

IT Policy for Organizations

Eric Blair

14 November 2006

This paper will provide a few useful points to the managers who are overseeing the managers of information technology. Its intent is to give those who have not spent their lives reading computer manuals an idea of what options exist for IT organization, and the social and business problems that the technologists must overcome.

Business v academia One could think of two paradigms in how an IT department is organized. The first is the business-oriented system. At an organization with such an IT department, every desktop has the same software, which is typically installed from a central server at the IT department. The IT department focuses its expertise upon this list of programs. All users must log in to the network, and all network activity is monitored and logged. Security is a very high priority.

One finds the other paradigm at many academic institutions, where students log in with virus-laden laptops that attempt to bring down the network without the student's knowledge. Ornery professors bring their 1985 copy of WordStar from home and insist that the IT department provide support. Network activity is generally logged, but everyone in the computer science department knows how to get around the logging system.

One may expect that the business-oriented IT is far-and-away more stable, but the reality is that the two paradigms are head-to-head. In the last few years, I have worked at two academic departments and two business-oriented organizations. Both academic departments had one (1) full time employee running the entire system, and suffered major failures at the rate of about once per year. Both business-oriented organizations had a IT staff taking up somewhere between one floor and one building. One organization was an IT mess where little worked, and the public-facing web page was filled with technical glitches and broken links. The other department suffered failures at the rate of about once per month.

Which leads us to the central question of this essay: with so much less effort put into security and stability, why don't universities have significantly more security and stability problems?

The remainder of this essay covers a series of small topics that address this question, including both business, social, and technical reasons. The summary: academics keep it simple, in a way that business users typically do not. There

are forces familiar to any businessperson, that push business systems toward complexity, that non-IT management needs to guard against.

I will try to avoid the details of software, but it is worth noting that the business-oriented users tend to run the Windows operating system, while academic-type users tend to run a POSIX system. [UNIX is a trademark of AT&T, so POSIX refers to any UNIX-like system.] This is not a hard-and-fast division, but you will see that each type of software facilitates its matching paradigm.

The division of labor Corporations are often divided into “Battlin’ business units” (as a comic by Scott Adams describes them). Each unit has an internal budget that it hopes to maximize, by billing other departments as much as possible while minimizing the list of tasks with which the department will dirty its hands.

Thank goodness the building services department is not organized like this. Imagine how unpleasant a workplace would be if a department was billed every time a radiator broke in mid-winter. The building services department, knowing that it has a full monopoly on radiator-fixing, could charge what it chose to, and perhaps the local department manager would give up and just leave the radiator broken. Maybe she would buy a few space heaters. Perhaps building services has already stated that radiators are not in the scope of things they are equipped to fix. Providing a vital service via a budget-maximizing, monopolistic department creates abundant opportunity for gaming on behalf of the monopolist.

Rather than allowing building services to define its list of services it will provide, it is typically given a broad mandate: keep the building in good condition. The details of what that means is left to evolve. On the consumption side, no one is ever billed for maintenance services.

This is how the academic IT department works. Their mandate is to keep the network working and the desktop PCs in decent working order. Some departments bill per computer, but this typically means a per-head charge rather than a per-service charge.

Conversely, many business-oriented IT departments bill per service or software item used, and carefully select the services they are willing to put on that price list. This is all entirely natural, and is exactly what is expected of them under the paradigm of the budget-maximizing business unit.

Having established that they will bill for services, what will the IT department offer? As a general rule, the more complex service justifies bigger budgets. That is, complexity goes hand-in-hand with budget-maximizing behavior—and complexity is the worst thing one can have in a computing system.

User expectations The sad truth is that the job title of every office worker may as well be “computer operator,” since almost all of us spend eight hours a day (about 2,000 hours/year) in front of a computer. Yet many complain bitterly when asked by a manager to spend a few hours learning details of the workings of the machine. IT departments and software authors often concur, stating that systems should be designed so that users can be blissfully ignorant

of the machines they use day in, day out.

Imagine this in any other context: a truck driver who does not know basic auto maintenance, an airline pilot who didn't read any flight manuals under the presumption that the controls will be entirely intuitive, a jackhammer that anybody can just pick up and use, a librarian who doesn't learn the cataloging system under the presumption that if it's not immediately obvious then the system is broken.

Intuitive is good, and a system that works *against* intuition needs to be fixed. Casual users (the archetypal Aunt Myrtle) will never see a payoff from hours of training. But the office worker of today is a "power user" by the standards of a decade ago, working at a job that vitally depends upon good software. It is absurd to say that the best tool for such a person is always the one that is most immediately intuitive and requires the least learning on the part of the user.

The presumption that users have the right to be ignorant of computer matters bolsters the Battlin' Business Units division of labor. Users who log in to a business-oriented computer see only those programs that they can use without training. The tools needed to do basic maintenance or adjust their system's configuration are for the most part missing. The expense and effort of training is saved, but in return workers can do less and are dependent upon IT for more. By mechanical metaphor, the oil for the jackhammer is kept in a locked box that only the jackhammer administrator can access. Keeping users ignorant and disempowered means that the IT department will never be obsolete, but means that even simple tasks require a call to the IT department.

In the academic approach, IT is more like building services: office workers aren't expected to take out the trash, but they are expected to maintain certain standards of cleanliness. There are usually abundant paper towels and waste bins scattered around to help with this. There is still a division of labor where most of the hard work in upkeep is given to specialists, but every user is expected to maintain partial responsibility and is given the tools to execute that responsibility. In the IT context, that means users are expected to have some level of training in maintaining the tools that they use every day.

The academic IT department aims to minimize users' dependence on support services. Typically, the IT administrator writes up a page explaining basic guidelines for maintaining system health, and users are expected to put out a reasonable effort to follow them. When major spills occur, the IT department is ready to clean up. Some users never read the instructions and never quite catch on. But they know enough to ask somebody nearby how to clean up their mess, and so a lightweight and decentralized support network evolves.

Security The basic strategy for secure systems is to keep it simple. A server on the Internet hosts a number of *services* that wait for data to come in, such as a web server or email server. To oversimplify a complex field, for an attacker to remotely break your organization's security, it must find a service that is taking in data and then send malicious data to the service.

If no services are listening, then there is no route to attack. So the basic

rule of security is to keep it simple: leave open as few services as possible.

Most academic departments closely adhere to the keep-it-simple rule. Their servers and workstations use a POSIX system that allows control over all network services, and leave open only ports for web, email, and a service known as *secure shell* (SSH).

Here, I am forced to briefly mention the architecture of the Windows operating system. For its operation, it requires a multitude of services that can not be turned off [Windows Time Service, taking input on port 123; Distributed Component Object Model, taking input from port 135; Microsoft Distributed Transaction Coordinator, taking input from port 3372; many more]. Are these services secure? Only Microsoft knows. So, for those who are daunted by the thought of keeping track of ports, services, and sockets, here is a simple summary for basic system security: don't run Windows. Because its complexity involves leaving open ports and services that users can not secure, it fails to follow the basic principle of keeping it simple.

Transparency means that you know what code you are running. Some argue that the route to security is to keep the programs you are using confidential—security through obscurity. The absurd name of this technique should tip you off to the fact that this method is not well-regarded. We must presume that those who hope to break into a system are smart enough to work out such details.

Unfortunately, many software vendors build their business on obscurity, keeping code that interacts with the outside world under lock and key. In such a situation, the IT department is simply handing its security concerns over to an external vendor, and hoping that that company will provide secure products.

Academia, meanwhile, has developed a number of systems that are entirely transparent—notably Apache to serve web pages, Sendmail to send email, and OpenSSH to serve SSH clients. In fact, the majority of the world's web and email (both academic plus business) is served using these open systems.

The summary: keep it simple. Every additional feature, in the operating system, in programs being run, and even in document formats, could potentially be adding a security hole. Which would be easier for an intruder to attack: a word processor document that only supports text, or a word processor document that can include embedded videos and web applications? Vendors and budget-maximizing IT departments press for these features, but in doing so they create potential security issues.

Interoperability The reader is well aware that standards are vital for interoperation. For example, the Internet exists because of the wealth of tools that implement the HTML standard in which all web pages are written. But the reader may not know how many sirens call to the IT worker pleading with him to break the standards.

For any standard, there are things that are difficult to do. Tool providers are well-aware of this, and thus provide 100% standards-compliant tools that one could use to write documents that comply to the standard—plus a few nifty features that make things easier. For example, Microsoft provides ActiveX controls

that make it easier to write web pages that change based upon conditions such as the user's location or preferences. One could implement a simple web page to get the point across using strict HTML, or use ActiveX to write a page filled with bells and whistles. However, only Microsoft's Internet Explorer (IE) can read ActiveX controls. This is no problem, our programmer reasons, because every Windows computer in the office has IE installed by default.

This is how the siren traps its victims: IE has never existed for a POSIX system, and is no longer supported on Apple systems. Thus, once our poor programmer has a large system in place using ActiveX, it is increasingly costly for the organization to switch to any other system.

The entrapment only gets worse. Let us say that you are confident that your company will never, ever switch away from systems that support ActiveX—that is, Microsoft Windows. Between now and the End of Time, no matter what shows up in the future, you will be a Windows user. Next month, the Microsoft representative will come in to negotiate licenses and upgrades for next year, and since your company has firmly committed in ActiveX code to using Windows until the End of Time, and your company can not operate without information technology, the sales rep can ask any price he wishes. By using ActiveX, you have signed away any and all bargaining power.

The summary: keep it simple. By sticking with standards rather than proprietary extensions, you have a lower risk of creating problems with other users, including both the people in the next department over and the clients who give you money. Standards allow you to keep your options open for future changes in the landscape, rather than wedding your enterprise to a single system.

Conclusion Politically, the stereotypical businessperson believes in the free market's ability to combine the efforts of free individuals to produce productive and even optimal outcomes, while the stereotypical academic is a socialist who believes in command-and-control central oversight by the well-informed elite. Given these stereotypes, it is ironic that the typical business IT system is a command-and-control system, and the typical academic IT system is a free-for-all.

The academic IT department makes promises like a *laissez-faire* government: we will make sure that the infrastructure you need—working computers and a working network—are in place and secure. If we have time, we will try to help you with your individual issues. Run whatever software you want, but you should be aware of the risks you take.

The business IT department expands upon this base significantly: we will also watch your behavior on the system, handle all underlying issues of computing so users can remain ignorant of the systems that they work with eight hours a day, and enforce our security policy by making sure that your computer runs only a limited range of programs. We will confer with our vendors to determine what new features users will receive.

They would do better to keep it simple. Every additional promise and demand by the business IT departments is one more potential security flaw and

one more moving part waiting to break. The reasons for the over-engineering are typical: the IT department wants to maximize its budget, and every new bell and whistle is a justification for more funding, and the IT department is feeling pressure from vendors who know that the company already has word processors and spreadsheets and must convince it to buy a new one anyway. Empowered users may compete with the centralized IT authority, and so tools are kept out of their hands—but supporting disempowered users requires greater complexity.

The academic approach depends on putting a modest amount of work in the hands of the users, giving them the responsibility of learning about and caring for their most essential tools, and then keeping things as simple as possible in the server room. Some especially inept users get to know the help desk very well, most just go about their business, and the entire system runs with a fraction of the oversight and costs of more command-and-control structures.