

Are we on the right track to achieve survivable computer network systems*

Yvo Desmedt and Mike Burmester
Department of Computer Science
Florida State University
Tallahassee, FL 32306-4530, USA
desmedt@cs.fsu.edu and burmester@cs.fsu.edu

Yongge Wang
Center for Applied Cryptographic Research
Department of Combinatorics and Optimization
University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada

Abstract

The model currently used to design survivable computation is based on the Byzantine fault model which originates from the problem of reliable *communication*. Borrowing principles from economics, artificial intelligence and control theory we argue that this approach is inappropriate to model *computer* systems. Consequently, the methods that are currently used to design survivable computer networks are questionable, from a security point of view.

Although our models can be used for information warfare to optimize an attack strategy, they can also be used to design survivable computer network systems. These models have several implications such as on PKIs (Public Key Infrastructures).

1 Introduction

Models play an important role in our modern technological society. They are the scientific foundation on which our scientific and technological progress is based. For example, better meteorological models lead to better weather forecasts. Similarly, the replacement of Newton's model for physics to the quantum model has led to solid state physics and the chip technology. Another example is modern economics, and game theory (quite recently, physicists and mathematicians were hired from universities to design models for the stock market []). It seems therefore logical to question whether we model correctly the problem of survivable computation.

*Preliminary research on this topic was supported by DARPA F30602-97-1-0205.

During the 1970's and 1980's it was generally believed that computers could be made 100% secure. It soon became obvious that this ideal situation did not correspond to reality. The possibility of computer viruses, the sloppy design of software, the lack of appropriate security policies (such as, for access control) and the size of operating systems are likely to be the most important reasons of this failure. The first model that took into account the fact that computers were not necessarily secure is the Byzantine model. It assumed that the enemy can only attack a maximum number of nodes. This model originated from the problem of reliable *communication* with malicious faults. It has since been used extensively to model secure distributed computation.

Before making the large investment that will be required to make computer systems survivable we need to question whether we are on the right track. As is common in science, we must examine whether there is a discrepancy between what the model predicts and what is observed. It is such observations of the behavior of the atom that made scientists realize that Newton's model was inadequate.

We observe that the Federal effort in the area of survivable computation has focused on what has been called the Critical Infrastructure, which indicates that depending on the *application* certain computers should be protected *more than others*. However we claim that the Byzantine model is, essentially, homogeneous in the way it deals with nodes (computers). One could argue that it allows certain nodes to be privileged. Indeed in the Byzantine model nodes on the vertex-disjoint paths do not play a role. However this is solely a consequence of the topology of the communication network and there is *no a priori* distinction between these nodes. In the rest of the text it will become clear that this is just one of the many differences between the theoretical model and the practical world. Indeed, the Byzantine model does not explain why it is more secure to have a WWW server isolated from the mainframe of an organization. Our model will.

Although one could argue that it is hard to model our complex world exactly, the sole use of heuristics may lead to conclusions that are far from correct. Did the heuristic identification of infrastructures as being the most critical truly identify the most critical ones? Even if it did, we know that the interdependencies can be very large (e.g. shutting down computers responsible for communication has far-reaching consequences). So, which computers should be protected the most? To answer such questions we need a good model.

Based on a heuristic understanding that depending on the application some computers are more important than others, we believe that we need to borrow some principles from economics. We first introduce the economics of the opponent in Section 2. We then analyze whether a communication model can be used in the context of a computer network system (see Section 3). In these discussions we reflect on the Byzantine model. By focusing on the economics from the designer's viewpoint (see Section 4), we are able to adapt our model in such a way that information security engineers and information security managers can optimize the limited resources to secure these computers that play the crucial role in our society. We discuss more general models that take time aspects into account in Section 5. We conclude by discussing the impact of our models and explain how these models could be generalized (see Section 6), by surveying briefly

how far these models have been developed and analyzed so far.

2 The economics of the enemy

It seems that the economy of the enemy is very hard to model since different opponents have different goals. In war, the enemy may try to undermine the economy and in particular the military output of a country. Terrorists however do not have the resources to use such a strategy. Therefore they may focus on very visible targets or targets that have a large impact. Finally hackers often try to demonstrate that a certain system is insecure, where a possible motivation is to boost their status in the community.

In our model we shall allow each enemy to have a *budget* that can be used towards the attack. This budget does not necessarily correspond to a dollar amount. Indeed in the case of a hacker the budget could be expressed in the number of leisure time in hours. So given such a budget the enemy has a choice of several possible attacks. Evidently if the budget is too small, the number of choices could be zero. We discuss this aspect in Section 2.1.

In the case of a warfare the enemy will optimize the choice of which computers to attack. The amateur hacker may not do such an optimization. We discuss a (simplified) model that allows such a sub-optimization in Section 2.2 and give a more complete model in Section 5.

2.1 What computers can the enemy attack

We are interested to study which subset of computers the enemy with a limited budget can attack. We are not yet interested in studying which of all achievable choices result in an optimal attack. We postpone the discussion on the optimization aspects until Sections 2.2 and 5.

For simplicity let us assume the attacker knows the structure of the computer network system (in a more complex variant, the enemy may only know a subgraph of the computer network). To each node (computer) of the computer network we assign a *cost* of penetration (see also Section 2.2). A question we address now is what is the cost of attacking two, three or in general k computers. To analyze this we consider from our viewpoint how this can be modeled in the Byzantine scenario.

2.1.1 Analysis of the Byzantine model

In the Byzantine model, one assumes that the enemy can attack at most k computers (nodes). We now analyze this from an economic viewpoint. We assume the enemy has a budget B_E .

In a first economic model one could assume that the cost of attacking machines is uniform and that the cost to attack any k machines is k times this basic price. This model however is not realistic. Indeed the cost to attack two computers running on the same platform (e.g. same operating system) is not twice the cost of attacking a single computer. So a more realistic cost model must be non-linear in the number of nodes.

Which non-linear economic models are compatible with the Byzantine model? To analyze this we view the computers in a network as a set V . To each subset $S \subseteq V$ of computers we let correspond a cost $c_{S,E}$, where E indicates that this is the cost of the enemy. In this general model the Byzantine model makes sense provided that:

- for each subset S of (at most) k computers the cost $c_{S,E}$ to attack those computers is less than the budget B_E of the enemy,
- for each subset S' of $k + 1$ computers (or more) the cost $c_{S',E}$ is larger than B_E , the available budget of the enemy.

We call these conditions the *Byzantine cost assumption*. We now argue that such an assumption is often unrealistic.

2.1.2 A more realistic model

If the Byzantine model were realistic the cost of attacking k computers on very different platforms would have to be less than attacking $k + 1$ computers using the same platform. It is obvious that this is unrealistic. We therefore suggest a different model.

In our model we consider the network graph $G = (V, E)$, where V is the set of nodes and E the set of links (edges). We allow the enemy to attack both nodes as well as links. We propose that to each subset S of computers/links (or nodes and edges) correspond a cost $c_{S,E}$. A subset of computers/links can be attacked by the enemy if and only if $c_{S,E} \leq B_E$. This defines a set Γ_G whose elements are subsets S of computers/nodes in $V \cup E$ that can be attacked by the enemy. This set Γ_G is a subset of the power set of $V \cup E$ (denoted by $\mathcal{P}(V \cup E)$), and is called the *access structure* of the enemy. If the enemy is remote, it may be impossible to attack the links. Then the access structure may consist entirely of nodes, say Γ_V . For example, if $V = \{1, 2, 3\}$ is the set of computers, Γ_V could be $\{\{1, 2\}, \{2, 3\}, \{1, 2, 3\}\}$.

2.1.3 Difficulty of estimating these costs

Although this model is quite general, a problem with such a detailed numeric approach is that it seems to introduce the problem of how to estimate the cost of attacking a subset of computers. However, we argue that it opens the door for alternatives that are better than the Byzantine one.

As an example of an alternative we consider a model in which the cost to attack many computers running the same platform is identical to the cost of attacking a single computer running the same platform. We call such an approach a *platform-oriented* model. A further simplification is to assume that the cost to attack a Windows 2000 operating system is identical to attack a Linux system. Even if this *homogeneous platform-oriented* approach is likely to be incorrect, it may produce models that are much more realistic than the Byzantine one.

Before using such a simplified model we need to study their *stability*. This means see how much the results are affected when changing the cost of attacking different

platforms. Provided these changes do not result in dramatically different conclusions, these models can be used. We borrow the stability approach from control theory.

2.2 Optimizing the attack

We now know which computers in the system the enemy can attack with his/her limited budget. The set Γ_G indicates this. If the cardinality of this set is larger than one, the enemy has a choice of which computers to attack. We now wonder how the enemy can optimize his/her choice.

The idea of just choosing as many computers as possible (formally the largest subset of V in Γ_V) may be appropriate for a hacker, but does not necessarily lead to the largest (economic or military) damage. As we will see in Sections 3 and 5 this question involves a number of issues. We only discuss one aspect here.

A computer may be involved in several applications. Each application could be viewed as a unit application. When considering a unit of an application many computers may be involved. Indeed when considering the problem of communication more than one router may be involved in one communication session. From this viewpoint one can view an application as *flowing* through network computer system. Depending on the application, the economic importance of a unit of flow may be different. Indeed compare a successful denial of service attack against an e-mailed love letter and one against a military communication from the commander in chief. Although the emotional and personal loss may be very large in the first case, the military loss is likely to be much larger in the second case. So to each unit of flow corresponds an *impact factor*. The unit in which this impact factor is measured is not necessarily expressed in dollars, but other units could be used.

When the enemy has taken over a subset S the enemy can reduce the flow by f_S .

2.2.1 Analysis of the Byzantine model

In the Byzantine model the goal of the enemy is to have the receiver accept a message the sender did not sent, or in general to stop the communication. While a hacker, who does not optimize the attack would go for taking over any k nodes, a more sophisticated enemy would only go for k nodes each on a path disjoint of the other paths. This is a direct consequence of the optimization. This illustrates the difference between the access structure and the optimized attack.

The claim that the enemy can succeed if the network is not sufficiently connected is a direct consequence of the following assumptions:

- Byzantine cost assumption (see Section 2.1.1),
- the assumption that each application has the same impact. We call this assumption the *Byzantine impact assumption*.

2.2.2 A more realistic model

If the Byzantine model were realistic, then a consequence of the Byzantine impact assumption would be that under this model, the enemy will have succeeded if just one reliable communication were prevented, regardless of the impact factor of this communication.

It is more realistic to set an impact threshold and state that the enemy has failed if the total impact reduction of the flow due to the enemy's action is below the threshold.

Note that as for cost we do *not* necessarily assume a *linear* impact factor. We allow for a non-linear one, so that the impact factor of two transactions may be less (or even larger) than the sum of the impact factor of each.

2.2.3 Different goals of different enemies

The enemy may have different goals in mind and the optimization by the enemy will depend on these goals. In an information warfare scenario the enemy may go for a “doomsday” strategy, e.g. shut down enough water distribution to starve the population of a country. So there is a critical threshold in flow reduction F_T . So if $G = (V, E)$ is the given computer network system, the enemy can win if there exists a subset S of $V \cup E$ such that:

1. the cost $c_{S,E}$ associated to take over the set S is less or equal to B_E (in other words $S \in \Gamma_G$), and
2. $f_S > F_T$, i.e. the impact factor of having taken over the set S is larger than the critical one.

Such a goal may be too expensive for a terrorist. The terrorist with a lower budget can only maximize the reduction of the flow.

Note that this water distribution example is more complex as will become clear after having read Sections 5 and 3. We now argue that a communication model is inappropriate to tackle such issues.

3 Communication model versus computation

3.1 The problems with the communication model

When using the Byzantine communication model as the basis of secure distributed computation, each computer is viewed as a general purpose computer. We agree that this model has lead to some interesting results. However, we claim that the approach is very limited and that it is unrealistic to model in particular existing network computer systems. We now explain our viewpoint.

In a distributed computation, such as a travel reservation system, several operations may take place (airline reservations, hotel reservations, etc). A transaction can only be completed if *all* sub-transactions have been performed. This aspect is well known

in the management of manufacturing industries. A car factory relies on the supply of its components (body parts, engines, tires, etc). Such systems can be modeled by a PERT (Program Evaluation and Review Technique) directed graph. With this graph an output can only be produced if *all* vertices in the graph have produced their output. A similar approach seems reasonable when dealing with computations that can only be executed provided other computations on other computers have. This can clearly *not* be modeled by the communication graph. However the PERT graph cannot deal with redundancy to achieve survivability.

3.2 AND/OR graphs as a model for distributed computation

To model both computation that requires outputs resulting from multiple other computation and redundancy, we have suggested to borrow the AND/OR graphs from artificial intelligence [2, 10]. The computer network systems is represented by a graph. If the computation in a computer depends on outputs from multiple other ones, then the vertex is labeled with an AND and else with an OR. The OR indicates that the input could come from the one node or the other. An OR node could correspond with choosing one of the inputs (e.g. by a vote or another strategy) or by taking the average (if applicable), etc.

If a computer is involved in different applications then each application in the computer is modeled as a node (or as multiple nodes). The AND/OR directed graph becomes larger. This also introduces natural dependencies in the sense that if a hacker can shut down or take over a computer then the nodes corresponding with applications run in that computer have all been taken over.

4 The economics of the designer

The economics of the enemy, using the AND/OR graph model, is suited to analyze the strength of an *existing given computer network system*. When designing a computer network system the designer knows:

the design budget B_D . The cost includes the basic cost of the hardware (i.e. the cost of computers, and links), the basic cost of the software (e.g. the cost of having different operating systems, different applications, etc.) and the cost for the hardware/software security.

a minimal required total capacity C_D when the system is not under attack. Each hardware/software unit will provide a certain maximal capacity. A flow through a node or through a link must be lower than the capacity. Different rules have been proposed to restrict the flow of an outgoing edge as a function of the incoming ones. The classical flow rule used in hydrolics is that the output flow is equal to the sum of the incoming flows. However, as pointed out by Martelli and Montanari [8, 9] data can be copied; they suggested an alternative model called “additive.”

a maximum flow reduction F_T the system can tolerate, and

the enemy's budget B_E .

Those involved in setting up computer network system know that this model is too simple. Other limiting factors are the cost of maintenance, the user friendliness of the system, etc. As long as we are aware of its limitations we can adjust it if necessary.

Whether the cost is linear depends of whether we are dealing with a small scale effort or a large scale effort. In a large scale hardware effort the cost to design a hardware platform makes the cost non-linear. A similar conclusion applies to medium scale. Indeed, when buying at such a medium scale one usually receives reductions, due to the non-linear aspects of the economy. The cost of software is clearly non-linear.

Due to our impact factor measure, it is clear that the idea of making all computers have the same security may be uneconomic. We therefore view the cost of making a computer more secure as an additional cost.

While for the enemy the main unit of target is the computer (in general the computer or link), the *application should be the main unit for the designer*. One of the questions to be addressed by the designer is which applications to conglomerate on one computer instead of using multiple servers. By using such a conglomeration, the cost clearly goes down, but it makes the computers more vulnerable.

The designer needs to find a graph $G = (V, E)$ of computers and links such that:

1. the cost(G) of the system is less or equal to the available budget B_D ,
2. the total flow possible is at least the required capacity, i.e. $f_{(V \cup E)} \geq C_D$,
3. the enemy cannot win, i.e. for all $S \subseteq (V \cup E)$ we have that:

(a) $c_{S,E} > B_E$, or

(b) $f_S \leq F_T$.

If such a system cannot be designed then the enemy has won, otherwise the enemy can only perform a terrorist type of attack.

Since we

- have allowed the cost of the enemy, the cost to build, and the flow, all to be non-linear,
- did not have a relation between the cost of setting up a computer and the cost of attacking a subset,

it is hard to say anything more. In Section 6.2 we survey which special cases of our model have been studied so far.

5 Control theory variants of the models

To anybody familiar with control theory it is obvious that our models are too simplistic since they do not take time aspects into account. Such aspect are important due to the existence of buffers, such as fuel supplies and emergency stock, water reserves and food stock. Other time related aspects of an attack/defense include:

- The time between the impact of an attack and the moment of detection of the attack.
- The time to recover from an attack after it has been detected.
- For truly critical systems, there is a time of no return. For example, within a few days people die without water to drink.

To survive we obviously must have that: (the time to repair the system) + (the time to detect an attack has taken place) \leq (the time of no return) + (the time the stock will last).

The enemy may be able to increase/decrease some of these time factors. Indeed, the time the stock will be usefull can be reduced since many of these stocks have computerized controls making them vulnerable.

6 Impact and what has been studied so far

6.1 Impact

One of the main reasons why the Byzantine model has not been adopted universally is the cost of redundancy in computers and links. If a node in a city is far away from sufficiently many other heavily populated cities, the cost of putting these disjoint links is just prohibitive. Moreover, viruses and other automated attacks make it hard to estimate how many computers the enemy can take over.

It is obvious that our “cost for the enemy” approach clearly demonstrates that the required redundancy may be much less than predicted by the Byzantine model. Indeed take the hypothetical case when the cost to take over a single node/link is prohibitive; then there is no need for redundancy. While this is extreme, it demonstrates that less redundancy may be required when there is more protection.

Our non-linear approach related to cost, and in particular the platform-oriented one (see Section 2.1.3), allows a much better estimate of the power of the adversary. It has been demonstrated in [4] that although redundancy is still required, the need for disjoint paths is no longer there, again reducing the cost of defending.

Finally our AND/OR model allows us to model better existing systems and to study how to adapt such systems to make them more secure at a cost less than required if one would have to use the Byzantine model.

6.2 What has been studied?

We already see confirmation of some of the impacts we claim our model will have. Indeed limited versions of our model have been studied and confirm our impact claims. These preliminary studies have inspired this paper. None of these studies is as general as the one we suggest here. However, the ones we now survey have been studied in much more details than what we have suggested here.

The idea of just replacing the Byzantine model by an AND/OR graph has been studied in [2, 10].

The flow approach to study how the enemy can optimize the attack strategy has been analyzed in [5]. In this study the impact factor was viewed as linear and different applications had identical impact.

Some studies demonstrated that the Byzantine model is inappropriate to model communication:

- when one is not dealing with point-to-point networks, but with ethernet and other broadcast channels. This has been studied e.g. in [7, 6, 11].
- homogeneous platform-oriented (see Section 2.1.3 for a definition) point-to-point communication networks [4].

Finally it has been pointed out that in a dynamic network the network may be unknown [1]. A partial solution to this problem has been studied in [3].

7 A call for an interdisciplinary effort

The goals of this paper were to demonstrate that:

- There are better models than the Byzantine one to address survivable computation. The Byzantine model makes unrealistic assumptions about the cost to attack multiple computers, is not suited to model computation networked systems and does not discriminate between the impact factor different applications have to our economy.
- Alternative models are worth studying.
- Other studies have started to point out the limitations of the Byzantine model.

It is obvious that the heuristics used to design survivable computation do not follow the Byzantine model. However, there is no guarantee that these heuristics protect us well and are efficient. To avoid such possibilities one should not be surprised that other research areas view models as very important. Indeed, billions of dollars have been spent in physics to design and check the “standard model of elementary particles.”

To allow us to optimize or sub-optimize our design of survivable computation, we believe that economists, control engineers, information security engineers, war strategists and other experts should join in an effort to better understand how survivable computer networks should be designed.

References

- [1] M. Burmester, Y. Desmedt, and G. Kabatianskii. Trust and security: A new look at the Byzantine generals problem. In R. N. Wright and P. G. Neumann, editors, *Network Threats, DIMACS, Series in Discrete Mathematics and Theoretical Computer Science, December 2-4, 1996, vol. 38*. AMS, 1998.

- [2] M. Burmester, Y. Desmedt, and Y. Wang. Using approximation hardness to achieve dependable computation. In M. Luby, J. Rolim, and M. Serna, editors, *Randomization and Approximation Techniques in Computer Science, Proceedings (Lecture Notes in Computer Science 1518)*, pp. 172–186. Springer-Verlag, October, 8–10 1998. Barcelona, Spain.
- [3] M. Burmester and Y. Desmedt. Secure communication in an unknown network using certificates. In K. Y. Lam, E. Okamoto, and C. Xing, editors, *Advances in Cryptology — Asiacrypt '99, Proceedings (Lecture Notes in Computer Science 1716)*, pp. 274–287. Springer-Verlag, November, 14–18 1999. Singapore.
- [4] Y. Desmedt, Y. Wang, and M. Burmester. Modeling platform depend attacks in communication systems. In preparation.
- [5] Y. Desmedt and Y. Wang. Maximum flows & critical vertices in AND/OR graphs. In *INFORMS (INstitute For Operations Research and the Management Sciences)*, p. 8, Baltimore, Maryland, May 2–5, 1999. INFORMS. <http://www.informs.org/Conf/Cincinnati99//TALKS/SA34.html> Cincinnati, Ohio.
- [6] M. Franklin and R. Wright. Secure communication in minimal connectivity models. In K. Nyberg, editor, *Advances in Cryptology — Eurocrypt '98, Proceedings (Lecture Notes in Computer Science 1403)*, pp. 346–360. Springer-Verlag, 1998. Espoo, Finland, May 31–June 4.
- [7] M. Franklin and M. Yung. Communication complexity of secure computation. In *Proceedings of the twenty fourth annual ACM Symp. Theory of Computing, STOC*, pp. 699–710, 1992.
- [8] A. Martelli and U. Montanari. Additive AND/OR graphs. In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 1–11, Morgan Kaufmann Publishers, Inc., 1973.
- [9] A. Martelli and U. Montanari. Optimizing decision trees through heuristically guided search. *Comm. of the Assoc. Comput. Mach.*, **21**(12):1025–1039, 1978.
- [10] Y. Wang, Y. Desmedt, and M. Burmester. Models for dependable computation with multiple inputs and some hardness results. *Fundamenta Informaticae*, 42(1), pp. 61–73, March 2000.
- [11] Y. Wang and Y. Desmedt. Secure communication in broadcast channels. In J. Stern, editor, *Advances in Cryptology — Eurocrypt '99, Proceedings (Lecture Notes in Computer Science 1592)*, pp. 446–458. Springer-Verlag, 1999. Prague, Czech Republic, May 2–6.