Security Issue Detection and Mitigation Patterns for Product Line Resource Variation

Rick Dove, dove@parshift.com

■ **ABSTRACT**
Product Line Engineering (PLE) builds upon an Agile Architectural Pattern—with reusable resources, evolving resource variations, and a standardized interconnect and sustainment infrastructure. Commercial PLE systems attract alternative resource suppliers, such as automotive parts. Defense PLE systems typically result from acquirers encouraging Open System Architectures to enable alternative resource suppliers. Alternative resource suppliers are a major resource variation source in product line engineered systems. Product line engineered systems are systems of systems with potential for complex interactions and unintended emergent behaviors. This article focuses on PLE cyber-physical-social system products, the security issues resource variation can introduce, and security patterns for detecting and mitigating these security issues. Resource variations may cause security issues unintentionally, but intentional introduction is also possible by malicious alternative resource suppliers, supply chain interdiction, and insiders. This article assumes malicious intent in resource variation as its security issue base line, as patterns for effective detection and mitigation of malicious variation intent encompass unintentional occurrences.

■ **KEYWORDS:** social contract; techno-social contract

CONTEXT

Product line engineering (PLE) can be fractal. Military radios produced as a product line can be assembled for specific features from an inventory of electronic circuit board components pooled for variations on general capabilities. One pool may have variations in sensor signal processing, another in transmission encryption. These radios may become part of an aircraft avionics system with an open systems architecture structured as a product line to accommodate different radio and other avionic devices. The avionics systems in turn may provide variations for use in an aircraft product family.

This article broadly addresses product components architected and designed as a product line part, and focuses on detection and mitigation of security issues introduced by component variation. Components have cyber, physical, and techno-social interactions with other components collectively configured as a product. Components adapt internally to facilitate variation in component features fit for different purposes. Variation may occur in cyber, physical, and techno-social features.

More specifically, this article focuses on PLE variation security from a techno-social pattern point of view. The techno-social viewpoint centers on the *social contract* concept among components. The social contract concept, introduced by the French philosopher Jean-Jacque Rousseau (Rousseau 1762), recognizes humans aggregate as communities for mutual preservation. A social contract is an implicit cultural agreement or contract among society members that "essentially binds the members into a community that exists for mutual preservation" (SparkNotes 2005).

We propose PLE variation security is more effective when there is a techno-social contract of mutual protection among product components, we discuss why this is, and we show ten patterns useful for social contract compliance.

TECHNO-SOCIAL CONTRACT CONCEPT

A product assembled from PLE component variants, built with intent to serve a specific user need (or desire), is a component collection which works to satisfy a user's total need. Ultimately, a component's task is to perform its intended functionality. A security issue in any other component it interacts with may affect its ability to function securely, but a security issue in any component may affect the ability of other components to function securely.

In a sense, we have a component community participating collectively to deliver total product satisfaction for the user. If the radio continues working in a disabled car the user experiences considerable dissatisfaction. This degrades the radio's participa-

tive intent. We propose the radio (and other components) should have a community sense specifically in the security domain, complying with a collective social contract for mutual protection.

WHY TECHNO-SOCIAL ASPECTS ARE USEFUL

Long considered truth, no unit or system testing, certification, and standards compliance can guarantee secure product operation. These practices are necessary, but insufficient. Accidental or insider-malicious security issues may occur undetected at engineering time, but damage manifests at operational time. Operational time is when unexpected emergent behaviors can occur.

An Original Equipment Manufacturer (OEM) initially assembles PLE products and, we suggest, provides a mutual protection social contract among OEM components. However, in deployed operation, 3rd party components may replace the OEM product components, or additional components may add to the product from 3rd party sources. Nevertheless, the OEM remains the producer of record, and bears the responsibility of product security failures. If a 3rd party component has a security issue not affecting other product components, the OEM is innocent, but if a 3rd party component acts as a gateway for security issues spreading to OEM components the OEM is at fault.

The 3rd party issue underscores the need for PLE product components to distrust other components. The OEM does not control the deployed product configuration.

A malicious OEM insider, the OEM's supply chain when procuring parts for product components from a malicious source, or the OEM ships a safe product component but interdiction and malicious alteration occurs in transit to the product integration point, or when installation or maintenance personnel maliciously introduce a component variation are various security issue introductions.

The malicious intent issue underscores the need for distributed real-time operational behavior assessment by OEM components.

There are many ways to introduce a vulnerable variation. Attempting to preclude such introductions before deploying a product is a form of perimeter defense. Implementing a product techno-social contract is a form of defense in depth.

For the reasons above, the operation must actively detect and mitigate variation vulnerability in a component-distributed manner sensitive to abnormal operational behavior. If the OEM product can detect and mitigate the uncontrollable 3rd party security issues and security issues introduced with malicious intent, then it encompasses unintended OEM engineering issues evading discovery.

TECHNO-SOCIAL PATTERNS

Peyton Quinn will provide a conceptual example. Peyton has a conscience, or so it seems. A voice saying you did something probably causing others some problems, and you ought to confess. Peyton resides in a gated community with a mutual protection social contract. A bit like neighborhood watch, but a lot more. The gate did not stop an intruder, evidenced by a mess made where a neighbor's keys were kept. Payton's conscience gets the upper hand and notifies the community association as well as the neighbors. The association responds shortly thereafter with a community broadcast saying a few residents have noticed security problems and recommends all go on high alert. Peyton double locks the doors, increases surveillance by cutting back on editing some videos as planned, and calls the cleaners to fix the mess the intruder made. Peyton Quinn is a blazing fast hardware/software techno-social device—pay a ton for edits, quintuple what software would have cost.

A social focus has patterns to consider from mutual security practices in human and animal social groups. The social focus in this article is technology-technology relationships rather than relationships involving humans or animals. Our concern is with components of a cohesive techno-social product community. The following ten patterns come from a paper discussing security in the Future of Systems Engineering (Dove, Willett 2020).

Self-Protection

When a techno-social contract is present there is an obligation for components to perform the contract, seemingly benefiting others but it is a contract for optimizing self-protection. Self protection is an encompassing macro-pattern including the nine following patterns and more.

Self Aware

Techno-social capabilities rely on self awareness, as socialness is a relationship between self and others. How much self awareness does a component need? At least awareness of the functional exchanges establishing interactive relationships with other components warranting attentive interest. Maximally, perhaps, as follows.

Self Behavior Judgement

This is like a conscience, an independent local agent evaluating behavior for expected norms and deviations constituting abnormality. This approach does not rely on other components' sustained integrity to make judgement, it distributes watchfulness diversely and widely and is independent of potentially aberrant functional mechanisms, regardless of cause. Such an agent might exist within the component or as a separate component-dedicated companion. See Horowitz 2015 for a functional example.

Self Behavior Mitigation

A self judgement may have different confidence levels. Some may be sufficient for unilateral immediate action. An extreme example proposed for ad hoc networks includes the ability for a component (node) to commit suicide for the greater good. Another component type might call for a wipe and reload. A less confident judgement may call for consensus among peer components or appeal to a higher authority, perhaps a component functioning as community overwatch attentive to multiparticipant appeals, or a human.

Peer Behavior Judgement

Peer-behavior monitoring and judgement occurs naturally and constantly in social animals. Each group member evaluates the others for adherence to social norms and threats to social coherence and security. Humans monitor others' behavior in more sophisticated and more complex ways than animals of lesser cognitive capability. A techno-social component interacts with other components through communication and observed behavior, can learn what to expect as normal, and vet for normalcy before, during, or after acting upon it. A two-part journal paper in Dove 2009a and 2009b reviews literature supporting concepts and methods for peer behavior monitoring among unmanned autonomous systems. "Trust but verify" might be a polite operable phrase but is fundamentally about the need for distrust.

Peer Behavior Mitigation

Rogue elephants are the result of banishment for unacceptable behavior. Social insects restrain and even kill group members that overstep certain social bounds. One of the 911 planes had passengers who took preventive action against the attackers. Nodes in some ad hoc networks will take a vote on questionable communication behaviors experienced with specific nodes and take collective action to refuse further interaction with a node receiving bad vote results.

Peer Collaboration

Vehicular communication systems are computer networks in which vehicles and roadside units are the communicating nodes, providing each other with

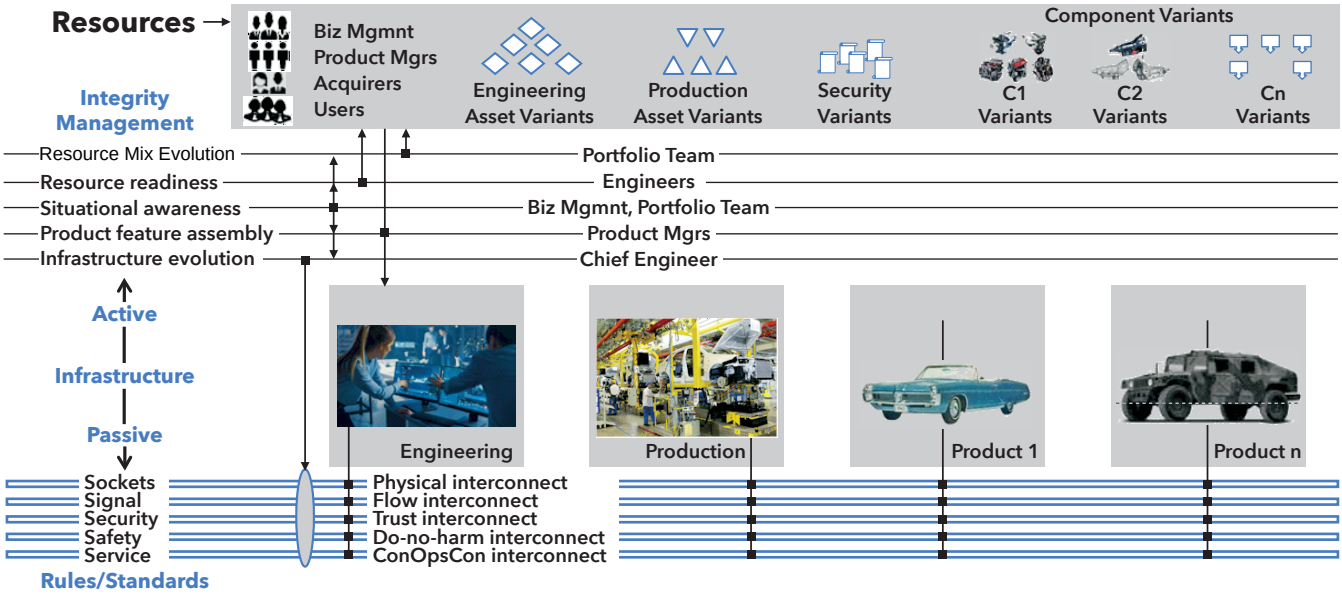

Figure 1. Notional Concept: Security-Relevant PLE-Process Agile Architecture Pattern

information, such as safety warnings and traffic information. They can effectively avoid accidents and traffic congestion. Both node types are dedicated, short-range communications devices. Vehicular communications usually develops as part of intelligent transportation systems (Wikipedia: "Vehicular communication systems").

Adaptable Attention Priorities

Maslow's [human] hierarchy of needs (Wikipedia) contends fuel and security are the first two of six, sustained existence needs taking precedence over higher level purpose needs. This occurs in robotic mobile devices interrupting their tasks to seek an electrical outlet, and in devices and operating systems with various anti-tamper detection and prevention capabilities (short of self-destruction). For a notional technical hierarchy of needs see Dove and Willett (2020).

Diversity

There is a socially attentive load on components attempting to cover a large awareness area, and inefficiency in duplicating their neighbors' same measures. All components do not have to participate, and all components should not look for the same things. One way this could implement might be to have a selection of (work intense) things to do randomly down selected by or for each component. Gal Kaminka makes this case in his doctoral thesis (Kaminka 2000) for distributed social behavior monitoring and detection, showing a centralized monitor does not do as well as multiple monitor/detectors among socially aware components. He also shows this can happen effectively without any one component monitoring all components, and without all the components having this monitoring capability.

Heterogeneous Awareness

A recent study of grey squirrels (Lilly, Lucore, and Tarvin 2019) found they use signals from multiple bird species to indicate a present threat is in the area, as well as to indicate no imminent threat is present. Normal calm bird chatter finds the squirrels attending to foraging tasks, while alarm notes cause heightened agitation and evasive moves. Technical components receiving signals about the general state of alarm or calm in other components not in direct peer communication can ratchet the relative component attention level between self protective activity and functional purpose. Heterogeneity differs from diversity in that different social sub-groups have some cross communication, whereas diversity addresses a single social subgroup.

ARCHITECTURAL VIEW

Engineered as systems, PLE components and component variations should apply good system security practices during their engineering activity, as with any system type. But a PLE product assembled from components and their variations provides an opportunity for security not readily available in a non-PLE product. This is true because of the PLE product architecture and the PLE process architecture. Both are classic agile architecture pattern forms (Dove and Schindel 2019), structured to facilitate reusable, reconfigurable, and scalable product and process configurations.

A PLE product has a standardized infrastructure facilitating interconnection among components and their variations to configure a product. A PLE process has a similar standardized infrastructure facilitating interconnection among engineering assets, production assets, and component assets, as depicted in Figure 1.

Techno-social security assets are a principle resource pool in the PLE process architecture. It is a pool of social contract variations because product intended for use by different customer types may need different capabilities. Figure 1 depicts a military vehicle and a drive-around-town vehicle as two possible product types in a product family. Military acquirers and users will likely want more security capability and features. The drive-around-town vehicle may have variations appropriate for evolving driverless operation or vehicle communication systems.

A techno-social contract for a PLE product family has three aspects for consideration: security assets associated with specific products, security assets associated with specific components, and system engineering assets associated with the PLE process. A product family employing a techno-social contract concept will likely have variants in all three asset classes.

Product security assets implement the techno-social contract at the product level of component-community interaction. Peyton Quinn's story referenced a community association providing security-related services to all community members. A product security asset may also phone home to the OEM with security information indicating issues needing attention in similar products.

Component security assets implement the techno-social contract at the component level. Some may also have direct communication capability with the OEM—to receive security updates and wipe-and-reloads, and to transmit component-specific security issues.

Systems engineering security assets enable assembling a techno-social contract for a product at two levels: resources include various techno-social contract assets which can draw upon various component and product security assets appropriate for a given techno-social contract. In human societies a social contract can be cultural, lawful, or both. A techno-social contract will generally rely upon a lawful approach governed by the contract's nature—short of artificial intelligence approaches beyond this article's scope. An OEM may decide contract governance includes contract enforcement, depending upon tolerance for 3rd party component inclusion.

CONCLUDING REMARKS

We cannot guarantee security. There is nothing absolutely preventing the possibility a PLE variation introduces an exploitable security issue. Good and improved security practices in the PLE factory management processes will surely help; but it is insufficient to believe good security practice during PLE factory operation will ensure a secure product. This argues for security vigilance during operational product behavior.

PLE invites 3rd party suppliers. With a standardized component interface, the OEM cannot control replacing components in an operational product or adding components for additional capability. The automotive after market is a prime example. Defense acquisition's push to open systems architecture intended to enable componentry from other than the OEM. In any event, operational product augmentation or replacing OEM componentry can cause security issues. Measures countering security issues in the 3rd party operational environment can inform security practices in the OEM variant engineering activity.

A techno-social contract can provide emergent security behavior adapting to a varying threat. While instantiating techno-social contracts will not address all threat variations, it addresses more than the static safeguard predecessors. The result has the potential to act in a non-deterministic manner.

This article introduced notional concepts and patterns for a behavior-based techno-social contract among components in an operational PLE product. It also suggested three areas needing consideration for enabling, designing, and implementing a techno-social contract. Fundamentally the concepts and areas requiring work apply to agile security in general and warrants more attention for security in the Future of Systems Engineering. ■

REFERENCES

- Dove, R. 2009a. "Paths for Peer Behavior Monitoring Among Unmanned Autonomous Systems." *ITEA Journal 2009* 30: 401–408. www.parshift.com/s/090901IteaJ-PathsForPeerBehaviorMonitoringAmongUAS.pdf.
- ——. 2009b. "Methods for Peer Behavior Monitoring Among Unmanned Autonomous Systems." *ITEA Journal 2009* 30: 504-512. www.parshift.com/s/091201IteaJ-MethodsForPeerBehaviorMonitoringAmongUas.pdf.
- Dove, R., and B. Schindel. 2019. "Agile Systems Engineering Life Cycle Model for Mixed Discipline Engineering." Paper presented at the 29th Annual International Symposium of INCOSE, Orlando, US-FL, 20-25 July. www.parshift.com/s/ASELCM-05Findings.pdf.
- Dove, R., and K.D. Willett. 2020. "Techno-Social Contracts for Security Orchestration in the Future of Systems Engineering." Proceedings 30th Annual International Symposium of INCOSE, virtual event, 18-23 July. www.parshift.com/s/200718IS20-FuSETechnoSocialContracts.pdf.
- Horowitz, B. (Principal Investigator). 2015. "Four part System Aware Cyber-Security Project report." Systems Engineering Research Center. Report No. SERC-2015-TR-036-4. https://apps.dtic.mil/dtic/tr/fulltext/u2/a626823.pdf.
- Kaminka, G.A. 2000. "Execution monitoring in multiagent environments." Ph.D. diss., University of Southern California (Los Angeles US-CA).
- Lilly, M.V., E.C. Lucore, and K.A. Tarvin. 2019. "Eavesdropping Grey Squirrels Infer Safety From Bird Chatter." PLOS ONE, 4 September. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0221279.
- Rousseau, J-J. 1762. *On the Social Contract*. Translated by Maurice Cranston. New York, US-NY: Penguin Publishing Group.
- SparkNotes. 2005. "The Social Contract." www.sparknotes.com/philosophy/socialcontract/characters.

ABOUT THE AUTHOR

Rick Dove is Paradigm Shift International's CEO, specializing in agile systems engineering and security research, engineering, and project management, and an adjunct professor at Stevens Institute of Technology teaching graduate courses in agile and self-organizing systems. He chairs the INCOSE working groups for Agile Systems and Systems Engineering, and for Systems Security Engineering. He is an INCOSE Fellow, and author of *Response Ability, the Language, Structure, and Culture of the Agile Enterprise*.