

The Super-User, an Anachronism?

John Giacomoni

Department of Computer Science
University of Colorado
Boulder, CO 80309-0430, USA

Abstract—Conventional wisdom tells us that an all powerful super-user is required for effective administration of a computer system. This wisdom, called ‘best practice’ by system administrators or ‘due diligence’ by managers, often comes with years of positive reinforcement that effectively stifles the conceptualization or adoption of alternatives. In this note, I present an overview of the conflicting technical, psychological, and economic aspects the dependence upon a super-user account/role presents to a system and its dependent users. I finish by briefly outlining a potential partial solution for managing the risks associated with super-user accounts.

Keywords—Super-User, Security, Economics, Psychology

I. INTRODUCTION

Computer systems have evolved from their origins as assistive tools to their current place at the foundation of modern business practices. Central to any computer system is the brokering of resources (information) to its users. Traditional research into computer systems security has focused on the techniques and mechanisms used to design a site-specific security environment. However, as the media points out on an alarmingly frequent basis, these traditional approaches are not serving us well.

Traditional approaches appear to be nothing more than the symptomatic application of band-aids. Ross Anderson recently described how perverse economic incentives work against improved security systems and policies [1]. I argue that there are certain underlying psychological issues in addition to the technical and economic ones. In particular, I question the continued dependence on a super-user account/role in the face of these issues regardless of its evaluated certification level.

II. PSYCHOLOGICAL ASPECTS

Interestingly, with the notable exception of computer interface researchers, the psychology of users appears to be undervalued by computer scientists, especially concerning security. The human element is a definite thorn in the side of system designers, due to its unpredictable nature.

Historically, a computer had only a single user, the operator, who had complete authority, and more importantly responsibility, for its correct operation. A dangerous precedent was established when systems became interactive, multi-tasking, and multi-user, as two classes of users were created: operators (super-users) and normal users. This division of duties resulted in an implicit caste system. System administrators reinforce this by ‘helpfully’ fixing all problems, including trivial user problems. Experience

has taught me that while the best administrators care for their systems as much as for their families, they are fallible humans like everyone else.

Interestingly, as in many socio-economic caste systems, the users reinforce this class division. Admittedly it is convenient to have one’s super-user fix all one’s problems. Unfortunately, as humans are “social-animals”, we willingly hand over unjustified levels of trust to those who are helpful and friendly (pp. 78–82 [2]). There is no reason why a super-user needs the ability to read private corporate documents, such as those belonging to accounting, to properly perform their duties.

Any solution expected to have a measurable impact must factor in this caste system as all users of a system continually reinforce it.

III. ECONOMIC ASPECTS

Perhaps the most perverse and conflicting of the aspects are the economic ones. Ross Anderson has given us a long overdue outline of the economic conflicts involved in real world distributed systems [1].

While economics can help us explain group behavior and quantify risks, we must realize that economic defense is limited to cases where financial reparations are possible. In many cases, however, a single incident can cause irreparable harm to the users of a system. Group entities (companies, etc.) are often most concerned with the premature disclosure of intellectual property, which could result in the dissolution of the entity or worse. Consider a hypothetical pharmaceutical company whose systems contain the formula of a new ‘drug’ which drug lords would be most interested in selling. Should the ‘drug’ formula be disclosed, irreparable harm would occur. Long-term secrecy applies as much to group entities as to government agencies [3].

Experience shows that many system administrators are effectively mercenaries hired into at-will relationships, looking for the most pay with the least risk. Super-user accounts controlled by mercenaries pose a potentially serious long-term risk to any entity that hires them. This situation has arisen in part due to ‘incentive failures’ (pay, punishments). Disenfranchisement is the other major contributing factor; administrators are held accountable, but rarely granted the authority, or tools, to properly perform their duties.

Current trends suggest users outsource their information technology, and in particular the security aspects, as the way to stay most current and competitive [4]. One must

realize that in doing so trust relationships with the super-users are weakened. Though the ability to shed liability in this manner could be very tempting to certain users.

Finally, there is one more economic issue that needs addressing: the capabilities of a small entity versus a large one. Small entities usually cannot acquire resources equivalent to large ones while, in many ways, having similar levels of exposure. If we consider “a trusted component or system [to be] one which [one] can insure” [5], all small entities are liabilities.

IV. TECHNICAL ASPECTS

Traditional approaches have focused on the purely technical aspects, which are fairly well understood so I will only briefly describe them here. Generally there are three different classes of technical flaws: Implementation, Design, and Emergent Properties. Many approaches have been devised to help minimize the three classes of flaws, all designed to meet the requirements of differing environments. These approaches include formal logics, language support, coding practices, discovery/patch cycles, access control mechanisms, and formal evaluation [6], [7]. Unfortunately these techniques appear to be insufficient, as the same classes of vulnerabilities repeatedly manifest themselves in ‘secure’ programs such as OpenSSH.

Observing the various security bulletins that are generated each year, the majority of the information critical vulnerabilities all involve exploiting privileged processes. Privileged processes are the obvious choice for any attacker, as their *raison d’être* is to do what a normal process cannot. In other words, violate security mechanisms and policies.

It appears that the best traditional approaches can realistically promise is a weak form of ‘containment’ with minimal assurances coupled with significantly increased maintenance issues.

V. THOUGHTS

Security is increasingly being recognized as a critical requirement for any remotely accessible system. Human nature combined with Moore’s Law allows us to safely predict the continued functional augmentation of systems. Over time this trend appears to exhibit a strong correlation with the complexity of the user experience and underlying systems. The direct result is an exacerbated separation between those who have the knowledge or money required to secure systems, and those who have not.

Traditional approaches have failed in large part due to their complexity in attempting to solve the ‘security problem.’ The cost, in terms of hours of labor and currency [8], involved in creating a traditional trusted operating system as specified by TCSEC [9], or related systems, is directly passed on to the consumer, resulting in a poor return on investment. The system complexity directly manifests itself to the users providing a greatly diminished experience and a steep learning curve. While such systems may prove, in the future, to be practical, our need is immediate.

Generally available systems are in dire need of simple protection mechanisms that small and large entities can

use with minimal effort and have a measurable return on investment. Without the adoption of improved systems it will be difficult to effectively manage the risks to a level where an insurance carrier is willing to extend a reasonable premium for coverage.

Potential solutions appear to hinge on the flaws inherent in the super-user account. Ideally one would remove the dependence on the super-user account from operating systems. This would serve to remove the most consistently attacked component of modern multi-user operating systems. However the briefly reviewed traditional approaches combined with the reviewed psychological and economic aspects show us that such an approach is destined for non-adoption.

The next best approach is to protect the users from the super-user. Thus honoring the psychological status quo for the users, improving the general security, and potential trust relationships with organizations providing security services.

VI. POTENTIAL SOLUTION

A complete solution will require many subtle and pervasive changes that are beyond the scope of this note. Abstractly there are two core requirements to ensure user protection: the super-user needs to honor discretionary access control specifications; user accounts need to be associated with a unique user.

ACKNOWLEDGEMENTS

I’d like to thank Alexander L. Wolf, Antonio Carzaniga, and Douglas Sicker for their feedback.

REFERENCES

- [1] Ross J. Anderson, “Why information security is hard - an economic perspective,” in *Proceedings of the Seventeenth Annual Computer Security Applications Conference*, New Orleans, Louisiana, Dec. 2001.
- [2] Elliot Aronson, *The Social Animal*, Worth Publishers, Inc., 1999, ISBN 0-7167-3312-9.
- [3] *The NSA Security Manual*.
- [4] Steve Hunt, “Market overview: Managed security services,” Tech. Rep., Giga Information Group, Apr. 2001.
- [5] Ross J. Anderson, “Liability and computer security: Nine principles,” in *Computer Security – Third European Symposium on Research in Computer Security (ESORICS ’94)*, D. Gollmann, Ed., Brighton, United Kingdom, 1994, vol. 875 of *Lecture Notes in Computer Science*, pp. 231–245, Springer-Verlag, ISBN 3-540-58618-0.
- [6] Ross J. Anderson, *Security Engineering - A Guide to Building Dependable Distributed Systems*, Wiley, 2001, ISBN 0-471-38922-6.
- [7] FreeBSD.org, “Secure programming guidelines,” <http://www.freebsd.org/security/security.html#spg>.
- [8] Richard E. Smith, “Cost profile of a highly assured, secure operating system,” *ACM Transactions on Information and System Security*, vol. 4, no. 1, pp. 72–101, Feb. 2001.
- [9] US Department of Defense, *Trusted Computer System Evaluation Criteria*, Dec. 1985.