used for the formal modelling have deliberately been restricted to include only those that illustrate dynamic requirements as follows:

*Req 1:* AX is permitted to communicate with AY and vice versa.

*Req 2:* AY is permitted to communicate with B and vice versa.

*Req 3:* AX is not permitted to communicate with B unless AY is unavailable.

*Req 4:* AX and AY are permitted to receive data from the parent organisation subject to the security level of that parent. This is either Restricted (R) or Secret (S) for Basic Check (BC) or Security Cleared (SC) users.

*Req 5:* AX and AY are permitted to communicate with the coalition partner however this is subject to constraint: if either is also communicating with B they may not establish or hold a connection with the coalition partner. Precedence will depend on the situation at the time. When engaged with the adversary precedence is given to the coalition partner. The clearance level of the coalition partner may not be known.

## **Policy Modelling**

Before requirements can be refined all stakeholders and their potential interactions must be identified. In this example stakeholders are the members of the different domains, thus a domain represents a community able to interact. The stakeholders can be identified from the DBSy diagram with relative ease. Establishing the potential interactions is more difficult, as the diagram can only express the existence (or non-existence through omission) of a business connection. Analysing the security requirements 1 to 5 it however becomes clear that the general interaction “communicate” consists of finer grained activities such as “receive data” (see Req 4) or “establish connection”, “hold connection” (See Req 5). It is also necessary to identify observable attributes and events. Examples for observable attributes are “security level” of data or “clearance level” of stakeholders. Examples of observable events are for example “engaged with the adversary” (see Req 5) or the fact that an interaction is (or has been) taking place. This is summarised in Table 1.

Stakeholders (Domains)			
AX	Member of the agile group	AX Parent	Functional Parent of AX
AY	Member of the agile group	AY Parent	Functional Parent of AX
B	Head Quarters		
CP	Coalition Partner		

Interactions	
communicate	Business Connection between two stakeholders encompasses:
open	establishing a connection over which communication takes place
close	closing a connection. We assume here that “holding” a connection means that the connection was open and has not been closed.
send (data)	send data over an established connection.
receive (data)	receive data over an established connection.

Attributes	
level(data)	Security Level attached to data (“unclassified”, “secret” ... “top-secret”)
clearance(stakeholder)	Clearance level of stakeholder (same as level)
engaged(stakeholder)	The stakeholder is engaged with the adversary (Boolean)

**Table 1**, Summary of Stakeholders and Interactions

The SANTA policy model is rule-based, that is a policy can be expressed as a collection of rules. Rule-based approaches are common to many policy languages; XACML (OASIS, 2005), KAOS Uszok *et al.* (2003) and Ponder (Damianou, 2002) allow for a declarative description of the requirements that can then be translated into a lower-level operational form (Janicke, 2007). Typically each rule captures a single requirement, such as the DBSy requirement that “members of the same domain can communicate without constraint”. Communication is modelled as sending and receiving messages. This authorisation requirement would translate into the following rules between domains D1 and D2:

*allow* (S,O, “communicate”) when S=O  
*allow* (S,O,A) when exists D1, D2 in objects : {  
 member(S, D1) and member(O, D2) and *allow*(D1, D2, A) }

The quotation marks indicate that “communicate” is a defined interaction. In contrast, the *S* and *O* are variables that range over the set of stakeholders. In the policy model it is possible to distinguish between subjects and objects. Subjects are the initiators of an interaction whereas objects are the target. In this paper it is assumed that both sets are identical. The first rule states a positive authorisation if the initiator *S* of “communicate” is the same as the target *O*. Since individuals typically establish communication, the permission is propagated in the next rule to the member of the domain if the permission was granted at domain level. Similarly the decomposition of the interaction “communicate” can be used to propagate rights accordingly to the more refined constituents:

*allow* (S,O,“open”) when *allow* (S,O,“communicate”)  
*allow* (S,O,“close”) when *allow* (S,O,“communicate”)

*allow (S,O,“send”) when allow (S,O,“communicate”)*  
*allow (S,O,“receive”) when allow (S,O,“communicate”)*

By granting the right to “communicate” to a stakeholder, the stakeholder's right to “open” and “close” a connection can be derived in addition to “send” and “receive” data. A DBSy permitted business connection could be expressed using a set of rules that are enforced at the Causeway. Due to the symmetry of business connections the policy rules will have to capture both directions of the communication:

*allow (“AX”, “B”, “communicate”) when true*  
*allow (“B”, “AX”, “communicate”) when true*

A policy language provides greater expressiveness when defining the constraints on a business connection. The condition under which the communication is permitted for example may be more complex than domain membership. Policies can also explicitly state conditions for a *denial* of an interaction, which cannot be expressed in the higher-level DBSy diagram. To resolve conflicts for example where both a permission and a denial can be derived from the policy, the policy contains decision rules that enable conflict resolution.

A key difference between SANTA policy rules and the aforementioned alternative models is that a rule can explicitly express temporal dependencies, viz. the condition describes a behaviour that has been observed in the past, rather than a simple condition. The behaviour is assumed to be observable by the Causeway that enforces the policy. As an example, the following rule would permit members of domain AX to communicate with B if they have not previously been a member of domain E.

*allow (“AX”, “B”, “send”) when true*  
*deny (S, “B”, “send”) when now member(S, “AX”) and sometime member(S, “E”)*  
*decide (S,O,A) when allow(S,O,A) and not deny(S,O,A)*

This shows how even complex constraints on a business connection can be easily expressed in a policy using positive authorisation, negative authorisation and decision rules. Similarly the requirements of the case study can also be expressed. The definition of a business connection between domains A and B may be abbreviated as the following simple policy:

*define bc(A,B) = {*  
    *allow (A,B,“communicate”) when true*  
    *allow (B,A,“communicate”) when true*  
*}*

This definition is now used to define the requirements for this case study as follows:

- *Req 1* : AX is permitted to communicate with AY and vice versa

*bc(“AX”, “AY”)*

- *Req 2* : AY is permitted to communicate with B and vice versa.

*bc(“AY”, “B”)*

- *Req 3* : AX is not permitted to communicate with B unless AY is unavailable.

*bc*(“AX”,“B”) *and*  
*deny* (“AX”,“B”,“send”) *when* *available*(“AY”) *and*  
*deny* (“AX”,“B”,“receive”) *when* *available*(“AY”)

The last rule of Req 3 shows that the high-level description of communication links (even of conditional ones) may not be sufficient to capture the original intent of the requirement precisely enough. It is unclear whether “AX” should be denied the permission to receive messages directly from “B” or not.

- *Req 4* : AX and AY are permitted to receive data from their parents

*allow* (“AX”,D,“receive”) *when* *parent*(“AX”,D)  
*allow* (“AY”,D,“receive”) *when* *parent*(“AY”,D)

Alternatively we could define this as a separate policy that is scoped:

*scope* ([“AX”,“AY”],*any*,“receive”) : ( *allow* (X,D,A) *when* *parent*(X,D) )

The *scope* operator can be used to limit the application of the policy to a specific set of stakeholder and interactions. The policy rule in the above example denotes simply that a stakeholder can engage in every interaction with his functional parent. This is then restricted to the stakeholders “AX” and “AY” and the interaction “receive”.

*Req 5* : is decomposed into several steps. Firstly, the change of precedence over time is ignored and the constrained communication is formulated. Secondly the dynamic change in the form of a policy composition is captured, as described below.

The policy is limited using the *scope* operator to apply to interactions between the relevant stakeholders (“AX”, “AY” interacting with “B” and “C”). The second line of the following policy establishes a business connection between “AX” and “B”, “C” as well as “AY” and “B”, “C”.

*scope* ({“AX”,“AY”}, {“B”,“C”}, *any*) : {  
*bc*(S, “C”) *and* *bc*(S, “B”) *and*  
*deny*(S,C,“open”) *when* *isCom*(“AX”,“B”) *or* *isCom*(“AY”,“B”)  
*oblige*(S,“C”,“close”) *when* *isCom*(S,“C”) *and* *now* *done*(S,“B”,“open”)  
}

The third line in the policy represents the constraint that none of the stakeholders can open a new connection to the coalition partner (“C”) if they are currently communicating with “B”. Holding a connection has not been defined as a primitive action; instead the view is taken whereby a connection remains established, equivalent to holding, all the time it is not closed

by either party. Line 4 states that if a connection is established with “B” (open) whilst communicating with the coalition partner (“C”) then the stakeholder is obliged to close the connection to the coalition partner immediately. In accordance with the intuition of *holding* a connection, ‘communicating’ is defined as behaviour such:

```
define isCom(X,Y) = {
  (done(X,Y,“open”) or done(Y,X,“open”)) and
  always not (done(X,Y,“close”) or done(Y,X,“close”))
}
```

isCom(X,Y) is true if in the past the connection has been opened by either X or Y and never subsequently closed by any of them.

The policy change reverses the precedence between communication with “B” and “C” therefore the following policy definition expresses the above policy where X replaces “B” and Y replaces the coalition partner “C”.

```
define Req5(X,Y) = scope( {“AX”,“AY”}, {“B”,“C”}, any) : {
  ... as above
}
```

The required change in precedence is formulated using the following policy composition:

```
scope( {“AX”,“AY”}, {“B”,“C”}, any) :
  repeat {
    unless adverse(S): Req5(“B”,“C”) ; aslongas adverse(S): Req5(“C”,“B”)
  }
```

Here, the *unless* operator denotes that the policy gives precedence to the communication with “B” (the headquarters) unless S (“AX” or “AY”) are engaged with the adversary. Following this the reverse precedence applies as long as S is engaged. This dynamic change repeats every time S engages with an adversary.

Finally, to capture the multi-level security restriction in *Req 4*, rules similar to those described in the Bell and LaPadula model are included:

```
allow (S,O,“receive”(data)) when now label(data) <= clearance(S)
allow (S,O,“send”(data)) when now label(data) <= clearance(O)
```

It is noted that traditional security policies are insufficient to manage cases where the security clearance of a domain is unknown. As a result, the authors are currently investigating the use of trust in the model. This would add an additional degree of flexibility as trust in coalition partners can be established over time, based on direct experience or reputation obtained from other (trusted) partners.

In summary, this section has illustrated how changing requirements can be captured and formalised using SANTA's dynamic policies. The increased conciseness of the description eliminates potential ambiguities of the informal requirements and is a necessary step towards a more detailed analysis of potential compromise paths.

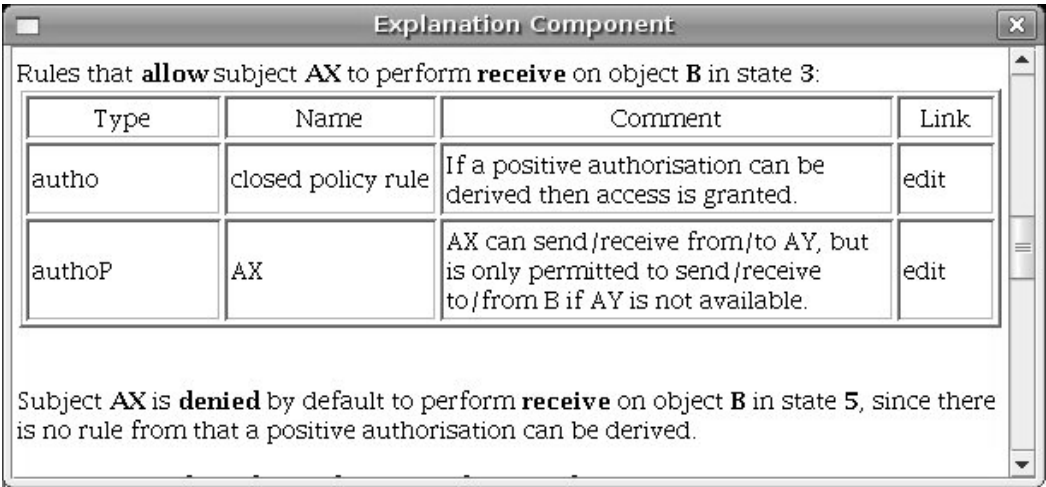
### Policy Analysis

This section illustrates how the requirements can be validated using the SPAT tool. Policy rules for requirements 1 to 3 were simulated under a small scenario in which domain AY is unavailable in states 2, 3 and 4. The access control matrix shown in Figure 2 is filtered to display the policy decisions for AX sending/receiving messages to/from B. The result validated the defined rule where a business connection from AX to B only existed in those states where domain AY was unavailable.

State : all	Subject : (AX)	Object : B	Action : (send) (receive)	Decision : all
0	AX	B	send	false
0	AX	B	receive	false
1	AX	B	receive	false
1	AX	B	send	false
2	AX	B	receive	true
2	AX	B	send	true
3	AX	B	receive	true
3	AX	B	send	true
4	AX	B	send	true
4	AX	B	receive	true
5	AX	B	receive	false
5	AX	B	send	false
6	AX	B	send	false
6	AX	B	receive	false

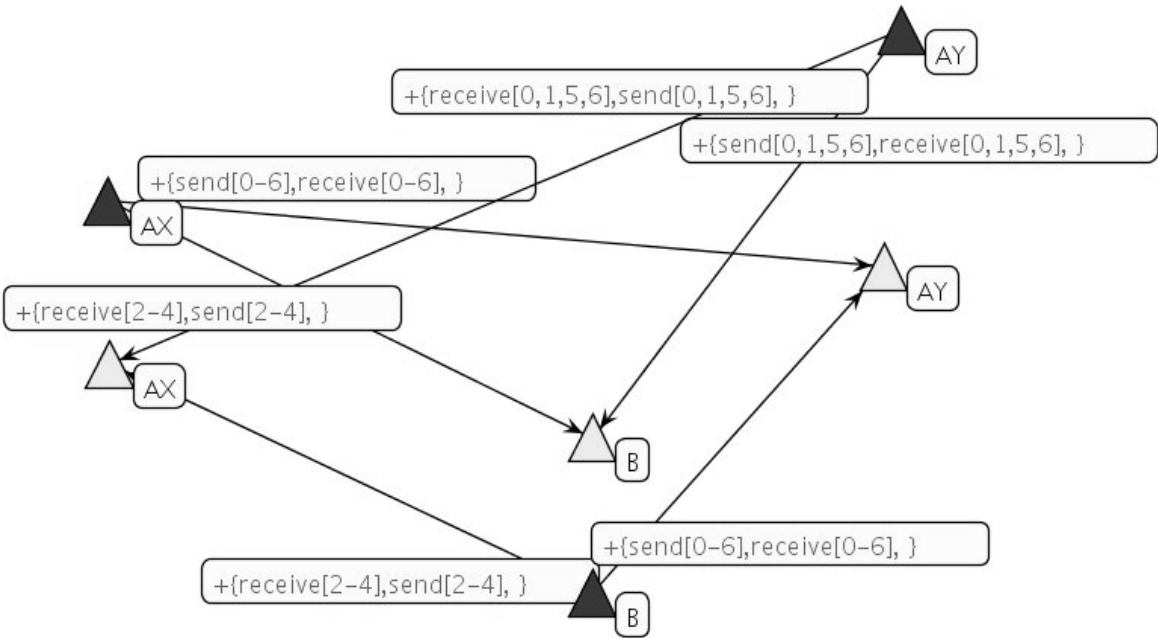
**Figure 2:** Access Control Matrix for communication between AX and B

To assist with policy development SPAT can provide feedback concerning the rules responsible for a particular policy decision. An explanatory note for the action “receive” in state 3 is shown in Figure 3.



**Figure 3:** Explanatory note from SPAT

The Access Control Matrix may also be viewed as a directed graph as shown in Figure 4.



**Figure 4:** Access Control Matrix graph for requirements (reqs.) 1,2 and 3

As SPAT distinguishes between the originator of an interaction and the target, each domain occurs in both roles. The dark triangle represents the originator and the light triangle the target. This is a valuable aid in both the analysis and validation of policies expressing non-persistent business connections in a specific scenario.

**Conclusions**

This research was designed to test the assertion that the ability to express, verify and implement flexible security policy is essential in order to achieve the agility demanded by a network enabled military organisation. The work presented in the paper concentrates on policy expression and verification; implementation is addressed as part of an on-going investigation. In the case study, the ability for domains to share information is permitted or denied based upon changing conditions in the environment. The results demonstrate that these security requirements can be expressed and validated using an existing and suitable security policy framework such as SANTA. The process started using DBSy which allowed the authors to capture high-level separation and sharing requirements in a graphical and easily comprehensible format. However DBSy has no provision for expressing change; a limitation which is under investigation as part of related work. It has been shown how the DBSy models can be translated into a more expressive policy language such as SANTA. The formalisation process allowed for the identification and correction of ambiguities in the high-level policy that would not have been visible otherwise. Given the policy representation of the DBSy model, it was possible to revisit the original requirements and capture more sophisticated properties associated with individual business connections. These include, for example, the

conditional statement in Req. 3. Furthermore this work has shown how high level communication actions can be decomposed into lower level primitives such as ‘open/ close’ and ‘send/ receive’. This was particularly relevant for Req. 4 where the ‘receive’ interaction was explicitly stated. The expressiveness of the policy language is evident in the formalisation of Req. 5 where the communication state is captured by the behavioural description ‘isCom(X,Y)’ in terms of communication primitives.

The SPAT tool enabled analysis of a subset of the policy to confirm that the defined rules behaved as expected and required. Validation of security mechanisms is particularly important for applications in the military domain and it is conjectured this will become increasingly critical in order to assess the impact of dynamic changes to the security profile of the target system.

In this work it has been demonstrated that flexibility in resource organisation can be controlled through security policy and furthermore the use of a formal language provides a level of proof and confidence that is required in a military system. As a result, the authors believe that this constitutes an initial step in developing systems that can achieve the military requirements of flexibility and high assurance which will become more prevalent in NEC.

## **Future Work**

Dynamic policy is an emerging area, particularly its conjunction with high assurance systems. Considerable work remains to develop existing toolsets and address the challenges of architecture and enforcement mechanisms where dynamic policy will be implemented. The tools used to develop requirements and produce policy must be accessible to different skill-sets, therefore an opportunity exists to close the gap between complex and specialist frameworks such as SANTA with the more intuitive and graphical approaches like DBSy.

The SANTA policy language is currently being extended to allow for the expression of trust policies. Trust policies define how trust in other stakeholders is established based on their behaviour observed at the Causeway or fed into the system by user interaction. The aim is to allow system administrators to define suspicious behaviours in terms of policy rules that decrease the level of trust in another stakeholder. If the trust level falls below a predefined threshold the stakeholder may either be denied access to sensitive resources or be required to fulfil certain obligations when access is made. Part of this development work also involves the development of a policy compiler that generates enforcement code for the Causeway based on the underlying formal semantics of the policy language. The compilation of policies into enforcement code has potential advantages over the interpretation of policy rules at runtime in terms of resource usage and tempo of decision making.

## References

- Abadi M. (2003) Logic in Access Control, *Proceedings of LICS'03*, IEEE Computer Society Press, Vol 15, pp 228-233.
- Becker M., Fournet C., and Gordon A. (2006) SecPAL: design and Semantics of a Decentralized Authorisation Language. *Technical Report*, Microsoft Research Cambridge (UK).
- Bell D., LaPadula L. (1975) Secure computer system unified exposition and multics interpretation, *Technical Report*, MTR-2997, MITRE.
- Cau A., Moszkowski B., Zedan H. (2007), The ITL homepage [online], DMU, <http://www.cse.dmu.ac.uk/STRL/ITL>
- Clark, D and Wilson, D. (1997) A Comparison of Commercial and Military Computer Security Policies in *IEEE Symposium on Security and Privacy* pp. 184-194.
- Damianou, N. (2002) *A Policy Framework for Management of Distributed Systems*, PhD Thesis, University of London.
- Denning D. (1976) A lattice model of secure information flow, *Communications of the ACM* Vol 19, pp 236-243.
- Ferraiolo, D and Kuhn, D. (1992) Role-Based Access Control *Proceedings of the NIST-NSA National (USA) Computer Security Conference*, pp. 554-563.
- Hughes, K. (2002) *Domain Based Security: Enabling security at the level of applications and business processes*. [online], QINETIQ, [http://www.qinetiq.com/home/security/information\\_and\\_network\\_security/security\\_consultancy\\_practice/Domain\\_Based\\_Security.SupportingPar.0001.File.pdf.p4](http://www.qinetiq.com/home/security/information_and_network_security/security_consultancy_practice/Domain_Based_Security.SupportingPar.0001.File.pdf.p4).
- Janicke H and Finch, L. (2007) The Role of Dynamic Security Policy in Military Scenarios. *Proceedings of the 6<sup>th</sup> European Conference on Information Warfare*. Shrivenham, UK.
- Janicke H., Cau A., Siewe F., Zedan H., Jones K. (2006) A Compositional Event & Time-based Policy Model. *Policy 2006*, London, Ontario, Canada.
- Janicke H., Siewe F, Jones K., Cau A., Zedan H. (2005) Analysis and Runtime-Verification of Dynamic Security Policies. *DAMAS 2005 at AAMAS'05*, Utrecht, Netherlands
- Janicke H. (2007) *The Development of Secure Multi-Agent Systems*. PhD Thesis, STRL, DeMontfort University, Leicester (UK).
- Khulmann, D and Gehring, R. (2003) "Trusted Platforms, DRM and Beyond" in E. Becker et al. eds. *Digital Rights Management; Technological, Economic, Legal and Political Aspects*, Berlin, Springer.
- OASIS (2005) eXtensible Access Control Markup Language (XACML) version 2.0. [online] [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=xacml#XACML20](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml#XACML20)

Phillips, C.E, Ting, T. C, and Demurjian, S. A. (2002) Information Sharing in Dynamic Coalitions. *Proceedings of seventh ACM symposium on access control models and technologies*, California, USA, pp. 87-96.

Robinson, C. (2001) *Security Requirements Models to Support the Accreditation Process*, [online], QINETIQ, [http://www.qinetiq.com/home/security/information\\_and\\_network\\_security/security\\_consultancy\\_practice/Domain\\_Based\\_Security.SupportingPar.0006.File.pdf](http://www.qinetiq.com/home/security/information_and_network_security/security_consultancy_practice/Domain_Based_Security.SupportingPar.0006.File.pdf).

Sandhu, R, Coyne, E, Feinstein, H and Youman, C. (1996) Role-Based Access Control Models. *IEEE Computer*, Vol 29, No. 2, pp. 38-47.

Siewe F. (2005) *A Compositional Framework for the Development of Secure Access Control Systems*, PhD Thesis, STRL, De Montfort University, Leicester (UK).

Thomas, R. K. and Sandhu, R. S. (1997), Task Based Authorisation Controls (TBAC): A Family of Models for Active and Enterprise-orientated Authorisation Management. *Proceedings of the IFIP WG11.3 Workshop on Database Security*, Lake Tahoe, California, pp. 166-181.

Uszok A., Bradshaw J., Jeffers R., Suri N., Hayes P., Breedy M., Bunch L, Johnson M., Kulkarni S. and Lott J. (2003) KAoS Policy and Domain Services: Toward a description-logic approach to policy representation, deconfliction and enforcement. *Proceedings POLICY 2003 Policies for Distributed Systems and Networks*, Lake Como, Italy, pp. 93-96.