Optimal Voting Strategy Against Random and Targeted Attacks

Li Wang*, Illinois Institute of Technology, US
Zheng Li, Illinois Institute of Technology, US
Shangping Ren, Illinois Institute of Technology, US
Kevin Kwiat, Air Force Research Laboratory, US

ABSTRACT

Replication and value selection through voting are commonly used approaches to tolerating naturally caused failures. Without considering intentionally introduced failures, such as failures caused by attacks, having more replication or residency often makes the system more reliable. However, when both the reliability of individual replicas and the existence of attackers are taken into consideration, the number of replicas that participate in a voting process has significant impact on system reliability. In this paper, we study the problem of deciding the optimal number of participating voters that maximizes the reliability of voting results under two different types of attacks, i.e., random attack and targeted attack, and develop algorithms to find the optimal voting strategy. A set of experiments are performed to illustrate how the optimal voting strategy varies under different system settings and how the number of voting participants affects the system's reliability.

Keywords: Voting Strategy; System Reliability; Random Attack; Targeted Attack; Attacker-defender Problem

1. INTRODUCTION

Ensuring a complex system's reliability has been a challenging issue for decades. One of the main reasons is that it is very difficult and costly, if not impossible, to have all system components reliable (Kwiat, Taylor, Zwicker, Hill, Wetzonis, & Ren 2010). What further exacerbates the challenge is the ever growing number of cyber attacks both in severity and in volume. Replication and voting are often used hand-in-hand to safe-guard a system from failures and ensure the system's reliability (Trivedi, 2002; Siewiorek & Swarz, 1998). For example, internet content providers (e.g., Google) often deploy multiple clusters world-wide not only to achieve fault tolerance but also to improve system performance. Another example of commonly used replication is to geographically distribute clusters to facilitate the recovery from localized disasters (e.g., hurricanes and earthquakes).

There are many different voting algorithms in literature. For instance, unanimity voting (Latif-Shabgahi, Bass, & Bennett, 2004) generates a result when all the replicas are in agreement. This type of voting algorithm is used when reaching an agreement by all replicas is necessary. Clearly, the unanimity voting does not tolerate any replica failures. The majority voting algorithm (Thomas, 1979), on the other hand, takes the majority as its final value. A less restrictive, or more general form of majority voting is plurality voting (Latif-Shabgahi et al., 2004). It requires *m-out-of-n* replicas to agree on the same result, where *m* can be less than a strict majority of n, i.e., $m < \left\lceil \frac{n+1}{2} \right\rceil$.

In our early work (Kwiat et al., 2010), a blending of fault and attack tolerance of three majority voting algorithms: Majority Rule (MR), Random Dictator (RD), and Random Troika (RT), is studied. For MR, it needs all the replicas to vote, while RD randomly chooses one replica's value as its result. Although RT also belongs to the majority voting family, the selection of the Troika is rather random, and the final result is decided by the majority of the three chosen replicas. The study concludes that neither of these three voting algorithms is always superior to the others when the reliability of replicas and the number of uncompromised replicas are different (Kwiat et al., 2010).

In this paper, we study the problem of deciding an optimal number of participating voters that maximizes the reliability of voting results under two different types of attacks, i.e., random attack and targeted attack and develop algorithms to find the optimal voting strategy. For random attack, the probability of each individual replica being compromised is the same, while with targeted attack, the attacker selects a subset of replicas as its targets. We assume that attackers can only attack a system when the system is in operation.

The rest of the paper is organized as follows. Section 2 discusses related work. In Section 3, we formally define the problem of deciding an optimal number of participating voters that maximizes the reliability of voting results under random and targeted attacks, respectively, and provide solutions to the problem in Section 4. The experimental results are shown and discussed in Section 5. Finally, we conclude and point out future work in Section 6.

2. RELATED WORK

The analysis of attacker-defender problems have been studied from different perspectives. For instance, in (Vicki & Bier, 2002; Bier, Nagaraj, & Abhichandani, 2005), Bier et al. study optimal defenses against intentional threats for both series and parallel systems, and illustrate how the optimal allocation of defensive investments changes with the structure of the system and the adversary's goals and constraints. Yalaoui et. al. consider redundant components in series systems and proposed a dynamic programming method to calculate the minimum cost required for a system to satisfy the minimum reliability requirement (Yalaoui, Chatelet, & Chu, 2005).

Levitin and Hausken have done substantial work regarding how to improve the reliability of a system subject to attacks. To be more specific, in (Hausken, 2008; Levitin & Hausken, 2009a; Levitin & Hausken, 2009b; Levitin & Hausken, 2009c), they consider series and parallel systems, series systems of parallel subsystems, and parallel systems of series subsystems, and analyze system reliability when systems have or do not have budget constraints. Some approaches considering protection and redundancy techniques to reduce the expected damages caused by the attacks are discussed in (Levitin & Hausken, 2008). In (Levitin & Hausken, 2009d), they further analyze how to allocate resources between deploying false targets and enhancing object protection when protecting a single object. They conclude that the optimal number of false targets does not depend on the attacker's resources but only depends on the relative target cost. While in (Levitin & Hausken, 2009e), they propose three defense approaches (i.e., false targets, protection, and replication) to minimize system damage when both the defender and the attacker have limited resources, and illustrate how the defender and attacker choose their strategies when the contest intensity changes.

Our work differs from those discussed above in two aspects. First, in our work, the reliability of the system does not depend on the system's structure nor on the number of uncompromised replicas, but instead it is decided by a nonempty subset of replicas which form the voting replicas. This is in contrast to the work discussed above where the system reliability either depends on the system structure or the amount of uncompromised replicas. Second, the defender model is different. In our work, the defender has no defense resources, but can only choose different number of voters to maximize the expected number of surviving clusters in the system. While in the work discussed above, defenders are given a fixed amount of defensive resources that are distributed among different defense approaches, and they decide how to optimally distribute the defensive resources to maximize system reliability.

In (Hardekopf, Kwiat, & Upadhyaya, 2001), Hardekopf et al. propose a decentralized voting algorithm that improves system dependability and protects the system from faults and hostile attacks. Their work mainly focuses on the voting protocol, while in our work, our goal is to decide the optimal number of voters.

Tong et al. show how to choose optimal weight assignments for the majority voting strategy in the system and also propose new effective vote assignment algorithms which aim to maximize the system reliability (Tong & Kain, 1991). Davcev proposes a dynamic weighted voting scheme for consistency and recovery control of replicated files in distributed systems (Davcev, 1989).

The main difference between our work and those discussed above is that in our system model, the number of voters can change under different conditions and up to N_r (i.e., the total number of replicas in a cluster) replicas can be used to form the voters. However, the work in (Tong & Kain, 1991; Davcev, 1989) only consider fixed number of voters.

Our earlier work focused on using an information hiding approach to prevent the attackers from quickly identifying the location of critical components in the system (Wang, Leiferman, Ren, Kwiat, & Li, 2010) and deciding optimal resource allocation for improving system reliability under random attack (Wang, Ren, Yue, & Kwiat, 2010, 2011). The current work differs from the previous work in two aspects. First, the system models are different. In this paper, the system reliability depends on the reliability of a set of selected voting replicas and whether they are compromised or not, while in (Wang et al., 2010, 2011), the system reliability is the probability that all critical components survive the attacks. Second, the protection approaches are different. The previous work considers creating camouflaged components, or enhancing component protection, etc.), while the work presented in this paper consider the number of voting replicas in a voting process.

3. Preliminaries and Problem Formulation

Before presenting the formal description of the system model and its assumptions, we first introduce the notations to be used throughout the paper.

C number of clusters available in the system

S	number of surviving clusters
A	number of clusters being attacked
C_i	the *i*th cluster
N_r	number of replicas in a cluster
N_v	number of voting replicas in a cluster
N_f	number of compromised replicas in an attacked cluster
N_v^u	number of uncompromised voting replicas in each attacked clusters
N_t	total number of compromised replicas in the system
r	replica's reliability
p	the probability of a replica being compromised under random attack
T	system operation time
$\vec{P}_d$	defense strategy selection vector
$\vec{P}_a$	attack strategy selection vector
S_d	number of defense strategies
S_a	number of attack strategies

Preliminaries and Assumptions

We assume a system consists of C independent clusters, and each of them is composed of N_r diverse replicas. By diverse, we mean the replicas are functionally equivalent, but their implementations may vary. The reliability of each individual uncompromised replica is r. For each cluster, its reliability is the probability that a correct final result is obtained through a voting process.

It is worth pointing out that the problem modeling and the algorithmic solutions developed in the paper reply on an overlay structure based on clusters. The overlay structure permits the ad hoc instantiation of standard network structures such as a star or a mesh when such structures are deemed optimal for system performance; however, as attack tolerance is our governing concern we restrict our discussion to cluster formation for the purpose of voting.

We further assume that the system is to operate for T time units, and it is only during the system operation time that the attacker can make attempts to compromise the system. For an attacker, the time needed to compromise a system replica's depends on the replicas vulnerabilities and the attacker's skill level (McQueen, Boyer, Flynn, & Beitel, 2006). We assume that the attacker can successfully compromise N_t replicas within that T time units.

When fighting against external attacks, the system defender cannot guarantee the attacker does not know the system information. As system's reliability is evaluated under the worst case condition (Dominguez-Garcia & Grainger, 2008), we assume that the attacker knows the static information about the system, i.e., the number of clusters C, the number of replicas in each cluster N_r, and the replicas' reliability r in the system. We consider that knowing such static information is an advantage for the attacker. However, the dynamic information, such as how many replicas are chosen to participate in a voting process within a cluster and who are the voting/non-voting replicas, which can adjust at run time, is unknown to attackers. For the defender, he/she does not know which clusters or replicas are currently being attacked, his/her option is to decide the number of voters for each cluster. Figure 1 gives a snapshot view of a cluster.

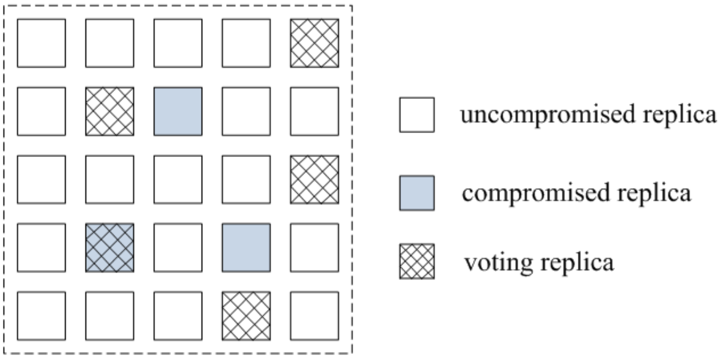

Figure 1. Snapshot view of a cluster

Cluster Reliability without Compromised Replicas

When the cluster is not under attack, none of the replicas in the cluster is compromised. Assume the defender chooses $N_v (1 \le N_v \le N_r)$ replicas to vote, where N_r refers to the number of replicas in each cluster. The final result is decided by the majority of the results from the N_v chosen replicas. In order to have a correct result, at least $\left\lceil \frac{N_v + 1}{2} \right\rceil$ replicas must vote correctly. We use $\theta(N_v)$ to refer to the probability that a correct result is achieved when N_v replicas are selected to vote and none of them is compromised, and we have

$$\theta(N_v) = \sum_{i=\left\lceil \frac{N_v+1}{2} \right\rceil}^{N_v} \binom{N_v}{i} r^i (1-r)^{N_v - i} \tag{1}$$

where N_v is the number of voting replicas, r is replica's reliability.

Cluster Reliability with Compromised Replicas

When a cluster is under attack, we assume N_f out of N_v replicas are compromised, and the defender chooses N_v replicas to vote. Let N_v^u represent the number of uncompromised voting replicas, we have max $\max\left\{0, N_v - N_f\right\} \le N_v^u \le \min\left\{N_v, N_r - N_f\right\}$,The probability, $\gamma(N_v^u, N_v)$, that when choosing N_v replicas from a cluster of size N_r with N_f compromised replicas, N_v^u out of N_v voting replicas are uncompromised is

$$\gamma(N_v^u, N_v) = \frac{\binom{N_r - N_f}{N_v^u} \times \binom{N_f}{N_v - N_v^u}}{\binom{N_r}{N_v}} \qquad (2\ )$$

Assume N_j out of N_v^u uncompromised replicas vote correctly, in order to produce a correct result, we must have $N_j \geq \left\lceil \frac{N_v + 1}{2} \right\rceil$. We use $\varphi(N_f, N_v)$ to denote the probability that the cluster is able to produce a correct result under N_f compromised replicas when N_v replicas are involved in the majority voting, and we have

$$\varphi(N_f, N_v) = \sum_{N_v^u = lb}^{ub} \left(\gamma(N_v^u, N_v) \times \sum_{N_j = \left\lceil \frac{N_v + 1}{2} \right\rceil}^{N_v^u} \binom{N_v^u}{N_j} r^{N_j} (1 - r)^{N_v^u - N_j} \right) \qquad (3)$$

where $lb = \max\{0, N_v - N_f\}$ and $ub = \min\{N_v, N_r - N_f\}$.

The Expected Number of Surviving Clusters under Random Attack

When the system is under random attack, for each replica, the probability of being compromised is the same. We use N_f to denote the number of compromised replicas in an individual cluster c_i, and we have $\max\{0, N_t - (C-1) \times N_r\} \leq N_f \leq \min\{N_t, N_r\}$, and the probability, $\varphi(N_f)$, that N_f replicas are compromised in cluster c_i is

$$\phi(N_f) = \frac{\binom{N_r}{N_f} \times \binom{N_r \times (C-1)}{N_t - N_f}}{\genfrac{}{}{0pt}{}{N_r \times C}{N_t}} \qquad (4)$$

Assume the number of replicas which participate in the voting process in cluster c_i is N_v, and the reliability of cluster c_i is $\omega(N_v)$, and we have

$$\omega(\mathrm{N}_v) = \sum_{N_f = \max\{0, N_t - (C-1) \times N_r\}}^{\min\{N_t, N_r\}} \left(\phi(N_f) \times \varphi(N_v, N_f) \right) \qquad (5)$$

As the number of voting replicas i.e., N_v , in each cluster is the same, therefore, when the system is under random attack, the number of surviving clusters is

$$S(N_v) = C \times \omega(\mathrm{N}_v) = C \times \sum_{N_f = \max\{0, N_t - (C-1) \times N_r\}}^{\min\{N_t, N_r\}} \left(\phi(N_f) \times \varphi(N_v, N_f) \right) \qquad (6)$$

The Expected Number of Surviving Clusters under Targeted Attack

When the system is under targeted attack, the attacker's strategic decision is to decide how many clusters to attack. Assume the attacker chooses $A(1 \le A \le C)$ clusters to attack and evenly distributes the effort among the clusters, which indicates that there are at least $N_f = \left\lfloor \frac{N_t}{A} \right\rfloor$ compromised replicas in the *A* chosen clusters, and no compromised components in the remaining *C-A* clusters. When N_t is not evenly divided by *A*, say that the division ends with remainder *q*, that is $q = N_t \bmod A$. Under this scenario, the attacker will compromise $\left\lfloor \frac{N_t}{A} \right\rfloor + 1$ replicas in the first *q* out of *A* clusters, and compromise $\left\lfloor \frac{N_t}{A} \right\rfloor$ replicas in the remaining $A - q$ clusters.

Assume the defender sets the number of participating voters for each cluster is $N_v(1 \le N_v \le N_r)$. Let $S(N_v, A)$ denote the expected number of clusters which are able to produce a correct result, based on the analysis above, we have

$$S(N_v, A) = (C - A) \times \theta(N_v) + (A - q) \times \varphi(N_f, N_v) + q \times \varphi(N_f + 1, N_v) \qquad (7)$$

where $N_f = \left\lfloor \frac{N_t}{A} \right\rfloor$, and $q = N_t \bmod A$.

The problem we are to solve is to decide an optimal strategy, i.e., N_v, to maximize $S(N_v, A)$ given that for any N_v the attacker chooses A that minimizes the value $S(N_v, A)$. That is

$$\max_{N_v} \min_{A} S(N_v, A) \qquad (8)$$

4. DETERMINING THE OPTIMAL VOTING STRATEGY UNDER RANDOM AND TARGETED ATTACK

In this section, we aim to find the optimal voting strategy for the system when it is under random and targeted attack, respectively.

Optimal Voting Strategy under Random Attack

When the system is under random attack, according to Section 3, we know the expected number of surviving clusters is

$$S(N_v) = C \times \sum_{N_f=\max\{0,N_t-(C-1)\times N_r\}}^{\min\{N_t,N_r\}} \left(\phi(N_f)\times\varphi(N_v,N_f)\right) \tag{9}$$

In (9), N_v is the only variable, and the range of N_v is between 1 and N_r. Therefore, in order to determine the optimal voting strategy, i.e., N_v, we can simply enumerate all the possible cases, and choose the one with maximized expected number of surviving clusters. Algorithm 1 shows the procedure of finding the value of N_v.

Algorithm 1 DECIDE THE OPTIMAL VOTING STRATEGY UNDER RANDOM ATTACK

Input: Number of clusters *C*, number of components in each cluster N_r, number of compromised replicas in the system N_t, and replicas' reliability *r*.

Output: Maximum number of surviving clusters $S_{\max}$, voting strategy N_v.

```
$S_{\max} \leftarrow -1$; $N_v \leftarrow -1$
for $N_v^{'} \leftarrow 1$ to $N_r$ do
    $S_{temp} \leftarrow S(N_v^{'})$
    if $S_{\max} < S_{temp}$ then
        $S_{\max} \leftarrow S_{temp}$
        $N_v \leftarrow N_v^{'}$
    end if
end for
return $S_{\max}, N_v$
```

Theorem 1: Algorithm 1 obtains the maximum expected number of surviving clusters when the system is under random attack.

Proof: We prove the theorem by contradiction. If there exists a higher expected number of surviving clusters, the corresponding voting strategy must be one of the N_r strategies.

In Algorithm 1, we compare the expected number of surviving clusters under each voting strategy and choose the largest one. Therefore, the assumption does not hold.

Complexity Analysis: From (9), we know that the time complexity of calculating $\varphi(N_f, N_v)$ is $O((N_r)^3)$. When performing the comparison, we need to enumerate all the cases, i.e., the total iteration is N_r. Therefore, the complexity of the algorithm is $O((N_r)^4)$.

Optimal Voting Strategy under Targeted Attack

When the system is under targeted attack, the attacker's goal is to strategically decide the number of clusters to attack, i.e., *A*, so that the total number of surviving clusters is minimized.

As the total number of replicas the attacker can successfully compromise under given time *T* is N_t, and these compromised replicas can be scattered among at least $\left\lceil \frac{N_t}{N_r} \right\rceil$ clusters. As there are total *C* clusters in the system, hence the number of clusters being attacked, *A*, is in the range of $\left[\left\lceil \frac{N_t}{N_r} \right\rceil, \min\{C, N_t\} \right]$. Therefore, the total number of possible attack strategies is $S_a = \min\{C, N_t\} - \left\lceil \frac{N_t}{N_r} \right\rceil + 1$, and the *j*th attack strategy is to attack $A = \left\lceil \frac{N_t}{N_r} \right\rceil + j - 1$ clusters.

For the defender, as he/she can choose any number of participants to perform majority voting in the cluster with size N_r, hence the number of possible voting strategies is $S_d = N_r$, and the *i*th voting strategy is to select $N_v = i$ replicas to vote. Once the strategies of both defender and attacker are decided, the expected number of surviving clusters $S(N_v, A)$, i.e., the clusters which produce a correct result under attack, can be calculated by using (7). Therefore, we define a matrix $M = (s_{i,j})_{S_d \times S_a}$ to record the number of surviving clusters under each possible defense and attack strategies, where $s_{i,j}$ refers to the number of surviving clusters when the defender chooses *i*th voting strategy and the attacker chooses *j*th attack strategy. In other words, $s_{i,j} = S(i, \left\lceil \frac{N_t}{N_r} \right\rceil + j - 1)$.

Clearly, for each voting strategy N_v, depending on which attack strategy, i.e., *A*, is taken by an attacker, the number of surviving clusters can vary. We introduce two vectors, i.e., $\vec{P_d} = [x_1, ..., x_{S_d}]^T$ and $\vec{P_a} = [y_1, ..., y_{S_a}]^T$, where x_i and y_i denote the probability that *i*th voting strategy is chosen by a defender and the *j*th attack strategy is chosen by an attacker, respectively.

Based on the voting strategy selection vector $\vec{P_d}$ and the attack strategy selection vector $\vec{P_a}$, the expected number of surviving clusters is given by (10).

$$S(\vec{P}_d, \vec{P}_a) = \vec{P}_d^T \, M \, \vec{P}_a = \sum_{i=1}^{S_d} \sum_{j=1}^{S_a} x_i s_{i,j} y_j \tag{10}$$

In this optimal voting strategy problem, the gain of the defender is the loss of the attacker, and vice versa. A game in which one player wins what the other player loses is called a two-person zero-sum game (Raghavan, 1994; Gass, 1984). Therefore, the optimal voting strategy problem can be mapped to the two-person zero-sum game problem.

It has been proved that the two-person zero-sum game is equivalent to the linear programming problem (Kuhn & Tucker, 1950). Therefore, in order to find out the optimal voting strategy, we transform the optimal voting strategy problem into a linear programming problem. In our work, we simply follow the steps proposed in (Zafra, 2011; Vanderbei, 2001) to perform the transformation. Here, we first present the steps in Algorithm 2 and provide the detailed explanation and discussion afterward.

Algorithm 2 DETERMINE THE OPTIMAL VOTING STRATEGY UNDER TARGETED ATTACK

Input:

C: total number of clusters in the system;
N_r: total number of replicas in each cluster;
r: the reliability of each replica;
N_t: total number of replicas the attacker can compromise;

Output:

$\vec{P}_d$: the defender’s optimal voting strategy;

$\vec{P}_a$: the attacker’s optimal attack strategy;

S: the expected number of surviving clusters;

1: $S_d \leftarrow N_r$

2: $S_a \leftarrow \min\{C, N_t\} - \left\lceil \frac{N_t}{N_r} \right\rceil + 1$

3: **Create** a $S_d \times S_a$ matrix $M = (s_{i,j})_{S_d \times S_a}$, where $s_{i,j}$ denotes the expected number of surviving clusters if defender chooses its *i*th voting strategy and attacker chooses its *j*th attack strategy.

4: **Create** a S_d dimensional vector $\vec{P}_d = [x_1, \ldots, x_{S_d}]^T$ and a S_a dimensional vector $\vec{P}_a = [y_1, \ldots, y_{S_a}]^T$ to denote the probability of choosing the *i*th voting strategy by defender, and *j*th attack strategy by attacker, respectively, where $\sum_{i=1}^{S_d} x_i = 1, \sum_{j=1}^{S_a} y_j = 1$.

5: **Create** the equivalent linear programming problem for the optimal voting problem.

6: **Use** simplex algorithm to solve the linear programming problem, and get $\vec{P}_d$, $\vec{P}_a$ and *S*.

For any strategy $\vec{P}_d$ that the defender chooses, the attacker's goal is to choose an attack strategy $\vec{P}_a$ which minimizes the expected number of surviving clusters. In this case, the defender can expect to have at least min $\min_{\vec{P}_a} S(\vec{P}_d, \vec{P}_a)$ surviving clusters. Therefore, the defender aims to select his particular strategy $\vec{P}_d$ to maximize $\min_{\vec{P}_a} S(\vec{P}_d, \vec{P}_a)$.

It has been proved that $\min_{\vec{P}_a} S(\vec{P}_d, \vec{P}_a) = \min_{1 \le j \le S_a} \sum_{i=1}^{S_d} x_i s_{ij}$ (Vanderbei, 2001) . Therefore, no matter what the attacker's strategy is, the defender is assured of obtaining at least $\min_{1 \le j \le S_a} \sum_{i=1}^{S_d} x_i s_{ij}$. Let *X* be the lower bound for each $\forall j = 1,2,...,S_a$, $\min_{1 \le j \le S_a} \sum_{i=1}^{S_d} x_i s_{ij} \ge x$ Then, the defender's goal is equivalent to maximizing *X*, and we have

$$maximize\ X \tag{11}$$

subject to:

$$\sum_{i=1}^{S_d} x_i = 1 \tag{12}$$

$$\sum_{i=1}^{S_d} s_{ij} x_i \ge X \quad \forall j = 1,2,...,S_a \tag{13}$$

$$x_i \ge 0 \quad \forall i = 1,2,...,S_d \tag{14}$$

The attacker, on the other hand, tries to minimize the expected number of surviving clusters by choosing an optimal attack strategy against the defender's strategies. By using a similar analysis above, the attacker's goal is to minimize $\max_{\vec{P}_a} S(\vec{P}_d, \vec{P}_a)$ where $\max_{\vec{P}_d} S(\vec{P}_d, \vec{P}_a) = \max_{1 \le i \le S_d} \sum_{j=1}^{S_a} y_j s_{ij}$. Let *Y* be the upper bound for each of the *i* summations, that $\forall i = 1,2,...,S_d$, $\max_{1 \le i \le S_d} \sum_{j=1}^{S_a} y_j s_{ij} \le Y$, the attacker's goal is equivalent to minimizing *Y* , and we have

$$minimize \quad Y \tag{15}$$

subject to:

$$\sum_{j=1}^{S_a} y_j = 1 \tag{16}$$

$$\sum_{j=1}^{S_a} s_{ij} y_j \le Y \qquad \forall i = 1,2,\ldots,S_d \tag{17}$$

$$y_i \ge 0 \qquad \forall j = 1,2,\ldots,S_a \tag{18}$$

After solving these two linear programming problems by using the simplex algorithm proposed in (Cormen, Stein, Rivest, & Leiserson, 2001), we will get *X*, *Y* , $\overset{\rightarrow}{P_d}$ and $\overset{\rightarrow}{P_a}$, where both *X* and *Y* refer to the expected number of surviving clusters, $\overset{\rightarrow}{P_d}$ and $\overset{\rightarrow}{P_a}$ contain the probability of each strategy taken by the defender and attacker, respectively. Actually, according to Von Neuman's Minimax Theorem (Neumann, 1928), for both defender and attacker, if $\overset{\rightarrow}{P_d}$ and $\overset{\rightarrow}{P_a}$ are their optimal strategies, we can have $X = Y$.

The following example illustrates the process of determining the optimal number of voting replicas in the cluster.

Example 1: Assume a system has $C = 10$ clusters, and each cluster consists of $N_r = 25$ replicas, the reliability of each replica is $r = 0.9$. The total number of replicas the attacker can compromise is $N_t = 25$. For the defender, we assume that only an odd number of replicas is chosen to vote, therefore, the total number of voting strategies is $S_d = \left\lceil \frac{N_r}{2} \right\rceil = 13$. For the attacker, the total number of attack strategies is $S_a = \min\{C, N_t\} - \left\lceil \frac{N_t}{N_r} \right\rceil + 1 = 10$.

Table 1 displays all possible outcomes of an attack on 10 clusters with 10 scenarios of attack strategies and 13 possible defense choices. The values in the table represent the expected number of surviving clusters given the defense choice and the attack strategy.

Based on the results in the table, if the attacker attacks 8 out of 10 clusters ($A = 8$), and the defender chooses 13 out of 25 replicas to vote (N_v= 13), then the expected number of clusters which generate a correct result is 9.9716, and obviously, this strategy is not optimal for the attacker because if the defender sticks to the same voting strategy (i.e., choosing 13 out of 25 replicas to vote), then the attacker will prefer to attack 2 out of 10 clusters to decrease the expected number of surviving clusters to 8.6389, rather than attacking 8 out of 10 clusters. On the other hand, if the attacker sticks to his/her strategy, the defender would like to choose 25 replicas to vote and increase the expected number of surviving clusters to 9.9998. Therefore, for both attacker and defender, their initial strategies of attacking 8 out of 10 clusters and choosing 13 out of 25 replicas to vote is not optimal against each other.

TABLE 1

The result matrix for defender and attacker in Example 1

	A = 1	A = 2	A = 3	A = 4	A = 5	A = 6	A = 7	A = 8	A = 9	A = 10
N_v=1	8.1	8.1	8.1	8.1	8.1	8.1	8.1	8.1	8.1	8.1
N_v=3	8.748	8.6216	8.7621	8.8665	8.9417	8.9938	9.0322	9.066	9.0894	9.1076
N_v=5	8.923	8.7326	9.0231	9.2195	9.3464	9.4248	9.478	9.5228	9.5514	9.5718
N_v=7	8.9754	8.7384	9.1734	9.4364	9.5863	9.6671	9.7173	9.7574	9.781	9.7964
N_v=9	8.992	8.7135	9.2868	9.5912	9.7407	9.8099	9.8493	9.8788	9.8951	9.9048
N_v=11	8.9973	8.6785	9.3856	9.7081	9.8418	9.894	9.9215	9.9407	9.9508	9.9563
N_v=13	8.9991	8.6389	9.4769	9.7971	9.9066	9.9425	9.9601	9.9716	9.9774	9.9803
N_v=15	8.9997	8.5957	9.5629	9.8633	9.9467	9.9696	9.9802	9.9867	9.9898	9.9913
N_v=17	8.9999	8.5482	9.6437	9.9111	9.9706	9.9843	9.9904	9.9939	9.9955	9.9962
N_v=19	9	8.4949	9.7182	9.9441	9.9842	9.9921	9.9954	9.9972	9.998	9.9983
N_v=21	9	8.433	9.7846	9.966	9.9918	9.9961	9.9979	9.9988	9.9991	9.9993
N_v=23	9	8.357	9.8413	9.9799	9.9958	9.9981	9.999	9.9994	9.9996	9.9997
N_v=25	9	8.2542	9.8873	9.9885	9.9979	9.9991	9.9996	9.9998	9.9998	9.9999

In order to decide the optimal strategy in this example, we set the probability of the defender taking the ith strategy is x_i, the probability of the attacker taking the jth strategy is y_j. After transforming the problem into the linear programming problem, we solve this linear programming problem by using simplex algorithm, and we have X=Y=8.7384, $\vec{P}_d \left[0,0,0,1,0,0,0,0,0,0,0,0,0\right]^T$, and $\vec{P}_a = \left[0,1,0,0,0,0,0,0,0,0\right]^T$, which indicates that the fourth voting strategy is the defender's optimal strategy, and that is to choose 7 out of 25 replicas to vote, and the attacker's optimal strategy is the second strategy which is to attack 2 out of 10 clusters. For this case, the expected number of surviving clusters is 8.7384.

In order to show that their strategies are optimal against each other, we will traverse all the defender's strategies and check whether a better voting strategy is available when the attacker's optimal strategy is fixed. The attacker's optimal strategy can also be checked in a similar way. Based on the result table, we can see that when attacker chooses the second strategy, that is A = 2, no better voting strategy for defender is available, and when defender chooses the second strategy, that is N_v=7, no better attack strategy for attacker is available, either. Therefore, for both attacker and defender, their strategies are optimal against each other.

If there exists one element in vector $\vec{P}_d = \left[x_1,\ldots,x_{S_d}\right]^T$ which is equal to 1, it means its corresponding strategy is always the best no matter how the attacker's strategy is chosen. A similar conclusion can be made for the attacker if there exists one element in vector $\vec{P}_a = \left[y_1,\ldots,y_{S_a}\right]^T$ which is equal to 1. However, if no such element exists in the probability vector, it indicates that no deterministic decision can be obtained. To illustrate it, consider another example shown below.

Example 2: In this example, we have the same system setting as given in Example 1 except the reliability of replica and number of compromised components, that is we have C = 10, N_r = *25*, r = 0.7, and N_t = 37. In addition, for the defender, we assume that only an odd number of replicas is chosen to vote, therefore, the total number of voting strategies is

$S_d = \left\lceil \frac{N_r}{2} \right\rceil = 13$. For the attacker, the total number of attack strategies is $S_a = \min\{C, N_t\} - \left\lceil \frac{N_t}{N_r} \right\rceil + 1 = 10$.

After applying the same steps shown in Algorithm 2, we have $\vec{P_d}\,[0,0,0,0,0,0,0,0.745,0.255,0,0,0,0]^T$, $\vec{P_a} = [0,0.385,0.615,0,0,0,0,0,0,0,0]^T$, and X = Y = 6.8844. In this case, none of decision probability is 1. Therefore, no deterministic decision can be made.

In order to decide the optimal voting strategy under targeted attack, we present a more general approach to solving this problem. Based on the analysis above, we know the objective of the defender is to maximize the number of surviving clusters under the worst case scenario, i.e., if the defender's strategy is determined, an attacker chooses a strategy that minimizes the number of surviving clusters. In other words, the defender's objective is to maximize $S_{\min}$, where

$$S_{\min} = \min_{1 \le j \le S_a} \sum_{i=1}^{S_d} x_i s_{ij} \tag{19}$$

Algorithm 3 gives the procedure of finding the system's maximized minimum number of surviving clusters. The main idea of Algorithm 3 is that for each voting strategy, we find the minimum number of surviving clusters under all possible attack strategies. Then we choose the voting strategy under which the minimum number of surviving clusters is maximized.

Algorithm 3 FINDING THE SYSTEM'S MAXIMIZED MINIMUMNUMBER OF SURVIVING CLUSTERS

Input: Matrix M = $(s_{i,j})S_a \times S_d$.

Output: System's maximized minimum number of surviving clusters $S_{\max\min}$, and voting strategy vector $\vec{P_d}$

```
$S_{\max\min} \leftarrow 0$;  $\vec{P_d} \leftarrow 0$
  **for** $i \leftarrow 1$ to $S_d$ **do**
    $S_{\min} \leftarrow C$
    **for** $j \leftarrow 1$ to $S_a$ **do**
```

```
5:   if $S_{\min} > s_{i,j}$ then
6:     $S_{\min} \leftarrow s_{i,j}$
7:   end if
8:   end for
9:   if $S_{\max\min} < S_{\min}$ then
10:    $S_{\max\min} \leftarrow S_{\min}$
11:    Set $x_i$ to 1, and the rest to 0.
12:  end if
13: end for
14: return $S_{\max\min}, \vec{P_d}$
```

A brief explanation of Algorithm 3: from Line 4 to Line 8, we obtain the system's minimum number of surviving clusters under all of the possible attack strategies when the defender chooses the ith $(1 \le i \le S_d)$ voting strategy. From Line 9 to Line 12, the defender chooses the strategy under which the system's minimum reliability is maximized. Finally, we output the system's maximized minimum reliability $S_{\max\min}$ and the corresponding defender’s strategy vector $\vec{P_d}$ (Line 14).

Theorem 2: Algorithm 3 obtains the maximized minimum number of surviving clusters under all possible attack strategies.

Proof: We prove the theorem by contradiction. If there exists a higher maximized minimum number of surviving clusters under all possible attack strategies, it must exist in one of the S_d strategies. Since Algorithm 3 compares the minimum number of surviving clusters under all possible attack strategies and chooses the largest among them. Therefore, we cannot obtain a higher maximized minimum number of surviving clusters than the one produced by Algorithm 3.

Complexity Analysis: In Algorithm 3, the time complexity of calculating matrix M = $(s_{i,j})S_a \times S_d$ is $O((N_r)^3 \times C)$. In order to find the maximized minimum number of surviving clusters, the time required is $O(N_r \times C)$ .Therefore, the complexity of the algorithm is $O((N_r)^3 \times C)$.

In order to illustrate how Algorithm 3 works, we revisit Example 2 below.

Example 3: Assume the system setting is the same as given in Example 3. That is C = 10, N_r = 25, r = 0.7, and N_t = 37. Table 2 shows the number of surviving clusters under possible voting and attack strategies.

According to Algorithm 3, we first get the minimum number of surviving clusters under each voting strategy, which is shown in Table 3.

TABLE 2

The result matrix for defender and attacker in Example 3

	A=2	*A=3*	*A=4*	*A=5*	*A=6*	*A=7*	*A=8*	*A=9*	*A=9*
$N_v=1$	5.964	5.964	5.964	5.964	5.964	5.964	5.964	5.964	5.964
$N_v=3$	6.432	6.3396	6.3457	6.3661	6.3867	6.4047	6.4199	6.4335	6.4439
$N_v=5$	6.7637	6.5554	6.5681	6.6175	6.6673	6.7101	6.7458	6.7776	6.8015
$N_v=7$	7.0181	6.6902	6.7079	6.7919	6.8764	6.9479	7.0067	7.0585	7.0967
$N_v=9$	7.2182	6.7761	6.7963	6.9189	7.0419	7.1444	7.2273	7.2996	7.3519
$N_v=11$	7.3763	6.8303	6.8494	7.0139	7.1784	7.3131	7.4203	7.5127	7.5783
$N_v=13$	7.5013	6.8633	6.8772	7.0864	7.2946	7.4619	7.593	7.7045	7.7821
$N_v=15$	7.5999	6.882	6.8859	7.1425	7.3962	7.5961	7.75	7.8792	7.9674
$N_v=17$	7.6778	6.8914	6.88	7.1864	7.4872	7.7192	7.8945	8.0398	8.1371
$N_v=19$	7.7396	6.8951	6.8626	7.2214	7.5705	7.8336	8.0288	8.1882	8.293
$N_v=21$	7.7889	6.8958	6.8357	7.2497	7.6481	7.941	8.1543	8.326	8.4366
$N_v=23$	7.8284	6.8959	6.801	7.2731	7.7215	8.0429	8.2722	8.4542	8.5692
$N_v=25$	7.8602	6.8971	6.7596	7.2932	7.792	8.14	8.3835	8.5736	8.6917

TABLE 3

The minimum number of surviving clusters under each defense strategy in Example 3

$N_v=1$	$N_v=3$	$N_v=5$	$N_v=7$	$N_v=9$	$N_v=11$	$N_v=13$
5.964	6.3396	6.5554	6.6902	6.7761	6.8303	6.8633
$N_v=15$	$N_v=17$	$N_v=19$	$N_v=21$	$N_v=23$	$N_v=25$	
6.882	6.88	6.8626	6.8357	6.801	6.7596	

From Table 3, we can see that the maximized minimum number of surviving clusters we can achieve is S = 6.882 when the number of voters is N_v = 15. In other words, no matter which attack strategy the attacker takes, the expected number of surviving clusters in the system is no smaller than S = 6.882.

5. EXPERIMENTAL RESULTS

In this section, we empirically investigate how the number of compromised replicas and replicas' reliability affect the voting strategy and expected number of surviving clusters, and how the number of voting replicas affects the reliability of voting results when a system is under random and targeted attacks, respectively. The voting strategies under random and target attacks are determined by Algorithm 1 and Algorithm 3, respectively.

Voting Strategy under Random Attack

In this subsection, we discuss how the number of compromised replicas and replicas' reliability impact the selection of a voting strategy to fight against random attacks. We assume the system consists of 10 clusters, the number of replicas in each cluster is 15, and the reliability of each replica is set to be 0.9, 0.8, 0.7, and 0.4, respectively. That is C = 10, N_r = 15, and r = 0.9, 0.8, 0.7, and 0.4. When the total number of replicas the attacker can compromise increases from 0 to 100, Figure 2 shows the change of the optimal voting strategy.

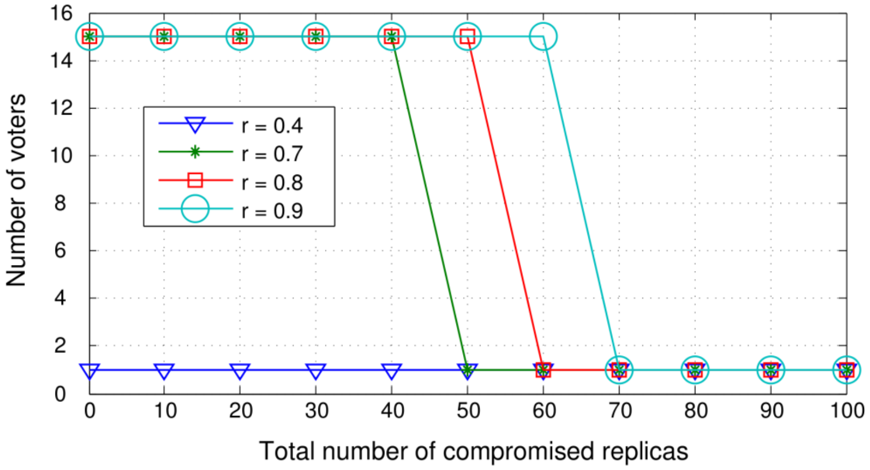

Figure 2. The relationship between the optimal voting strategy N_v and total number of compromised replicas N_t when system is under random attack

From Figure 2, we can see that when the number of compromised replicas increases, the voting strategy changes dramatically. For instance, when the reliability of replica is r = 0.9 and the total number of compromised replicas increases from 60 to 70, the number of voters decreases from 15 to 1. The reason is that when the number of compromised replicas is small, the majority of the replicas in each cluster is uncompromised, and a higher reliability can be achieved when more replicas participate in the voting. However, if the majority of replicas in the cluster is compromised, the more replicas participate in the voting, the more likely that a compromised replica participated in a voting hence lowers the probability of reaching a correct result. Therefore, under this situation, the best option is to select only one replica and uses its result as the final value.

Another observation that can be made from Figure 2 is that the changing point of the voting strategies varies under different replicas' reliability. For example, if r = 0.7, the voting strategy changes when the total number of compromised replicas exceeds 40, while if r = 0.9, the voting strategy changes at the point where the total number of compromised replicas exceeds 60. This is because when compromised replicas exist in a voting process, if the reliability of uncompromised replicas is low, the probability that the majority of those replicas produce a correct is smaller than the case if only one replica is selected as the sole voter. However, if the reliability of replicas is high, the probability that the majority of those highly reliable replicas produce a correct result is larger than the case if only one replica is selected as voter.

To better explain it, consider an example in which there are five replicas, and two of them are compromised. If the reliability of the replicas is r = 0.7 and all five replicas are participating in a voting process, the probability that a correct result can be obtained is $r^3 = 0.7^3 = 0.343$. On the other hand, if we only select one replica as the voting replica, the probability to obtain a correct result is $0.7 \times \frac{3}{5} = 0.42$, which is larger than the case in which all replicas participate in the voting. However, if the reliability of the replicas is r = 0.9 and all five replicas are participating in the voting process, the probability to produce a correct result is r^3= 0.93 = 0.729, which is larger than the case in which only one replica is selected as the voter, i.e., $0.9 \times \frac{3}{5}$ = 0.54.

In the case when the reliability of the replicas is low, i.e., r = 0.4, from Figure 2, we can see that even the number of compromised replicas is 0, the optimal voting strategy is to select only one replica. This is because if the reliability of replicas is below 0.5, the probability to produce an incorrect output is larger than producing a correct output. Therefore, involving more replicas with low reliability decreases the probability of obtaining a correct result. This also indicates that having more replicas in a voting scheme with low reliability does not improve system's reliability.

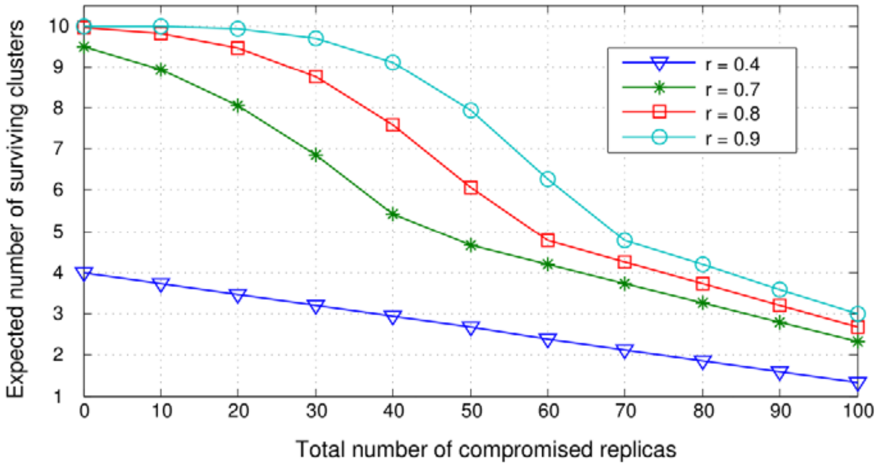

Figure 3: The relationship between the expected number of surviving clusters S and the total number of compromised replicas N_t when system is under random attack

Figure 3 shows how expected number of surviving clusters changes with the number of compromised replicas. From Figure 3 we can see that for all cases, the expected number of surviving clusters decreases when the number of compromised replicas increases. This is because as the number of compromised replicas increases, the probability that voting components are compromised increases accordingly, thus the probability of producing a correct result through majority voting decreases. From Figure 3, we can also see that under the same attack condition, i.e., the total number of compromised components is the same, systems with highly reliable replicas have more surviving clusters (i.e., more clusters can produce correct results). A complete presentation regarding the relationship between the expected number of surviving clusters S, the replica's reliability r, and the number of compromised replicas N_t is shown in Figure 4.

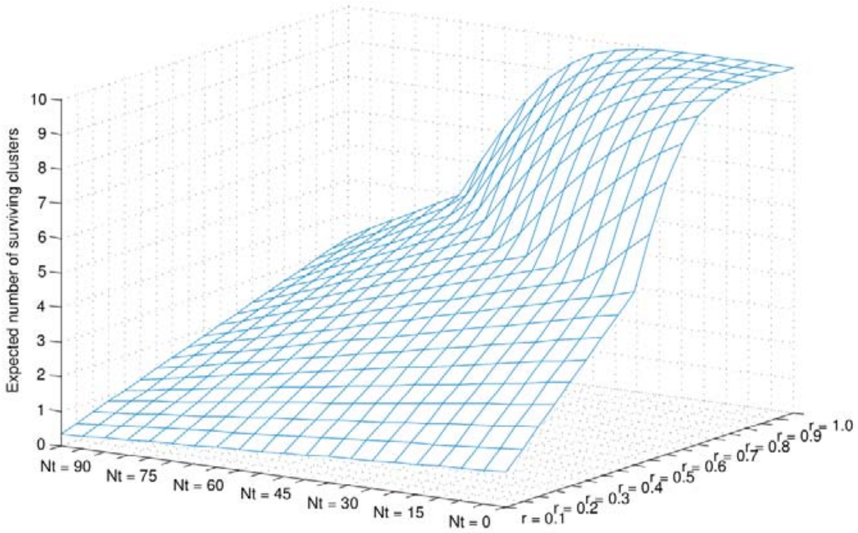

Figure 4. The relationship between the expected number of surviving clusters S, the replica's reliability r, and the number of compromised replicas N_a when system is under random attack

Figure 5 shows the performance of fixed voting strategies and optimal voting strategy when system is under random attack. In particular, we set the value of N_v to be constant 1, 7, and 15, or decided by Algorithm 1, respectively, and the reliability of replicas is $r = 0.9$. From Figure 5 we can see that the optimal voting strategy obtained by Algorithm 1 performs the best than other fixed voting strategies. For instance, when the number of compromised replicas is no greater than 60, the optimal voting strategy is to select all replicas (i.e., $N_v = 15$) to vote, otherwise, we should select only one voter. This indicates that under different system settings, the voting strategy may be changed to maximize the number of surviving clusters.

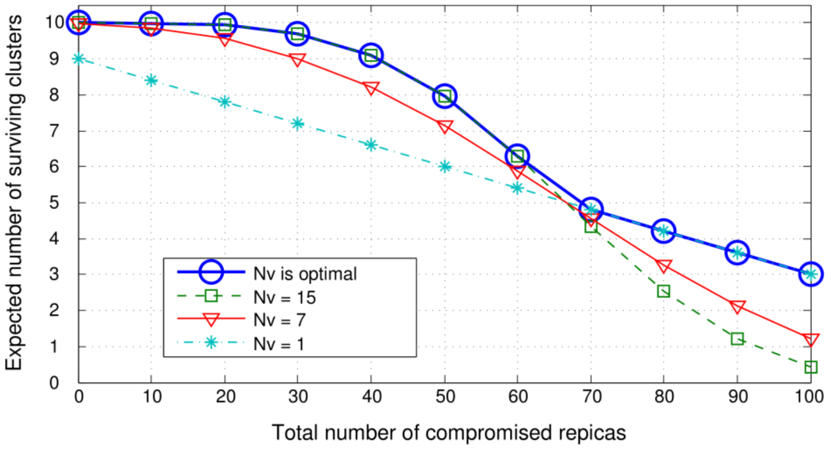

Figure 5. The expected number of surviving clusters under different numbers of voting replicas

Voting Strategy under Targeted Attack

In this subsection, we discuss how the number of compromised replicas and replicas' reliability impact the selection of a voting strategy to fight against targeted attacks. The experiment settings for C, N_r, and r are the same as given in the previous experiment settings, In other words, we have $C = 10$, $N_r = 15$, and $r = 0.9$, 0.8, 0.7, and 0.4. The number of compromised replicas increases from 0 to 70.

Figure 6 shows how voting strategy changes with different total number of compromised replicas. From Figure 6, we can see that the more replicas are compromised, the less voters are involved in the voting process. This is because if the majority of the replicas in a cluster is compromised, choosing less voters can achieve higher probability of obtaining a correct result.

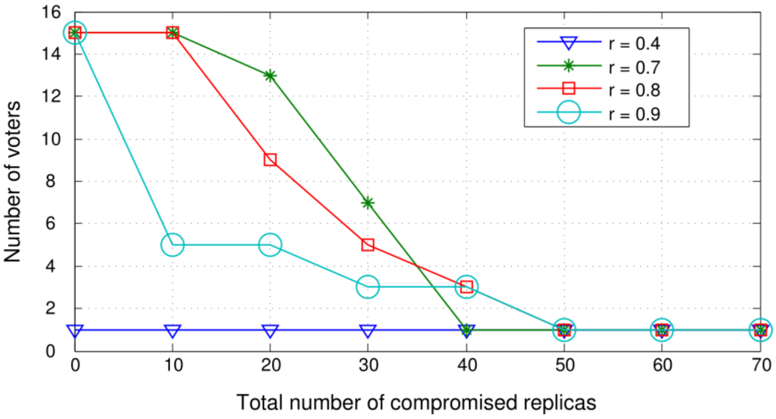

Figure 6. The relationship between the optimal voting strategy N_v and total number of compromised replicas N_t when system is under targeted attack

Another observation can be made in Figure 6 is that unlike the change of voting strategy under random attack where the number of voters decreases from 15 to 1 abruptly, the number of voters under targeted attacks decreases gradually. This is because when the system is under targeted attack, only a subset of clusters are under attack. In other words, for unattacked cluster, having more replicas to vote improves the reliability of these clusters. But for the attacked clusters, having more replicas may decrease voting reliabilities. Therefore, a tradeoff is considered when both unattacked and attacked clusters exist.

When system is under targeted attack, the relationship between the expected number of surviving clusters S and total number of compromised replicas N_t is shown in Figure 7. Similar to random attack, the expected number of surviving clusters decreases when the total number of compromised replicas increases. However, from Figure 3 and Figure 7, we can see that the expected number of surviving clusters under targeted attack is smaller than the one under random attack even the number of compromised components is the same. In other words, for attackers, targeted attacks are more effective.

A more complete presentation regarding how the expected number of surviving clusters changes under different the replicas' reliability and different number of compromised replicas is shown in Figure 8. In this experiment, we have $C = 10$; $N_r = 15$, and the replicas' reliability increases from 0.1 to 1.0, and the total number of compromised replicas increases from 0 to 100. Figure 8 shows that the number of surviving clusters increases when the replicas' reliability increases, and the number of surviving clusters decreases when the number of replicas the attacker can compromise becomes larger, which is consistent with Figure 7.

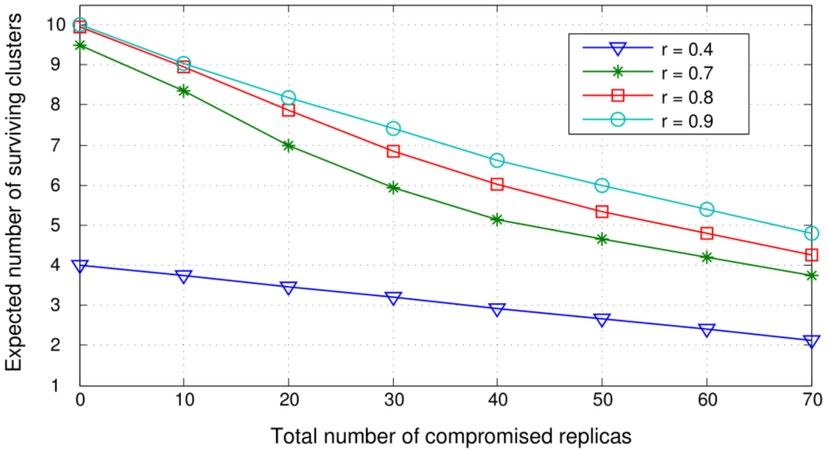

Figure 7. The relationship between the expected number of surviving clusters S and total number of compromised replicas N_t when system is under targeted attack

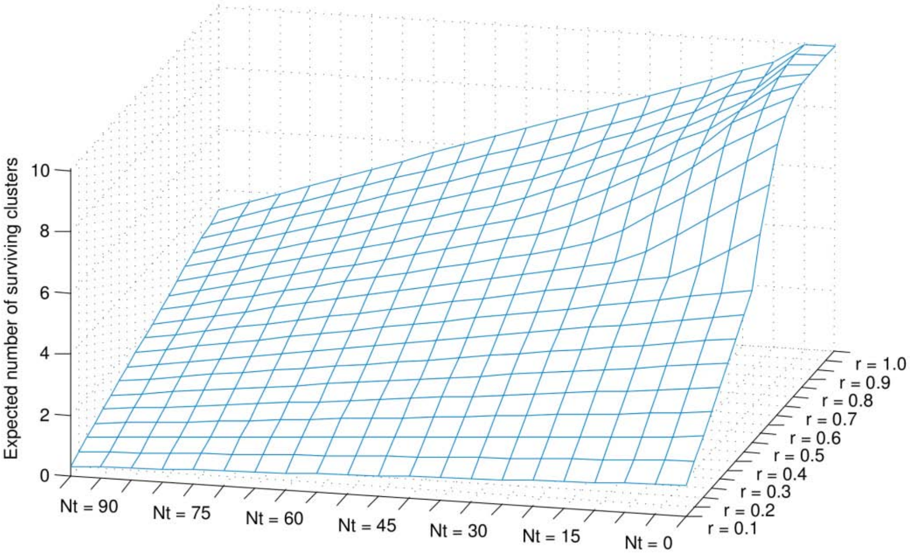

Figure 8. The relationship between the expected number of surviving clusters S, the replica's reliability r, and the number of compromised replicas N_t when system is under targeted attack.

Figure 9 shows the expected number of surviving clusters under different number of voting replicas. More precisely, we set C = 10; N_r = 15, and r = 0.9, and let the number of compromised replicas vary from 0 to 80. The value of N_v is set to constant 1, 7, and 15, or chosen by the Algorithm 3, respectively. From Figure 9 we can see that the expected number of surviving clusters with a fixed number of voting replicas are always no greater than the case in which the number of voting replicas is optimally decided. This result validates our conjecture that when the system is under attack, the number of replicas that participate in a voting processing may not be positively proportional to the reliability of the system.

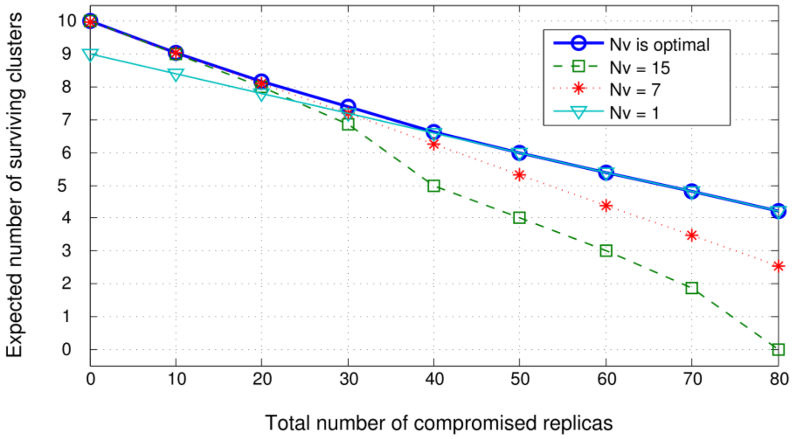

Figure 9. The expected number of surviving clusters under different numbers of voting replicas

6. CONCLUSION

In this paper, we studied how replicas' reliability and the number of compromised replicas impact the decisions on number of participating voters when the system is under random attack or targeted attack. In addition, we also investigated how the number of participating voters may impact the reliability of a voting algorithm. We presented a solution to decide the optimal number of voting replicas to maximize the expected number of clusters that can produce a correct result under attack. In addition to formally analyzing the relationships among the voting strategy, replicas' reliability, number of compromised replicas, and the number of surviving clusters, empirical studies are also performed to provide concrete observations. In this paper, the replicas are assumed to be homogeneous. In other words, they have the same reliability and the same voting weight. It would be interesting to know, which is also our next research target, how the system behaves when replicas are heterogeneous with respect to their reliabilities and voting weights.

ACKNOWLEDGEMENT

The work is supported in part by NSF CAREER CNS 0746643 and NSF CNS 1018731. Approved for Public Release; Distribution Unlimited: 88ABW-2011-5180, Sep 28,2012.

References

Bier, V. M., Nagaraj, A., & Abhichandani, V. (2005). Protection of simple series and parallel systems with components of different values. *Reliability Engineering & System Safety, 87*(3), 315 - 323.

Cormen, T. H., Stein, C., Rivest, R. L., & Leiserson, C. E. (2001). *Introduction to Algorithms* (2nd ed.): McGraw-Hill Higher Education.

Davcev, D. (1989). A dynamic voting scheme in distributed systems. *IEEE Transactions on*

Software Engineering, 15(1), 93 -97.

Dominguez-Garcia, A. D., & Grainger, P. T. K. (2008). *A Framework for Multi-Level Reliability Evaluation of Electrical Energy Systems.* IEEE Energy 2030 Conference.

Gass, S. I. (1984). *Linear programming: methods and applications (5th ed.).* New York, NY, USA: McGraw-Hill, Inc.

Hardekopf, B., Kwiat, K., & Upadhyaya, S. (2001). *A Decentralized Voting Algorithm for Increasing Dependability in Distributed Systems.* World Multiconference on Systemic, Cybernetics and Informatics.

Hausken, K. (2008). Strategic defense and attack for series and parallel reliability systems. *European Journal of Operational Research, 186*(2), 856-881.

Kuhn, H. W., & Tucker, A. W. (1950). *Contributions to the Theory of Game* (Vol. 1): Princeton University Press.

Kwiat, K., Taylor, A., Zwicker, W., Hill, D., Wetzonis, S., & Ren, S. (2010). *Analysis of binary voting algorithms for use in fault-tolerant and secure computing.* International Conference on the Computer Engineering and Systems (ICCES).

Latif-Shabgahi, G., Bass, J. M., & Bennett, S. (2004). A taxonomy for software voting algorithms used in safety-critical systems. *IEEE Transactions on Reliability, 53*(3), 319 - 328.

Levitin, G., & Hausken, K. (2008). Protection vs. redundancy in homogeneous parallel systems. *Reliability Engineering & System Safety, 93*(10), 1444 - 1451.

Levitin, G., & Hausken, K. (2009). False targets efficiency in defense strategy. *European Journal of Operational Research, 194*(1), 155-162.

Levitin, G., & Hausken, K. (2009). False targets vs. redundancy in homogeneous parallel systems. *Reliability Engineering & System Safety, 94*(2), 588 - 595.

Levitin, G., & Hausken, K. (2009). Meeting a demand vs. enhancing protections in homogeneous parallel systems. *Reliability Engineering & System Safety, 94*(11), 1711 - 1717.

Levitin, G., & Hausken, K. (2009). Parallel systems under two sequential attacks. *Reliability Engineering & System Safety, 94*(3), 763-772.

Levitin, G., & Hausken, K. (2009). Redundancy vs. Protection vs. False Targets for Systems Under Attack. *IEEE Transactions on Reliability, 58*(1), 58 -68.

McQueen, M. A., Boyer, W. F., Flynn, M. A., & Beitel, G. A. (2006). Time-to-Compromise Model for Cyber Risk Reduction Estimation. *Quality of Protection, 23*, 49-64.

Neumann, J. V. (1928). Zur Theorie der Gesellschaftsspiele. Mathematische Annalen, 100(1), 295-320.

Raghavan, T. E. S. (1994). *Zero-sum two-person games.* Handbook of Game Theory with Economic Applications.

Siewiorek, D. P., & Swarz, R. S. (1998). *Reliable computer systems (3rd ed.): design and evaluation*. Natick, MA, USA: A. K. Peters, Ltd.

Thomas, R. H. (1979). A Majority consensus approach to concurrency control for multiple copy databases. *ACM Trans. Database Syst., 4*, 180-209.

Tong, Z., & Kain, R. Y. (1991). Vote assignments in weighted voting mechanisms. *IEEE Transactions on Computers, 40*(5), 664 -667.

Trivedi, K. S. (2002). *Probability and statistics with reliability, queuing and computer* *science applications* (2nd edition ed.). Chichester, UK: John Wiley and Sons Ltd.

Vanderbei, R. J. (2001). *Linear Programming: Foundations and Extensions* (Second ed.): Springer.

Vicki M. Bier, V. A. (2002). *Optimal allocation of resources for defense of simple series and parallel systems from determined adversaries.* Engineering Foundation Conference on Risk-Based Decision making in Water Resources X.

Wang, L., Leiferman, Y., Ren, S., Kwiat, K., & Li, X. (2010). *Improving complex distributed software system availability through information hiding.* ACM Symposium on Applied Computing.

Wang, L., Ren, S., Yue, K., & Kwiat, K. (2010). Optimal Resource Allocation to Improve Distributed System Reliability. *Workshop on Secure Knowledge Management*.

Wang, L., Ren, S., Yue, K., & Kwiat, K. (2011, March). *Optimal Resource Allocation for Protecting System Availability against Random Cyber Attacks.* International Conference on Computer Research and Development.

Yalaoui, A., Chatelet, E., & Chu, C. (2005). A new dynamic programming method for reliability redundancy allocation in a parallel-series system. *IEEE Transactions on Reliability, 54*(2), 254 - 261.

Zafra, P. (2011). *Linear Programming and Two-Person Zero-Sum Games*: Wiley Encyclopedia of Operations Research and Management Science.