Diversity As a Computer Defense Mechanism
A Panel

Carol Taylor
University of Idaho
Computer Science Depart.
Moscow, ID 83844-1010
208-885-0680
ctaylor@cs.uidaho.edu

Jim Alves-Foss
University of Idaho
Computer Science Depart.
Moscow, ID 83844-1010
208-885-5196
jimaf@cs.uidaho.edu

ABSTRACT
This panel addressed the use of computer diversity as a strategy for computer security. It is our view that there are significant knowledge gaps in the science underlying diversity as a computer defense mechanism which hinders its usefulness. These gaps include the true cost of diversity, a lack of metrics for diversity and the trade offs between diversity and other defensive strategies. We also wanted to highlight on-going diversity research from other disciplines which could potentially be applied to diversity for computer security.

Four panelists were selected based on their experience with diversity within the context of computer security or other research areas. The panelists' backgrounds include biology, software reliability, security and dependable systems. Each panelist presented a statement which was discussed by NSPW participants.

The discussion was lively and informative both during and after the panelists' statements and is reported in a later section of this summary.

1. INTRODUCTION
Our panel, *Diversity as a Computer Defense Mechanism,* was presented at NSPW. The panelists were chosen for their background experience or knowledge of diversity within the context of computer security.

The goal for the panel was to encourage discussion and debate by NSPW participants in order to better define computer diversity as a research area. Current research into the defensive role of computer diversity is not mature and includes many unanswered questions. A secondary panel goal was to foster interest in diversity research which might help answer some of the open questions surrounding computer based diversity for security.

2. DIVERSITY AS A TOPIC
While diversity for computer defense is not a new paradigm it is a paradigm in need of further definition. Currently, not enough is known about diversity to make it useful for computer security and yet it continues to surface as a proposed solution. There is no quantitative information on the costs associated with implementing diversity. Thus diversity may be prohibitively expensive compared to other security strategies. Another unknown is the strength of protection offered by diversity. The typical way diversity is used in computer security is to generate either a code or system obfuscation in order to increase attacker effort. Yet, the quantity of effort needed to overcome diversity is undefined. The general relationship between diversity and typical attacks has not been determined for even the average case so the amount of diversity required to thwart a specific attack is unknown.

Diversity is also of interest because it has recently been the focus of a controversy involving Microsoft as a national computer security risk. A 2003 Computer & Communications Industry Association (CCIA) report asserted that the US is at risk from computer insecurities because of the overwhelming dominance of Microsoft Windows as the Operating System (OS) of choice [8]. This report was the subject of a lively debate at the 2004 Usenix Security Conference between Scott Charney of Microsoft and Dan Geer, one of the report's authors[1] [13]. A common misconception held by the security community is that more diversity is always better which is not necessarily true.

3. BACKGROUND

3.1 Diversity for Greater Reliability
Software that operates in safety critical applications must be highly reliable in order to avoid catastrophic consequences such as loss of lives or huge financial loss. Yet, how do you improve software reliability knowing it is nearly impossible to eliminate all faults that could potentially cause system failure?

The fault tolerant community addresses this problem through redundancy, running several identical components and by diversity, using a number of different components. Voting is then typically done to determine differences between components which could signify component failure. For hardware, failure is typically caused by random faults so duplicating components provides added insurance since the assumption is that failures are independent. However, software failures are generally due to design faults created by developers. Consequently, faults are embedded within the software and every copy of that software

[1] There was no clear winner of that debate!

NSPW 2005 Lake Arrowhead CA USA

behaves identically with respect to a given input. In this case, having multiple copies of the software doesn't help with reliability since each one will fail identically.

In an effort to increase the failure independence between software versions, N-version programming was proposed back in the 70's [1] as a technique for increasing overall reliability. Diversity is introduced by having different programming teams produce versions of the same program. It was hypothesized that diversifying the software producers should result in programs with the same functionality but not the same faults. Research on the outcomes of N-version programming showed that the assumption of total independence of failures was false [10]. Programmers tended to make similar mistakes over more difficult parts of a problem. These experiments [10], plus others reported in [12], led to doubts as to the actual benefits of N-version software diversity. However, other researchers suggested that failure independence did not provide the complete picture of software reliability [11, 12] using multi-version programs.

Research conducted by Littlewood and Miller [11] who studied forced[2] diversity in N-version programming, built a model of the probability of version failure over sub-domains of the program space. They showed that problems difficult for one sub-domain might be easy for a different method and averaging the results could prove favorable. A summary of their results on software diversity includes the following findings:

- Benefits of software design diversity are difficult to measure
- Software diversity has been used in real safety critical applications
- Diversity seems to help with reliability, but there is not enough data to say that diversity absolutely helped with overall reliability
- The same level of reliability might have been achieved by some other means

Even after many years of study, there are unanswered questions relating to design diversity. Yet, it is likely that knowledge gleaned about failure mechanisms of diverse software for reliability could possibly be applied to the area of computer security.

3.2 Diversity for Biology

Diversity in the biological world appears to function by maintaining species and ecosystems. Within a species, diversity is credited with assisting species survival by varying the genetic make-up since all members won't be equally susceptible to environmental threats such as predators or disease. At the ecosystem level, higher species diversity is correlated with ecosystem stability. The main idea is that if there are enough species, a high diversity state, substitution can occur among functionally equivalent species in cases of species extinction.

This is the primary idea behind the often cited statement that monocultures are less stable since one disease event could potentially wipe out an ecosystem of one species if the one species is susceptible to the disease.

As noted previously the benefits of diversity may be overstated when applied to computer security. Clarification of diversity concepts plus a way to measure diversity would benefit researchers that want to use these ideas to solve computer based security problems.

3.3 Diversity for Computer Security

Diversity has been studied as a technique for increasing system security. Research results are highlighted from computer security to provide an overview of the current state of computer diversity research. While the goal of diversity in fault tolerance is to promote failure independence between program versions, the purpose of diversity in computer security is to increase the attacker's effort to compromise a system. Independence of failure is assumed but not measured in most security diversity research.

Past research examined the feasibility of obscuring programs and OS components plus examined architecture levels where diversity could help defend computers from attacks. Forrest, Somayaji and Ackley investigated potential sources of diversity within the operating system [7]. Their research discussed ways to introduce obfuscation such as changing program memory layout, reordering code, adding padding to stack frames and changing names of important system files [7]. Cowan et al [5] evaluated and compared restrictiveness techniques - methods that restrict certain behavior, with obscuring strategies - techniques that hide some system aspect from would-be attackers. The study discussed the relative merits of the two approaches and found that in most cases obscuring techniques are more difficult to implement plus introduce greater complexity which is less likely to be implemented correctly. Another study examined the effectiveness of diversity in stopping several well known attacks [2]. The authors point out that in theory diversity could have helped resist penetration for some attacks. But they note that diversity for most OS and network protocols is limited and likely would not have presented much difficulty for the attackers. Another study by Deswarte [6] approached diversity for security from a fault tolerant view. He described various fault classes that could affect systems and noted where diversity could assist with masking each fault. Deswarte considered diversity at five levels including, operator, user interface, OS level, N-version and execution level and believed that diversity can help ameliorate both design and intrusion (malicious) faults [6].

Current diversity research looks at very specific vulnerabilities that can be defended against through obfuscation. Instruction set diversity was examined in two separate studies [3, 9]. One used a binary translation technique [3] while the other performed a kernel modification to achieve instruction set variance [9]. Both techniques were guarding against code injection attacks. Neither technique is affective against all code injection attacks. Another study involved using randomization for address obfuscation [4]. The authors tried a number of ways to randomize the location of code and data. Their address randomization methodology covers some but not all memory exploits. The assumption behind all three studies is that the obfuscation will result in greater effort by the attacker to compromise each randomized version.

[2] Forced diversity is where diversity is deliberately introduced by requiring different languages, tools, testing suites or some other required differences between development teams

4. PROPOSED PANELISTS

The proposed panelists along with their background qualifications include:

- **Bev Littlewood** is a Professor of Software Engineering at City University London. Bev has worked for many years on problems associated with the modeling and evaluation of dependability of software-based systems. He is a member of the UK Nuclear Safety Advisory Committee, of IFIP Working Group 10.4 on Reliable Computing and Fault Tolerance, and of the BCS Safety-Critical Systems Task Force. He is a Fellow of the Royal Statistical Society. In particular, Dr. Littlewood has studied probability associated with N-Version program design. He has developed probability models related to forced N-version programs where diversity is deliberately induced between the various programming teams. More recently, Dr. Littlewood has studied diversity for computer security. Dr. Littlewood's knowledge of the use of diversity to increase reliability for fault tolerance is extensive and his interest in applying that knowledge to computer security makes him an ideal panelist.

- **Roy Maxion** is a professor in computer science at Carnegie Mellon University. Roy Maxion's research covers several areas of computer science, including development and evaluation of highly reliable systems, machine-based concept learning, and human-computer interfaces. He is developing dependable systems for automated detection, diagnosis and remediation of faulty or unanticipated events in many domains -- international banking, telecommunications networks, vendor help systems, semiconductor fabrication, information warfare and others.

- **John McHugh** is a senior member of the technical staff at the CERT® Coordination Center, part of the Software Engineering Institute (SEI) at Carnegie Mellon University where he does research in survivability, network security, and intrusion detection. Prior to joining CERT®, Dr. McHugh was a professor and chairman of the Computer Science Department at Portland State University in Portland, Oregon where he held a Tektronix Professorship. He has been a member of the research faculty at the University of North Carolina and has taught at UNC and at Duke University. For a number of years, Dr. McHugh was a Vice President of Computational Logic, Inc. (CLI), a contract research company formed to further the application of formal methods of software design and analysis in support of security and safety critical systems.

- **Carol Taylor** recently finished her Ph.D. in Computer Science in May, 2004 and continues to work as a Post Doctorate Fellow at the University of Idaho. She has several degrees in biology in addition to her CS background and has worked as an ecologist in a previous position. Dr. Taylor has a strong interest in the application of biology to computer security including diversity. She has studied the way diversity is currently applied in much of the security research and believes that stronger, more usable results could be achieved with better quantification and more experimentation. A discussion about the limitations of biological diversity for computer security would hopefully produce more realistic, usable results.

5. THE PANEL

Each panelist presented his or her statement which prompted discussion by NSPW participants who then commented on the panelists' views. The major points made during the panel are listed below:

Cost of Diversity and Design

Carla Marceau believes the cost of redundancy is much higher than just the cost of generating n versions. For example, consider supporting multiple browsers. There is training, interoperability, testing and other costs.

Ken O. asks if it would be more efficient to begin with one anomaly detector and then design 2 ... n more detectors specifically to cover the blind spots of the first detector.

Roy Maxion answered that the approach suggested by Ken is exactly the right approach plus you would try to define an attack taxonomy from the viewpoint of defenders and use that to design detectors.

Diversity Measurement and Propability

Bev Littlewood observes that a good deal of diversity work is anecdotal - desires a metric and states that probability would be useful in measuring diversity.

Bob Blakeley comments that if an attacker is stochastic, it might not be possible to model the attacker since the probability distributions will be too difficult to model

Steve Greenwald points out that metrics for security are one of the fields' acknowledged grand challenges

John McDermott believes we don't have enough data to form a good statistical or actuarial database. Also many security problems are rare (e.g. attacks at the nation state level)

Bev Littlewood states that the same claim was made 30 years ago about software reliability.

Roy Maxion thinks we can help drive the creation of metrics by rejecting papers which have only anecdotal data.

Bob Blakeley refers to Lampson's result on Byzantine faults and claims it demonstrates that statistical models cannot describe intelligent, malicious behavior.

Brian Snow asks how can we predict what an attacker is about to do - the problem is we are dealing with malice.

Bev Littlewood replies that the malice argument is irrelevant to statistical analysis of a system.

6. DIVERSITY PANEL ANALYSIS

The panel was lively, informative and fulfilled one of the panel goals: to encourage debate on diversity for computer security in order to help define it as a legitimate research topic. Success for the sub-goal of increasing research activity surrounding diversity can only be determined later by the interest generated from the security research community.

During the panel, other topics emerged that are relevant to much of computer security research and warrant further discussion. These topics include:

- The role of Probability for Computer Security
- The continuing lack of Security Metrics

6.1 Probability for Computer Security

Much of the panel discussion revolved around the usefulness of probability for modeling diversity. Probability was proposed as one way to measure diversity in a computer security context. Bev Littlewood's view is that probability can add rigor to a system that uses diversity as a security mechanism such as multiple Intrusion Detection Systems (IDS's). In this case, one could measure the probability of both systems failing under a given attack plus estimate the benefit of having multiple systems through the covariance term of the probability model.

However, there is the general belief in the security community that because attacks are intentional as opposed to random acts of nature, probability is not useful. For example, say an unknown software vulnerability will likely only be found and exploited by a small group of motivated individuals and almost never be discovered through normal use. Yet, once a vulnerability is known and an exploit developed, the probability it will be exploited becomes one with only the time-to-compromise being variable. However, if we only consider the initial discovery and removal of vulnerabilities (which is a stochastic process) and not the subsequent exploitation of these vulnerabilities, then the problem is similar to a software reliability growth model and probability can play a useful role.

The questions about where and when probability can assist in modeling security go beyond diversity and apply to most areas of computer security. The panel discussion highlighted the need for more research in order to determine how probability can aid in security estimation and measurement.

6.2 Security Metrics

The lack of security metrics was mentioned during the panel discussion as an on-going challenge for the security community. Yet, no suggestions were made for measuring diversity other than through probability as discussed in the previous section. Multiple diversity metrics exist in the biological world, which take into account species composition or distribution for a given area. While computer-based diversity can't be compared with its biological counterpart, these diversity measures might suggest ways to quantify diversity for computer security. This remains an area in need of further research for diversity, and in general, computer security.

7. ACKNOWLEDGEMENTS

We would like to thank all the NSPW participants for their comments and willingness to share their knowledge with the panelists and each other. In typical NSPW form, the discussion helped frame the issues and highlighted new areas of future investigation.

8. REFERENCES

[1] Avizienis, A. and L. Chen. On the implementation of N-version programming for software fault tolerance during execution. In *Proc. First IEEE-CS Int. Computer Software and Applications Conf. (COMPSAC 77)*, Chicago, Nov., 1977.

[2] Bain, C., D. Faatz, A. Fayad, D. Williams. Diversity as a defense strategy for information systems. Mitre Corp., 2000.

[3] Barrantes, E. G. et al. Random instruction set emulation to disrupt binary code injection attacks. In *Proc. CCS'03, Oct. 27-31, Wash., D.C.*, 2003.

[4] Bhatkar, S., D.C. DuVarney and R. Sekar. Address obfuscation: an efficient approach to combat a broad range of memory error exploits. In *Proc. 12th Usenix Sec. Symp., Wash., D.C.*, 2003.

[5] Cowan, C., H. Hinton, C. Pu, J. Walpole. Cracker patch choice: an analysis of post hoc security techniques, 23rd NISSC, Baltimore, MD, 2000.

[6] Deswarte, Y., K. Kanoun, J. Laprie. Diversity against accidental and deliberate faults. In **Computer Security, Dependability and Assurance: From Needs to Solutions,** 1998, York, England.

[7] Forrest, S., A. Somayaji and D.H. Ackley. Building diverse computer systems. In *Proc. 6th Workshop Hot Topics in Operating Systems*, 1997, pp. 67-72.

[8] Geer, D. et al. Cyber Insecurity: the cost of monopoly, how the dominance of Microsoft products poses a risk to security. Computer & Communications Industry Association Report, 2003.

[9] Kc, G. S., A. D. Keromytis, V. Prevelakis. Countering code-injection attacks with instruction set randomization. *CCS'03, Wash., D.C.*, 2003.

[10] Knight, J. and N.G. Leveson. An experimental evaluation of the assumption of independence in multi-version programming. *IEEE Trans. Soft. Eng., SE-12, 1.*

[11] Littlewood, B. and D. R. Miller. Conceptual model of coincident failures in multi-version software. *IEEE Trans. Soft. Eng. SE-15,12.* 1989.

[12] Littlewood, B., P. Popov, and L. Strigini. Modeling software design diversity – A Review. *ACM Computing Surveys, Vol. 33, No. 2*, June 2001.

[13] Usenix Security Symposium, www.usenix.org, 2004.