The Anatomy of Security Disasters

Marcus J. Ranum

Chief Security Officer, Tenable Network Security, Inc.
www.tenablesecurity.com

1   Introduction : Truth

I would like to use the opportunity I have been given today to tell a few unpleasant truths. Most of you already know the things I am going to discuss, or are at least aware of them. Since this is my microphone, for the next hour, you'll just have to bear with it – and, in doing so, I want you to know that I am not aiming these remarks at you. I am, simply, preaching to the choir. But I hope it's useful – if you are like virtually every other security practitioner I know, you feel either dis-empowered or puzzled, or some combination of the two. I'll try to explain and, if I do a good enough job, we'll have a shared understanding that might help us be gracious losers in this long war we're fighting. Since I started in security, 20 years ago, "they aren't taking security seriously" has been the constant complaint of the security expert. Even in organizations where security is taken seriously, it has been at the expense of living in a constant relationship of opposing management or other business units. Some of us enjoy the strife ; most don't. In fact, most of us enjoy being employed more than we enjoy being right. So, what's going on ? We've finally managed to get security on the road-map for many major organizations, thanks to things like PCI and some of the government IT audit standards. But is that true ? Was it PCI that got security its current place at the table, or was it Choicepoint, TJX, and the Social Security Administration ? This is a serious, and important, question because its answer tells us a lot about whether or not the effort is ultimately going to be successful.

It used to be that the most complicated thing ever built by humans was the space shuttle. We could argue a lot about how to measure complexity, but nowadays the popular wisdom is that software has become vastly more complicated than the space shuttle ever was. Space shuttles are insanely expensive things that are, unfortunately, vastly less reliable than they were originally expected to be. Personally, I find the idea of re-entering from orbit at a speed of 25,000 miles/hour to be literally incredible – but so is the idea of an operating system comprising of 40 million lines of C code hand-written by humans. As Rob Kolstad used to say "It's a miracle that it works at all !" While Rob can sit at a café comfortably sipping a hot chocolate while marvelling at his MacIntosh, I doubt very much that those are words you'll ever hear spoken out loud by a space shuttle pilot. If there were as many space shuttles being flown as commercial jets, they would be raining out of the sky every day.

We will come back to the topic of space shuttles in a few minutes.

2   It Hasn't Happened Yet

Computer security hasn't been tagged with an epic failure, yet. So far, computer security has "merely" been blamed for things like millions of dollars of losses. In its simplest form, the problem with computer security is that (like most risky propositions) it's easy to simply not worry about it

as long as "nothing has gone wrong, yet." That is the anatomy of simple disasters. But after a few years or decades of things going wrong, the problem eventually filters into popular awareness and people are sufficiently concerned to ask about it. In my opinion, this stage of the internet security disaster was passed in the late 1990s - everyone knew you needed some basic protections (firewalls, system hardening, etc) but the problem was sufficiently close to the cutting edge of experience that standard practices had not yet evolved.

et seems as if computer security has had some massive failures (TJX, Choicepoint, the US DOD NIPRnet) - and they are, indeed, significant, but we're now at the point where people are starting to realize that critical infrastructure attacks are practical. Quietly, while most security practitioners were worrying about how to disable ActiveX in their browsers, massive numbers of control and process systems were hooked up to networks that, simply put, they shouldn't be. I describe that as having happened in the past tense because it's important to emphasize that the computer security disaster has already happened - we simply have not yet reached the end of the sequence of events that started being put into motion in the mid 1990s. In other venues I've been fairly skeptical about the likelihood of "cyberterror" attacks - computer-based terrorist attacks on critical infrastructure - but the potential is vast. If (or when) something starts to go wrong in that area, we will be seeing the end result of a disaster that occurred over 10 years ago.

3   Time Line of Disaster

Now, let's take a slightly abstract look at how security disasters actually happen. This is based on my personal, direct, involvement with dozens of minor disasters, at every position in the time-line. These days I mostly seem to get involved on the forensic and archeological side - trying to figure out what went wrong, and where.

The time line of a typical disaster is straightforward. At the beginning of the disaster, a bad idea is proposed. Often, someone immediately tries to shoot it down, or point out its flaws. In very rare corporate cultures, the idea dies there and the whole disaster is averted. More typically, the bad idea survives - as does the trail of Emails pointing out the initial flaws. In archeological terms, those Emails are fascinating reading ; in forensic terms they are "discoverable information" or "evidence." They are usually forgotten, and sit in some engineer's out-box archive, just waiting to come back and haunt someone.

Next comes the most interesting part of the disaster. Suppose management is duly and accurately apprised of the fact that the idea is bad. If the idea something management really wants to do, there is sometimes a period of negotiation, or re-tuning. The idea bounces back and forth and has various tweaks applied to it, but two important things remain :
  – it is still a bad idea.
  – it is going to happen anyway.
Anyone who has ever been involved in this kind of disaster, from the technical side, will doubtless recall the horrifying feeling you get when you realize that you're trapped trying to deal with a bad idea zombie. No matter how many times you shoot it, or hit it with a shovel, it just keeps crawling forward : the longer it survives as a zombie, the higher the likelihood that it will be un-killable in the long-term.

Then comes the most crucial part of the disaster : the point at which management's expectations begin to form a reality gap. Generally, this happens because management believes it has set out some objectives, and does not realize that those objectives are being renegotiated because the basic objectives are literally impossible or simply ridiculous.

Let's look at an imaginary example of how the beginning of a security disaster might play itself out. Our CIO attends a CIO conference in which other CIOs are bragging about how successfully they have outsourced management of their credit card processing to an internet banking service in Nigeria. So the CIO comes back, and announces that we are going to do likewise and gets the disaster started. One of the CIO's direct reports asks a member of the security team to research the topic and "get back to me with some recommendations." The security engineer thinks about it for a day or two and writes his boss an Email explaining why it's a terrible idea, namely : a) we're exposing our most critical data and b) the Nigerian banking service has an @yahoo.com mailing address.1 This feedback gets filtered back up the chain of command and bounces back and forth in the "Negotiation" process. During this process, the bad idea has various hypothetical mitigating fixes applied to it, making it more complicated and less obviously bad. For example, we agree we'll have "compensating controls" and a "service level agreement" with the partner, etc. Perhaps a completely different plan (plan B) will be proposed, but by this time too much effort has been expended in attempting to dress the zombie up in a tuxedo, it's too late.

The executive suite's expectations have not been adequately reset. Up in the corner suite, they see people working hard on making the idea come to fruition, plans are being made, and considerations are being weighed. The trade-offs that are being made, which place the organization at risk, are being somewhat improved with compensating controls, but nobody has been able to break it to the corner suite that we're still dealing with a fundamentally bad idea. In the most dysfunctional organizations, you get senior (or sometimes mid-level) executives who 'shop a bad idea' until they find someone who is willing to tell them it is good. One security disaster I was involved with happened in exactly this manner : a senior executive hit upon a bad idea and asked the security team. The security team explained why it was a bad idea. So the executive asked the web design team, who declared it a great idea and highly do-able. Months later, the "whiners" in the security team was presented with a fait accompli in the form of "we're ready to go live with this, would you like to review the security ?"

I have seen this play itself out dozens of times, in critical decisions made regarding IT security during the course of my career. The reality gap comes into play when the executive decided to shop the idea around : he asked for this thing to be done securely, and got a resounding "no" from the security group, but a "yes" (it can be done, but not securely) from the web group. All he hears is the "yes" and when the whole thing blows up a year later, his memory will be that he asked if it could be done safely, but someone lied to him. What's really happening is that the disaster was inevitable from the moment when the bad idea took hold.

4   The Post Disaster

Archeology of a disaster is almost always extremely depressing, because you learn over and over again that disasters are not unavoidable. If senior management believes that they were lied to they wind up getting disabused of that theory when the Email memos start to surface. I've been involved in a couple of disasters in which senior managers say "why didn't they tell me about this ? !" and someone has to suck up the courage to point out their Email address was on the Cc : line of recipients : "They did."

Simply put, such disasters are purely the fault of bad management ; managers who 'shop' bad ideas, or who create organizational cultures in which staff that point out problems are "whiners" or "nay-sayers." Unfortunately, in most businesses, senior level managers are recruited for being "can

do" types who get the job done, which means that you're particularly in danger of having to deal with a senior executive that is comfortable with living with a serious reality gap.

Technical security experts that I've worked with often feel that you can "butt cover" by making sure that the managers who "own the decision" accept responsibility for it, and that if something goes wrong, it'll be the managers who are accepting the fall. The unfortunate truth is that that seldom happens. What usually happens is that senior management simply claims they were not fully apprised of the decision and that they never would have approved it if they had been" - in short "they were lied to" as Alan Paller would say. When a security disaster happens, the person it is most likely to fall upon is the mid-level security manager - ironically, the person who is most likely to have been the recipient of the Emails from the technical staff saying "this is a bad idea." That mid-level manager is then the person in the ideal position to vindicate their actions by turning whistle-blower or dumping the Emails in which senior management was informed of the reality of the bad idea they were approving.

What is fascinating and sad to me is that when the dust clears and the bodies are buried, very little has changed that would prevent another disaster from happening. The reason for this is that all the focus is paid on the tail end of the disaster, when the real disaster happened at the moment when the bad idea was allowed to become a zombie. I did some work a number of years ago with a large company that had a very skillful security manager who was a master at killing bad ideas before they took on a life of their own. In that case, it was his willingness to tell senior executives "this is going to blow up in your face if we proceed." The only way to prevent security disasters is to have a security team that is fearless about feeding back information up to the top of the chain of command, and to have senior executives who make decisions based on reality not projection of their fantasies.

5   Risk Management : A Disaster Waiting To Happen

If you accept the argument I am making so far, perhaps I can convince you that "risk management" is a fiction that plays into the disaster-cycle. The premise of risk management is that you will quantify the risk/reward of a decision, then assess the likely failure modes and attempt to reduce them appropriately in detail. Inherently, the risk management approach is too late in the cycle : we've already chosen to execute a bad idea, and now we're arguing about what we can do to reduce the impact when it goes wrong - not if. If you think back to my example of outsourcing to the Nigerian bank scammer, I think you can see that there's no amount of compensating controls that can usefully be employed if you're dealing with someone who is willing to lie. I've actually seen this happen before, in an outsourcing project in which it became clear that the outsourcers were going to : a) say whatever took to win the project and b) do whatever they were going to do, anyhow, after they did. The premise of risk management - that the risks of certain activities can be understood and managed, falls apart when you're dealing with a reality gap.

On that particular topic, I cannot better the words of Richard Feynman[1] :

> It appears that there are enormous differences of opinion as to the probability of a failure with loss of vehicle and of human life. The estimates range from roughly 1 in 100 to 1 in 100,000. The higher figures come from the working engineers, and the very low figures from management. What are the causes and consequences of this lack of agreement ? Since 1 part in 100,000 would imply that one could put a Shuttle up each day for 300 years expecting to

---

[1] From Richard Feynman's "Personal observations on the reliability of the Shuttle"

> *lose only one, we could properly ask "What is the cause of management's fantastic faith in the machinery ?"*

Ultimately, risk management is a numbers game ; you multiply a wild-ass guess by a fudge factor. Worse, the potential cost of failure is estimated in as a factor, too. So you're trying to balance an unjustified estimate of cost of failure against a wild-ass guess multiplied by a fudge factor. Generally, what is really going on is that risk management is used as a sort of statistical shell-game to manipulate the perceived value of security when dealing with a clueless senior manager. Bluntly : it's lying with statistics.

Feynman's description of the foolishness of trying to estimate the "effective lifetime" of a space shuttle main engine should be required reading for anyone who claims to believe risk management is practical. To summarize it : you can only play Las Vegas odds-maker when you're working on small numbers of variables and extremely well-understood conditions.

6   Improving Communication and Education

Another model for preventing security disasters is to attempt to improve intra-organizational focus and communication about security. When I've seen organizations promote security awareness efforts, they usually don't work. If they do it's almost always because senior management has been careful to create an environment in which bad ideas are less likely to take hold, rather than one in which there is improved communication about bad ideas. I had one client years ago, which instantiated a process of "security sign off" on design - essentially formalizing a delay-loop caused by paperwork between Stage #6 and Stage #7. Essentially, only the certified zombies of bad ideas could get through the process.

The place to improve communication is at the earliest stages of the disaster-in-progress : in other words, you can simplify this to the simple observation : "You need executive management that does not make bad decisions, takes security into account, and listens." Formulated that way, it's pretty much a statement of the obvious - because it's an obvious truth. The organizations where I have seen effective communication about security it begins and ends with senior management asking direct questions about security considerations and not accepting hand-waving for an answer. In an environment in which senior management place security concerns on the agenda for technical decisions, you'll find that security education is not required ; it simply happens because the appropriate people come to realize that it's part of their job.

7   Legislating Security Failures

The problem with legislative approaches to encouraging security is that the legislation always happens too late. Take, for example, the rush of legislation regarding personal information disclosure. We are dealing with a rush to regulate all sorts of aspects of data leakage, but it would make vastly more sense to regulate who has access to what data. Of course, it is way too late for most businesses to even form a vague idea who has access to what ; consequently the press is filled with accounts that read : "laptop full of customer data left in airport departure lounge." The focus is on "what do we do about that data leak ?" not "why on earth is customer data wandering around airports on laptops ?"

For people like me, who enjoy reading between the lines, the reports of data leakage ought to be particularly terrifying because of the unsaid revelations that they make : namely, random employees

and contractors at most organizations have complete, unfettered access to virtually everything. Imagine taking an organization that utterly lacks internal controls on data access, and attempting to retrofit them! Most IT managers would throw up their hands in despair - can you imagine what it would cost to figure it out, now?

In case I am not being sufficiently clear, I think the IT world crossed a Rubicon in the late 1980s, in which control over information was effectively abdicated. That has huge implications for the scope and lethality of security disasters because it generally means that a single penetration into an organization is effectively a complete penetration of all the organization's information assets. Those of us who enjoy living in global military/economic superpowers should be very unhappy at the thought that the situation I'm describing applies to most US Government agencies and virtually every significant commercial entity.

A few years ago, when a friend and I were discussing this problem at a conference, he said, "Yeah, but what should we do about it?" The only answer I can honestly give is : "The wrong decisions got made 15 years ago and now it's too late to go back and un-make them."

Legislation leaves us simply with formalizing damage control when the disasters occur, with a few good ideas for damage containment thrown in where possible. But the assumption (like with risk management) is that security disasters are going to be inevitable, huge, and frequent.

A fad that is related to risk management is the use of economic models to talk about security. I'm not, I confess, "up" on the latest literature in this area, but the gist of the idea seems to be to take a risk management approach and then try to reason about what actions make the most sense from a standpoint of cost-effectiveness. I have already asserted my assumption that risk management consists largely of compounded wild-ass guesses, so an economic model built to optimize wild-ass guesses is not going to be worth any more than the paper it's printed on. Fans of economic models that I've talked to have explained security to me in terms of it being a "market failure" - a market failure being what happens when a market does not adequately self-regulate because customers don't have anything rational on which to base their decisions. Outside of Las Vegas, there must be very few markets that are not "failures." Security is, certainly, one. When economists talk about a "market failure" I hear the sound of hands waving; I do not think we can expect any help or illumination from that quarter.

8   The Disconnect

What inspired this paper is a discussion I had with Alan Paller, the founder of SANS and the CIO Forum. He was quoted in an article as saying something to the effect that CIOs were regularly lied to regarding security by their technical staff. Bear in mind that this is the viewpoint as expressed from the position of the CIO : corporate executives felt that they had done their job when they told technical staff to "make it secure." Technical staff had cheated by doing naughty things like leaving unauthorized connections between critical networks, etc. He concluded by offering the following observation :

> It's a problem for security people because their career depends on their ability to enable the business securely. We have had six years of 'regulation-based job security' for the whiners. That era is coming to an end.

My intent here is not to pick on Alan, specifically - it's a viewpoint that I've heard over and over again from IT executives as well as security practitioners. The short form is "there is a train coming and you'll either clear the track or get cleared off the track." In either case, when the train has gone

through, the track will be clear - that is the important end-result we need to focus on. What I find particularly interesting is that Alan's viewpoint appears to be that security has actually managed to regulate, contain, or slow the oncoming train-wreck whereas I think it hasn't. My suspicion is that the "reality gap" between management's expectations and what they actually have out there on their networks is larger than they realize. I think it is vastly larger.

9   The Reality Gap

Here is the problem : if there is a 'reality gap' between how secure our networks are expected to be, and how secure they actually are, how do we bring them back into line with expectations ? At this point, I expect most of you to be scratching your heads, thinking, "that's impossible. It was hard enough to get things as locked down as they are, now, never mind going back and re-assessing the status quo." If you look at the implicit criticism in Paller's comment, the "whiners" careers are already at stake, and business enablement holds the upper hand. Those of us who have spent our lives as security "whiners" and "nay-sayers" have cause to be concerned because most of us see that business enablement has always held the upper hand. If Paller's right, he's saying that the gap between security perception and reality is about to get dramatically worse in the next decade.

In fact, there is no way that it can get better. Because of the depth of the reality gap, "throw it out and start over" is not a justifiable option (after all, nothing terrible has happened yet !) and there is now a huge installed base that represents massive intellectual and financial inertia. Perhaps a few of you have already had the experience of trying to encourage a client to think a little bit before embarking on a Web 2.0 rewrite of a crucial application. That particular train has already left the station and is coming toward us. Old timer security practitioners know that the place to build in security is at design-time, but we are faced with a vast mass of moving code, all of which is past the critical point at which it could really be improved[2]. If it's too late to get into the cycle at design time, the only option remaining to the industry is "disaster and patch."

10   Space Shuttles

When the Space Shuttle Challenger blew up on take-off, NASA went into disaster management mode. A truly epic failure had occurred on prime time television, and everyone was asking "what went wrong ?!" The simple answer to give would have been "space travel is dangerous" but unfortunately NASA management had distanced itself from that reality ; there was a gigantic reality gap between the actual safety of space flight and the expected safety of space flight. NASA empanelled a blue-ribbon panel of experts who were chartered to stumble around and write a set of conclusions that would basically read "space travel is dangerous, but NASA's doing a great job." Instead, something unique happened : Nobel Laureate Richard Feynman got invited to join the panel, and he wound up conducting his own investigation and produced his own report - a masterpiece that succinctly and brilliantly explained how an organization like NASA could establish a reality gap regarding safety explanations. It's one of the best technical white papers ever written, I believe, and I highly recommend you read it (it's widely available on the internet). Feynman describes an environment in which management expects a certain level of performance from a component, then accepts compromises that reduce the performance level - without adjusting their expectations. Does

---

[2] There is a metric called the "Doomey Microsecond" and it is defined as "the period of time between which not enough is known about a problem to make a decision, and everything is set in stone."

that sound familiar ? Furthermore, he goes on describe how NASA rocket scientists attempted to wave away significant component failures as "acceptable flight risks" because they had not resulted in a flight failure yet.

If the attitude of "this risk is acceptable because it has not resulted in a failure yet" sounds familiar to you, it should. That's the history of the computer security disaster in a nutshell. The important lesson from looking at Feynman's analysis of the shuttle is that we have another field full of very smart people thinking very hard and giving good advice - all of which is contributing to a widening gap between expectation and reality.

Not to belabor the point too much, when the Space Shuttle Columbia broke up on re-entry, the failure analysis revealed exactly the same type of expectation to reality gap had evolved regarding the shuttle's tendency to lose tiles, as in the Challenger's solid rocket booster leaks. Basically, NASA managed to learn nothing from the first failure, for exactly the same reasons that the computer security industry manages to learn nothing from its failures : fixing the problem would entail re-visiting decisions that were made decades ago and, besides, it would be too expensive to re-visit them now. There is a very real and useful analogy here, though : periodically even NASA will look at a design like the shuttle's and obliquely admit that it was a botch. "It's a miracle that it works at all" does not just apply to the Internet. Unlike virtually everyone else, the pilots of the space shuttles have a fairly realistic assessment of the situation : "space travel is dangerous."

Putting an organization or a country's information assets online is dangerous, too. Putting them on a network is even more dangerous, and exposing them to the Internet is most dangerous of all. There is simply no other conclusion that you can realistically reach. Consequently, I argue that there are a great many places where it would make sense to retrench capabilities off the interwebs and to reduce the number of network-controllable SCADA systems. Every day that goes by increases the reality gap between our expectation of data security and the degree to which it is at risk.

11   Breaking the Cycle

What can we do to break the cycle ? The most important thing is to make sure you are direct and honest about expectations at all times. Do not allow management or clients to believe that they can do dumb things in safety, and do not hide behind bogus probability guesses. "Safety" is not the same thing as "relative safety."

Pre-allocating blame is crucial to keeping the reality gap as small as possible. When management negotiates a control out of the loop, do not simply allow them to assume "it's OK" - go back and remind them that the parameters of the design have changed. As Alan Paller said, executives feel that they are being lied to by technical staff that are taking shortcuts[3] - reality check them by circling back with an update. E.g. : "By the way, since you asked us to keep the management costs on that system down, we have followed your directive and are now using an Internet-based remote control interface. Since the old system had zero chance of compromise over the Internet, and the new one has something more than zero chance, we can realistically say that it is infinitely more dangerous to proceed in this fashion." I believe that is how a nay-sayer would put it.

During Feynman's analysis of the Challenger disaster[4] he describes a number of the memos from rocket engineers attempting to express concerns up the management tree. The general tone of the

---

[3] If you believe them, I have a big iron tower in Paris for sale ; contact me if you're interested.

[4] A new book, "Classic Feynman" includes a short article by Feynman about the entire experience of being on the panel and producing his report. It's a fascinating study in its own right of how large organizations respond to failures.

memos is uncertain and rich with weasel-words and "could be," "might," and "should." In order to help bridge the reality gap, you must keep your communications as clear and unambiguous as possible.

I.e. : "Mister President, I believe that the US Government's over-reliance on contractors in the area of federal government-related information technology represents a clear danger to our national sovereignty."

12   Hope

The failures I am describing are failures of hope - they are the consequence of human optimism. With regards to computer technology, the hopefulness combines our love of shiny new widgets and the scientific culture of information technology : more really is better, even if it's unnecessary and dangerous to the point of stupidity.

Speaking of "dangerous to the point of stupidity" the next disaster is already beginning, and it's in the form of Web2.0. I don't think it will be possible for it to be as bad as Web1.0 was - we are nowhere near to reaping the "benefits" of that one - but the Web2.0 model encourages dis-integration of information assets at the data level. Nobody will even have an idea where their data is getting processed, because it's all being sent "out there" for mysterious things to be done to it. When a page is rendered, part of the page may be rendered by a site (we think we...) trust, while another part may be rendered by a commercial provider who aggregates data with our competitors. What does "trust model" even mean in that kind of environment ? How do you separate reliable data from unreliable data, when everything is loosely coupled behind chains of uncontrolled servers. I'm already seeing the tip of the iceberg - the other day I was running a copy of Tenable's Passive Vulnerability Scanner while I browsed to a couple of myspace pages. During the course of a few minutes of browsing, the vulnerability scanner identified easily exploitable holes in 5 of the banner/ad server applets that were coupled into the myspace pages. Users have no notion of which data items are served from a "trusted" provider and which from a "trustworthy" provider. On the Web1.0 systems we are already dealing with transitive trust weaknesses that scare me ; Web2.0 promises vastly worse problems. Cybercriminals are already making effective use of the basic Web1.0's failings in this regard ; doesn't the rush to Web2.0 simply signal a desire for heavier arterial bleeding ?

13   Giving Up

When things get sufficiently bad, eventually you simply give up. If you're NASA, once you've spent your budget on space shuttles, you can't go back and say "Oops. Give us another umpty-billion and let us try again." But if your system is too complex and inter-dependent, there's no way to reconstruct everything that might have happened. At that point, failure analysis becomes prohibitively expensive, as well.

Think about that for a second : if it is prohibitively expensive to figure out what went wrong, then it's impossible to fix the problem. You're left with no alternative but to slap duct tape over it and keep going. But, unless you understand the problem, you're left with the fact that you have a flawed design and your failure rate (all things being equal) will be pretty constant. Put another way : we can state that space shuttles seem to have about a 1

In 1997, at The Black Hat Briefings, I suggested that we should scrap the Internet and our installed base of Internet apps, and start over (blaming the whole thing on Y2K). Everyone in the room laughed. But I wasn't joking.

Obviously, from the content and the tone of this presentation, I think it is already too late. There is too much momentum to an inherently dangerous process, and it will go forward until there are severe-enough disasters that something has to change. But, consider, when you look at an organization like NASA that can lose not one, but two, multibillion-dollar space shuttles and their pilots to the same kind of reality gap, it will take something extremely severe to wake up a national-level response. What might that be ? We already have US Pentagon spokespeople alleging that "Chinese hackers" have stolen "10+ terabytes" of information from the DOD's unclassified networks[5] - such an information leak could result in a superpower transitioning into a 3rd rate power, but the failure would be too for anyone to figure out.

I used to get satisfaction from saying "I told you so" but this is one I'd rather miss.

Thank you.

---

[5] The "unclassified" networks are the ones where all the military's logistical systems, email, payroll, and personnel data are stored. Do you think it would be possible for an enemy to learn anything useful from that ? No way !