Bend, Don't Break: Using Reconfiguration to Achieve Survivability

Alexander L. Wolf[†] Dennis Heimbigner[†] John Knight[‡]
Premkumar Devanbu[⋆] Michael Gertz[⋆] Antonio Carzaniga[†]

[†]Dept. of Computer Science
University of Colorado
Boulder, CO 80309-0430 USA
{alw,dennis,carzanig}@cs.colorado.edu

[‡]Dept. of Computer Science
University of Virginia
Charlottesville, VA 22904-4740 USA
knight@cs.virginia.edu

[⋆]Dept. of Computer Science
University of California
Davis, CA 95616-8562 USA
{devanbu,gertz}@cs.ucdavis.edu

Our national interests are becoming increasingly dependent on the continuous, proper functioning of large-scale, heterogeneous, and decentralized computing enterprises. Examples of such systems abound, ranging from military command and control to vital national security assets such as the financial and banking system. They are formed from large numbers of components originating from multiple sources, some trusted and some not, assembled into complex and dynamically evolving structures. Protecting these interests is critical, yet their sheer scale and diversity has gone far beyond our organizational and technical abilities to protect them. Manual procedures—however well designed and tested—cannot keep pace with the dynamicity of the environment and cannot react to security breaches in a timely and coordinated fashion, especially in the context of a networked enterprise.

We are designing a secure, automated framework for proactive and reactive reconfiguration of large-scale, heterogeneous, distributed systems so that critical networked computing enterprises can tolerate intrusions and continue to provide an acceptable level of service. Proactive reconfiguration adds, removes, and replaces components and interconnections to cause a system to assume postures that achieve enterprise-wide intrusion tolerance goals, such as increased resilience to specific kinds of attacks or increased preparedness for recovery from specific kinds of failures. Proactive reconfiguration can also cause a relaxation of tolerance procedures once a threat has passed, in order to reduce costs, increase system performance, or even restore previously excised data and functionality. In a complementary fashion, reactive reconfiguration adds, removes, and replaces components and interconnections to restore the integrity of a system in bounded time once an intrusion has been detected and the system is known or suspected to have been compromised. Recovery strategies made possible by reactive reconfiguration include restoring the system to some previously consistent state, adapting the system to some alternative non-compromised configuration, or gracefully shedding non-trustworthy data and functionality. In our view, proactive and reactive reconfiguration are two sides of the same coin that can be profitably unified into a coherent and comprehensive survivability mechanism.

Our framework, which we refer to as BdB ("bend, don't break"), consists of two interrelated elements: an *application architecture* for designing large-scale, dynamically reconfigurable systems and a *common infrastructure* for building and securely operating those systems. The architecture embodies essential principles for effectively integrating active control mechanisms with components, thus permitting application developers to concentrate on the specialized aspects of their systems. A coordinated set of models, a standard component reconfiguration interface, and an agent-based policy mechanism together form the innovative core of the architecture. The infrastructure element of BdB provides several common components for inclusion into implementations adhering to the architecture, as well as several development tools for generating, optimizing, and verifying distributed reconfiguration control code. Key to the utility of our approach is the use of a novel high-level language for specifying intrusion tolerance policies that are then translated into the control code. Also used in realizing control is an advanced software configuration and deployment system that is designed to operate in a wide-area, large-scale, and heterogeneous enterprise environment. The infrastructure itself is secured using a new technique we refer to as view-based delegation of trust.

Our work on BdB is a novel blend of results from the disciplines of fault tolerance, configuration management, and security, building on a foundation laid by our previous research efforts.

- The University of Virginia is developing architectural techniques that are designed to enhance the survivability of critical information networks through secure application fault tolerance [5]. The faults of interest are non-local, meaning that they affect multiple network nodes and cannot be masked by traditional redundancy mechanisms. Approaches are being developed for the formal specification both of required application-level error detection and error recovery. Implementation is by direct synthesis from the formal specification so as to cope with the size of the required implementation and the need for rapid update. Secure operation of these mechanisms is essential, yet part of the trusted survivability mechanism has to operate on untrustworthy hosts. This problem is being addressed by developing means of preventing a variety of attacks on trusted binary programs.

- The University of Colorado is developing a distributed, mobile agent-based framework for supporting software configuration and deployment in a wide-area setting [2, 4]. The framework focuses on deployable sets of software artifacts, either software or data. Mobile agents are used to define generic software deployment processes such as install, activate, update, adapt, and remove. These generic processes are parameterized by declarative information in the form of a standardized software description schema, which provides content, constraint, and dependency information about the software systems being deployed [3]. The mobile agents combine the semantic information with their generic process definition to perform a specific configuration and deployment task at a given field site. The result is an instantiated software system configuration at the field site that is annotated with the semantic information provided in the software description schema. The annotated semantic information can then be used at the field site for software system maintenance.

- The University of California is developing a secure mediation infrastructure (SMI) for controlled access to data resources. There are two design goals for the SMI. First, administrators can precisely control and coordinate where and how data are obtained. Second, the SMI allows authentic data distribution, without the use of on-line signing keys [1]. Keyless hashing primitives are used to validate the distributed data, with the data signed infrequently using off-line keys. By avoiding on-line keys, which might be stolen by an attacker, we enhance security and reduce administrative burden.

In developing the BdB framework, we make several important assumptions about the environment into which it will be situated. First, we must explicitly accommodate COTS components, which means that we cannot impose unrealistic expectations on their functional support of reconfiguration activities. Therefore, we assume only that a component is able to perform a commanded checkpoint of its state and a commanded restart from that state. If it is unable to do so, then we assume we can fall back to a commanded deactivation and (re)activation from an initial state. Second, we assume that we can use an available wrapper technology to add our standardized reconfiguration interface to components so that all components can be treated in a uniform manner by the infrastructure facilities. Third, attending to component-local, maskable faults is not within the purview of this project. Our focus is on non-maskable faults that have implications for continued operation of the system as a whole. Fourth, we assume the use of secure communication channels using known techniques developed elsewhere. While these assumptions have pervasive effects on the design of our framework, they are all reasonable assumptions that do little to limit the applicability of our results to a wide range of critical networked computing enterprises.

References

[1] P. Devanbu, M. Gertz, C. Martel, and S. Stubblebine. Authentic Third-Party Data Publication. In *Proceedings of the Fourteenth IFIP Working Conference on Database Security*, August 2000. To appear.

[2] R.S. Hall, D.M. Heimbigner, A. van der Hoek, and A.L. Wolf. An Architecture for Post-Development Configuration Management in a Wide-Area Network. In *Proceedings of the 1997 International Conference on Distributed Computing Systems*, pages 269–278. IEEE Computer Society, May 1997.

[3] R.S. Hall, D.M. Heimbigner, and A.L. Wolf. Evaluating Software Deployment Languages and Schema. In *Proceedings of the 1998 International Conference on Software Maintenance*, pages 177–185. IEEE Computer Society, November 1998.

[4] R.S. Hall, D.M. Heimbigner, and A.L. Wolf. A Cooperative Approach to Support Software Deployment Using the Software Dock. In *Proceedings of the 1999 International Conference on Software Engineering*, pages 174–183. Association for Computer Machinery, May 1999.

[5] J.C. Knight, K. Sullivan, M.C. Elder, and C. Wang. Survivability Architectures: Issues and Approaches. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, pages 157–171, Los Alamitos, California, January 2000. IEEE Computer Society Press.