Work - in - Progress

Pareto-Secure

A definition of security using the theory of Pareto Efficiency

Ian Grigg
Systemics, Inc.
2004 - 2005

$Revision: 1.16 $
$Date: 2005/12/25 23:04:21 $
Comments on FC++

Abstract: What do people mean when they say something is secure?

Shamir's 1st law says absolute security does not exist, yet the popular press and the security buying process is inundated in secure product. For some of these products, there may be merit in the term, but for many it is more debatable. Such differences of meaning and applicability suggest low efficiency in the market for security, as well as a blackspot on the claim for security as a robust science.

One way to define 'secure' is to apply the economics theory and terminology of *Pareto efficiency*. This simple structure gives an easy way to categorise and choose among alternates, and identifies when an optimum has been reached. We suggest that this meaning may already be in wide spread usage, intuitively, among security practitioners and the popular press.

Preamble

On the Nature of Security

Pareto-efficiency is an economics concept dating from the early 20th Century [VP]:

> *A change that can make at least one individual better off, without making any other individual worse off is called a Pareto improvement: an allocation of resources is Pareto efficient when no further Pareto improvements can be made.*

This essay applies this framework to the field of security, especially that of cryptographic systems.

Security is substantially complex. In constructing security systems, we build from strong components. A most common reductionist technique of security professionals is to create every link in the chain as strong as possible, and hope that no link becomes too weak. At the component level, this is well understood, and indeed in cryptography we have many strong components.

Strong cryptographic components might be a mixed blessing. Much effort is undertaken to deal with the inherent complexities in interaction between these components, yet the framework for reducing these complexities is not well founded. At the systems level, our concentration on strong components has sometimes distracted us from the difficulty of combining them. The result is that we often do not challenge our system assumptions, sometimes holding on to these assumptions beyond their scope and lifetimes.

This is seen in its extreme form in the popular press and the purchasing process. For example, in cryptosystems, it is almost not a caricature to say that as long as a cryptosystem uses AES, triple DES, RSA with key lengths of 1024 bits or more, and so forth and so on, not only are these considered secure choices, but the entire system is considered secure. Such a leap from components to systems is unwarranted, yet the signal of an over-strong algorithm remains strong.

Can economics add anything to these questions? This essay introduces Pareto efficiency to security. We suggest that this framework is already in use at an intuitive level; when a component is claimed to be secure, it is meant that it is Pareto-secure. Using Pareto-security provides at least one view on how we can rate components within a subsystem, and gives us limits in statements about systems security.

S1 - 3 laws of security

- Absolutely secure systems do not exist
- To halve your vulnerability, you have to double your expenditure
- Cryptography is typically bypassed, not penetrated

Adi Shamir, Turing Award lecture, 2004.

Acknowledgments

I received valuable feedback from Twan van der Schoot, Bryce Wilcox, Daniel Nagy, Graeme Burnett and Nick Szabo. The original idea to apply the analogue of Pareto-efficiency came out of discussions with Adam Shostack.

Prior Work

Measuring Strength in Cryptography Components

Lenstra and Verheul created a framework to analyse key strengths based on the life span of security [LV1]. This approach was suggested by their observations that it was understandable to the external user community, and they explicitly caveated that they offered no rational founding for it other than user comprehension [LV2].

The 1st Law

In his Turing Award lecture, Adi Shamir introduced three laws of security, see Sidebox 1. The 1st law states that

> *"Absolutely secure systems do not exist."*

We can show this by a simple contradiction argument. A secure system must be secure according to a frame of reference. Within the context of the frame of reference, it is possible to analyse and prove all component statements and system assumptions, and to construct an absolute security statement for the system.

Yet security, unlike many other sciences, suffers from the drawback that the attacker is aggressive. Within the real world, it is impossible to reliably create a frame of reference that is absolute. It is always possible for an attacker to challenge convenient assumptions and to redraw the framework in order to break the system.

As there can be no reliable frame of reference, there can be no absolute in security.

Relative Security and Components

Notwithstanding the absence of absolute security, we can make some relative judgments about security components and we can make statements of practical utility based on those relative judgments. We can make comparative judgments between two components bound by a common set of assumptions, and we can make judgments about how differing components interact within an encompassing system.

S2 - Is AES absolutely secure?

Consider the AES algorithm, the current day standard in secure secret key encryption systems. There are no known attacks on this algorithm that are better than brute force, and current calculations put brute force attacks outside our capabilities [NIST1]. Thus to all intents and purposes, AES is secure, and it is recommended without caveat by many experts.

Yet this rides on a simple assumption that the secret key is kept secret. Within a real world cryptosystem, if an attacker adjusts the security framework and reveals the key, then the cryptosystem is broken, and AES is "no longer secure" by some measure.

It is in the challenge of the assumptions that the algorithm is rendered defeated, which is, according to the 1st law, acceptable. Because there are no absolute assumptions, there are no absolutely secure security systems.

Consider two algorithms, DES and AES. These are interchangeable at some level of software engineering, and most cryptosystems envisage some level of selection of these or similar algorithms. (From here on in, we assume that the key is part of the wider cryptosystem.)

As seen in Sidebox 2, we know of no practical attack against AES. In contrast we do know of practical attacks against DES [DT] [NIST2]. One algorithm was designed in the 1970s, the other was designed in the late 1990s, and these differences are telling.

Hence, we can state with some degree of confidence that replacing DES with AES results in an improvement of security: it eliminates the class of attacks that now afflict DES. Furthermore, it does so at a cost that is low or trivial. The only cost appears to be that AES is slower than DES and we ignore that here [TDES].

This improvement then is a decided win, at no cost. Now consider two variants of AES, being 128 bit key length and 256 bit key length. Both are strong, but AES256 is many orders of magnitude stronger. As it is relatively much stronger, it might be thought that swapping AES256 for AES128 would be an improvement.

Yet, in terms of results, even though one is stronger than the other, both are so strong that there is no practical attack. Thus, there is no improvement in resultant security possible in either choice, even though one has a stronger theoretical envelope.

Note however that the claimed improvements are only as robust as our assumptions, for example there being relevant attacks against DES. Later, we challenge these assumptions, and modify our approach.

Components and Systems

As Sidebox 2 suggests, AES is secure, unless we have to consider the key. Normally, a design would consider the algorithm to be a secure component, and the key's security would be punted up to the next layer. Often, such issues as key security get punted from layer to layer until they end up as assumptions in the highest layer, being the security system.

This then indicates the relationship between components and systems. Components can refer their weaknesses as assumptions out and upwards, but this only goes so far as the system. The system then must take these assumptions and live with their consequences; "the buck stops here."

And this is why the 1st law states that there are no absolutely secure *systems*. Each system inherits the assumptions of the components. Hence a corollary to the 1st law might be that in a security system, *all assumptions are subject to challenge*, and the same corollary bridges to the 3rd law, *cryptography is typically bypassed, not penetrated.*

Pareto-Secure

The theory of Pareto efficiency

S3 - Wikipedia on Pareto

"Pareto efficiency, or Pareto optimality, is a central concept in game theory with broad applications in economics, engineering and the social sciences. A change that can make at least one individual better off, without making any other individual worse off is called a Pareto improvement: an allocation of resources is Pareto efficient when no further Pareto improvements can be made."

Economics describes such improvements within an allocation of resources under the theory of *Pareto efficiency* [VP2]. An improvement is a *Pareto improvement* if a change made results in an improvement in efficiency at no commensurate cost elsewhere in the equation. And a solution is *Pareto efficient* if there is no further *Pareto improvement* that can be made.

Introducing *Pareto-secure*

Economists use the metric of economic efficiency of allocations between competing agents. Applying this framework to our security metrics, we introduce the analogous terms *Pareto-secure* and *Pareto-secure improvement* to refer to security metrics resulting from allocations of competing choices in an overall design.

A change is a Pareto-secure improvement if a measurable and useful improvement in security results, at no commensurate loss of security elsewhere. A choice of algorithm could be a Pareto-secure improvement if we could measure the result as being more secure, and it would not be a Pareto improvement if either some other cost in security is incurred, or no improvement is measured. (Where there is no need to distinguish the term, Pareto improvement should be sufficient.)

Then, a choice of AES over DES is a Pareto-secure improvement [K]. But using AES256 over AES128 is not a Pareto-secure improvement [AES5].

In economics, if we were to show that there was no algorithm that delivered a Pareto improvement in economics efficiency over AES, we would conclude that AES is Pareto-efficient. Likewise, a component is Pareto-secure within a system if

1. It is secure against all practical attacks, and
2. there is no Pareto-secure improvement that can be made

As AES is both secure against all practical attacks, and there is no improvement we can make that will offer better realised security in an algorithm, we suggest that AES is Pareto-secure. All of the AES family of algorithms are by this logic Pareto-secure, according to our assumptions. However, none of them represent a Pareto-improvement over any of the others.

Components within a Security System

We have to this point been quite loose on the environment of the component. Note that the above requirements were limited to a single system (and that the first requirement implies the second). Now consider a security system of two cooperating asymmetric parts, *Yin* and *Yang*. The two parts work together to make a complete system, where each part does a different job and they link together well.

S4 - A Simple Unbalanced Cryptosystem

Consider a system with a 4 digit PIN encrypting with a strong encryption algorithm. Attackers can take any number, encrypt with it and test it against an oracle that confirms, correct or not. Or they can attack the algorithm.

In this simple cryptosystem, the PIN dominates as the weak link. Assuming the encryption algorithm is even remotely strong, the dominating attack is to try all of the 10,000 possibilities. Attacking the algorithm would be inefficient.

If chosen for such a cryptosystem, DES would be Pareto-secure. Putting in AES would not result in a Pareto improvement. Any Pareto improvements would come from adding digits to the PIN, and many more digits would be needed to match and stress the strength of DES.

Let us assume that the first part, *Yin*, reaches a particular strength, with known weaknesses. Let us further assume that the second part, *Yang*, well exceeds that strength, and there thus results an imbalance between the two in contribution to the security of the system. If the imbalance is severe, the weaker component becomes the dominating factor - the *weak link* - and it will always be attacked in preference.

The strength of this system can not be improved by substituting in a yet stronger alternate for the second component, *Yang*. This means that there is no Pareto-secure improvement to be made with *Yang*, and therefore *within the confines of the security system* we can assert that *Yang* is Pareto-secure.

Introducing *Pareto-Complete*

Note how narrowing our context to a specific cryptosystem enables us to identify a context-dependent security statement. Within this narrowed context, we declare components to be Pareto-secure, but they may not be so secure elsewhere. This is very useful within the context of that security system, but it does add the cost of needing to be always aware of our specific set of assumptions.

In the construction of practical security systems, we need to analyse and design according to a set of requirements, and within the limitations of a set of assumptions. This saves costs in the design phase, but incurs costs in the long term. The original designers of a system often know well their requirements and assumptions, but later participants will not. Engineers, implementers and operators that follow in the footsteps of the original designers may make changes that challenge the security of the system.

There is then a use for a definition of security that survives regardless of the limitations of a chosen security system. For this, we introduce *Pareto-complete*.

If we can show that *for all reasonable sets of assumptions and all reasonable security systems*, there is no choice that offers a Pareto-secure improvement over our initial choice, then we can say that this component is not only Pareto-secure, but also *Pareto-complete*. That is, choosing this component is a reliable security choice, no matter what other weaknesses exist in the rest of the system and no matter how those assumptions are changed [PC].

S5 - Is AES Pareto-complete?

AES is both secure against all practical attacks, and there is no improvement we can make that will offer a better security choice in a cryptosystem, regardless of the cryptosystem. Of course this rests dangerously on our choice of what exactly are reasonable assumptions in all security systems.

1. The key. As the key represents such an asymmetrical threat to the algorithm, it is common security practice to model the key breach as a breach in another component or layer of the system, rather than as a breach in the algorithm.

2. Time. The second assumption to challenge would be the march of time. NIST states that AES is "secure for at least 20-30 years [NIST]." The Key Length Calculator places 128 bit key lengths as secure until 2088, using Lenstra and Verheul's framework [UCL] [LV3]. We know of no generally fielded cryptosystems that require even 20 years.

3. Esoterica. It may be that AES is Pareto-secure for all commercial security systems, but there may also be esoteric 'national security' systems where the assumptions are not so robust. Our attitude to those exceptions is to excuse them from being reasonable; such systems designers know that they are pushing the envelope, and they have the budget for that.

Thus, we suggest as our working hypothesis that AES is Pareto-complete.

Combining Components

It is possible to combine components, and apply a similar logic. Consider the following stylised example based on the commonly accepted models of TLS and SSH. Note however that real world implementations can markedly differ, and can indeed reverse many of the following assumptions.

The popular TLS protocol is built out of algorithms such as AES and other components [TLS] [SSL]. It is mature, well studied, and widely implemented. Researchers have been finding minor flaws for some time, yet the discoveries of major flaws have dried up.

In terms of the deliverables as a public key connection-oriented transport layer protocol, we do not know of one that delivers more measurable security, as long as a Pareto-secure set of algorithms is chosen. For comparative example, the underlying protocol behind SSH is thought to be highly similar [SSH] [SecSh]. We could then suggest that both TLS *and* SSH are Pareto-secure protocols, and may even be Pareto-complete.

Yet, if we go wider and higher, and bring in their respective regimes for public key exchange, as additional components, the differences become more marked. TLS applications commonly punt key authentication to a trusted third party (TTP), while SSH applications commonly ask the user to confirm a newly discovered key [Options]. In order to overcome the weakness inherent in adding another party to TLS, we could switch to the SSH key exchange model, but that brings in the weakness of the user's first confirmation leaving open a potential *man-in-the-middle* attack. Likewise, switching SSH over to TTP authentication adds the weakness of the TTP.

Hence, we cannot suggest that either of TLS+TTP, or SSH+user-confirm are Pareto-secure. Substitution of either key exchange regime results in benefits but more importantly, costs. In order to achieve a Pareto improvement, we would need to do one or both of:

- limit our assessment to a particular security system where we could overcome the weaknesses, or
- expand the key exchange regimes to include the benefits of both, and incur no costs over either alone.

By way of example, consider secure browsing. We could hypothesize that TLS+TTP was Pareto-secure within that framework, a framework that includes the user, her browser, the net, the site server and the TTP. Consider however the unfortunate weakness of external links being presented by pseudo-authoritive means such as emails. That is, phishing, which at its minimum is a breach in the spoofing protection of the secure browser.

Several parties have proposed a fix to the browser security model to address phishing links [IG] [AA] [TC]. Each certificate would be augmented by additional information known only to the user (such as a petname or an individual logo) and this would be displayed to her to assist her in discovering fraudulent links. As this addition to the certificate improves security, and does not result in any security costs (the TTP authentication is augmented rather than replaced), it identifies a Pareto-secure improvement. Thus, TLS+TTP is not Pareto-secure, within the context of the secure browsing application.

Until these Pareto-secure improvements are made, TLS+TTP would remain short of Pareto-secure. Likewise, a component built of SSH and its key system would not be Pareto-secure, as improvements are needed in how keys are distributed and authorised in the initial instance.

Conclusions

Summary

A Pareto-secure improvement is made to a security system when a substituted component results firstly in an improvement in security, and secondly with no cost to security elsewhere.

We say that a component is Pareto-secure if within the confines of its security system, there is no Pareto improvement to security in changing it for another component, and it is secure against known threats. Further, we say that a component is Pareto-complete if within any reasonable security model, there is no Pareto improvement.

Choice

Using a Pareto-complete component is always sufficient. Designers prefer and select Pareto-complete components in order to reduce local calculation costs, risks of incomplete calculations, external costs of verification, and risks of future changes weakening the system.

If unavailable, a Pareto-secure component is also sufficient, but it carries risks that the security system may face unpredictable changes in the future that are dramatic enough to challenge the security. It also carries external costs of analysis, in that verifiers of security need to confirm that a component is indeed Pareto-secure within that system.

With more resistance, designers will choose components that trade costs and benefits in security and other non-security variables, both internally and against other components. A resistance to trade reflects a valid desire to reduce the above costs, and is sometimes referred to as *conservatism*. In cryptography especially, conservatism may indicate a bias introduced by the mixed blessing of so many good and strong cryptographic algorithms. It is an open question as to whether an economics model of interests, risks and rewards can be constructed to explain such preferences.

Applicability

Especially, in discussing a cryptographic system, we believe that we can now put a firmer meaning to the statement that a component is secure. A system using AES could be considered to be using a Pareto-complete algorithm, yet the wider cryptosystem has other components that also need some degree of attention. A system using DES may still be using an algorithm that is Pareto-secure, within its assumptions and limitations, as long as substituting another algorithm does not improve security.

This essay presents cryptographic examples, but we suggest the idea may apply to all fields of security, especially where choices and interactions are complex. Further, at least in cryptology, it would seem that Pareto-secure as a meaning is already applied in less formal settings such as product sales and the general media.

> *"Security is a chain; it's only as strong as the weakest link. Currently encryption is the strongest link we have. Everything else is worse: software, networks, people. There's absolutely no value in taking the strongest link and making it even stronger [MK-BS]."*

By refining the meaning of the term, we have the opportunity to reduce insecurity based on poorly understood product characteristics.

Limitations

The full system

Within each system, there are other components to consider, and even if all were Pareto-secure, we lack a meaningful language and theory for linking components into systems. Further, we fall short of offering a meaningful definition for security of the entire system. The first law stops us from totally eliminating all assumptions of weakness from a system; and it seems that combining components into a system simply means that at this point we can no longer pass the buck on assumptions. As a corollary to the law of no absolutely secure systems, we propose that at the systems level, all assumptions are challengeable.

Time

Nothing is forever. Even though we assert that a status of Pareto-secure means that analysis is good, this is still not an absolute.

	Before Crypto 2004		***After Crypto 2004***	
	Pareto-secure (signatures)	**Pareto-complete**	**Pareto-secure (signatures)**	**Pareto-complete**
MD5	?	No	No	No
SHA0	Yes	?	No	No
SHA1	Yes	Yes	?	No
SHA256	Yes	Yes	Yes	?

T1 - Pareto-secure status of Message Digests

This status can change overnight. In fact, at *Crypto 2004* the status of many message digest functions was shaken up overnight [Crypto] [FC]. See Table 1. We leave as an exercise the before and after Pareto-security of the various message digests.

Kaldor-Hicks

The economic theory of Pareto efficiency is limited to the assumption of *no commensurate cost*. Yet, much of security is based on risk analysis that leads to cost-benefits tradeoffs. That is, many systems explicitly impose costs in some areas to acquire benefits in other areas, which in economic terms would be an allocation closer to Kaldor-Hicks efficiency [KH].

Such choices are not compatible with the simple and direct assumptions of Pareto efficiency, and this means that the application of the Pareto theory is by no means universally applicable. Still, there are many security systems where the complexities of interaction give rise to a tendency to Pareto analysis. That is, designers try to make Pareto secure choices, even when they could achieve better risk / cost-benefit tradeoffs with superior information.

A future suggested line may be to apply a Kaldor-Hicks analogy. This would require a method of valuation of the comparative costs and benefits of alternate choices, which is beyond the easy scope of this note [LV4] [PL].

References

[VP] Vilfredo Pareto, Pareto efficiency, *Wikipedia*.

[LV1] Arjen K. Lenstra and Eric R. Verheul, Selecting Cryptographic Key Sizes. *Journal of Cryptology*, 14(4):255-293, August 2001. Also see DOC slides.

[LV2] Ibid., DOC slides.

[NIST1] Elaine Barker Cryptographic Standards and Guidelines: a Status Report NIST, 2002.

[DT] Electronic Frontier Foundation, *Cracking DES* O'Reilly, 1998

[NIST2] DES is to be withdrawn as a US-government approved algorithm in 2004. NIST, Op cit.

[TDES] Note that a closer argument can be made for 3DES versus AES, which both run at the same speed. 3DES suffers from a birthday attack as a consequence of its small block size, which AES does not suffer.

[VP2] Pareto efficiency, op cit.

[K] The issue of the key secret can be ignored for two reasons; being that both algorithms have the same assumptions, and the key secret could be considered part of the wider cryptosystem.

[AES5] Similarly to the above, an argument can be made that AES256 is more expensive in cycle time than AES128. If this marginal argument is accepted, AES256 is no longer Pareto-efficient, as swapping to AES128 will save cycles at no cost to security. We could make a case that AES128 is both Pareto-secure and Pareto-efficient, whereas AES256 is Pareto-secure but not Pareto-efficient.

[PC] This may not accord with the economics term of art (*B(Z)W*).

[NIST] NIST, OMB Guidance to Federal Agencies on Data Availability and Encryption.

[UCL] Bulens Philippe and Giry Damien Key Length Calculator, UCL Crypto Group.

[LV3] Lenstra and Verheul, op cit.

[TLS] T. Dierks and C. Allen, RFC 2246 - The TLS Protocol Version 1.0 Network Working Group

[SSL] Eric Rescorla, *SSL and TLS: Designing and Building Secure Systems*, Addison-Wesley, 2000.

[SSH] Tatu Ylönen, *The SSH (Secure Shell) Remote Login Protocol* 15 November 1995.

[SecSh] See SecSh Working Group for modern *Internet Draft* standards.

[Options] Both of these cryptoprotocols can be configured for other behaviors, but we stick here with the canonical examples.

[IG] Ian Grigg, Collected rants on SSL, 2003 - 2005.

[AA] Amir Herzberg and Ahmad Gbara, TrustBar: Protecting (even Naïve) Web Users from Spoofing and Phishing Attacks, draft paper, forthcoming.

[TC] Tyler Close, YURL - Trust Management for Humans, web article, July 2003 - July 2004. Actually, the YURL / petname proposal disposes of the certificate altogether, so that may not be appropriate in this example.

[MK-BS] Michael Kanellos quoting Bruce Schneier, " Quantum crypto firm charts way to mainstream," *CNET News.com*, 6th February 2005.

[Crypto] Crypto 2004 Santa Barbara August 2004, International Association of Cryptological Research

[FC] Ian Grigg, SHA0 is Cracked Financial Cryptography blog entry, 15th August 2004

[KH] Nicholas Kaldor and John Hicks, Kaldor-Hicks efficiency, *Wikipedia*.

[LV4] Lenstra and Verheul go some way in that direction with calculation costs, Op Cit.

[PL] Pete Lindstrom, quoting Bruce Schneier, " Security: Measuring Up," *Information Security*, Feb 2005