A Quantitative Comparison of Reactive and Proactive Replicated Storage Systems

Rossana Motta and Joseph Pasquale

Department of Computer Science and Engineering
University of California San Diego
La Jolla, USA

Email: {rmotta, pasquale}@eng.ucsd.edu

Abstract—Replicated storage systems allow their stored data objects to outlive the life of the nodes storing them through replication. In this paper, we focus on *durability*, and more specifically on the concept of an object’s lifetime, i.e., the duration of time between the creation of an object and when it is permanently irretrievable from the system. We analyze two main replication strategies: *reactive*, in which replication occurs in response to failures, and *proactive*, in which replication occurs in anticipation of failures. Our work presents a quantitative analysis that compares reactive and proactive through analytical models and simulations, considering exponentially distributed failures and reactive repairs, and periodic proactive replications. We also present a derivation of the analytical formula for the variance of the lifetime in the reactive model. Our results indicate that a proactive strategy leads to multiple times higher storage requirements than a reactive strategy. In addition, reactive systems are only moderately bursty in terms of bandwidth consumption, with rare peaks of at most five times the bandwidth consumption in proactive systems (given input parameter values that are compatible with real systems). Finally, for both strategies, the standard deviation is very close to the expected lifetime, and consequently, the lifetimes close to being exponentially distributed.

I. INTRODUCTION

Replicated storage systems are currently used in a wide variety of contexts to extend the lifetime of data objects beyond the lifetime of the node(s) hosting the data. In this work, we focus on *durability* in replicated storage systems, that is the duration for which an object is retrievable from the system. More precisely, the *lifetime* of an object is the interval of time between the creation of the object and the time that it is permanently irretrievable from the system. The lifetime is extended by replicating, i.e., making copies, of the object (in whole or in parts). New replicas can only be created if there still are existing replicas on some node(s) in the system; if there are not, then the object’s lifetime is ended at the point in time when there are no more replicas. Since repairing replicas takes time (depending on the repair rate), if, at some point in time, failures happen faster than repairs, the number of active replicas may drop to zero before there is a chance to repair, thus ending the lifetime of the object.

Note that durability is different from *availability*, the latter measuring the fraction of time that the object is available (i.e., can be retrieved immediately) during its lifetime. An object

would be alive but unavailable if its replicas were on nodes that were temporarily down.

One can consider two main strategies of replication for replicated storage systems:

- *reactive*, in which failures trigger repairs, that is the creation of new replicas (immediately, or after a delay determined by some condition such as when the number of replicas falls below a given threshold);
- *proactive*, in which replications occur in anticipation of failures, often in a periodic fashion.

An open question regarding these strategies is: Which one is better? The differences between reactive and proactive models may be characterized in terms of expected lifetime, variability of lifetime, and resource consumption (total storage used for replicas, bandwidth used to transfer copies of replicas). For instance, one may want to know which strategy achieves longer durabilities given different parameters, or which values of the parameters allow for a certain durability, for each strategy of replication. Questions like these require a quantitative and comparative analysis of reactive and proactive models, which is the focus of this paper.

The variability of the lifetime in models for durability is another important topic. Knowing the variability of the lifetime is important because it allows one to estimate the dispersion of the values around the expected value. We analyze the variability of the lifetime through its standard deviation, deriving a new formula for the variance of the lifetime for the reactive model, and present results of a simulation study in the proactive model.

II. RELATED WORK

A number of distributed storage systems have been proposed. Among those implemented on physical nodes or simulated, there are some that can be classified as reactive, such as Oceanstore [12], Glacier [9], Carbonite [8], TotalRecall [2] and Thriftstore [7], and those that can be classified as proactive, such as Phoenix [11] and Tempo [1]. Oceanstore, TotalRecall and Phoenix focus on availability. The others focus on durability.

Each system has unique features: Oceanstore [12] opportunistically caches objects in a nomadic fashion, overcoming the lack of trust through cryptography. Glacier [9] focuses on

erasure coding, dealing with massive correlated failures and the traffic due to replication. Carbonite [8] aims at minimizing the resource consumption by replicating only when the total number of replicas falls below a given threshold.

TotalRecall [2] automates the management of replication, allowing users to choose between full replication and erasure coding and to specify a given availability target.

Thriftstore [7] allows one to choose either availability or durability as a system parameter. Phoenix [11] is a cooperative architecture for backup that focuses on the diversity of hosts to replicate in a way that correlated failures can be avoided and object can survive catastrophic events.

Tempo [1], similar to Carbonite, uses threshold to stop replicating when the number of replicas reaches a given number. Tempo also presents a quantitative comparison with Carbonite, thus offering a comparison of a proactive system with a reactive one, which is especially relevant to this paper. A primary finding is that Tempo performs better than Carbonite in terms of bandwidth usage, as peaks are avoided.

Our primary difference with the above related works is that we analyze the core replication strategies in their most basic forms, and how they affect the expected lifetime, its variability and the resource requirements. We also assume that the system is fully distributed, with no central authority monitoring how many copies of objects are in the system.

Regarding the approach to modeling, other studies have used analytic models, including reactive ([14] [4] [3] [13]) and proactive ([5]). However, none of these analytic works present results about the variability of the lifetime.

III. MODELS AND ASSUMPTIONS

We make the following simplifying assumptions to our models. These allow us to obtain results on interesting first-order behaviors while still making the models tractable and easy to understand:

- We consider a single replicated object whose lifetime is to be determined.
- A node can hold exactly one replica (thus, if the node fails, the replica is gone).
- There are $n > 1$ initial nodes, with n depending on the available storage.
- We base our models on Markovian hypotheses, according to which the distributions of inter-failure times for reactive and proactive are exponential with parameter λ (the mean failure rate). The distributions of repair times (i.e., time to repair) are also exponential for reactive, with parameter μ (the mean repair rate). In a proactive system, a replication occurs every $1/\rho$ interval of time, with ρ being the replication rate.
- When we select values for λ and μ , we base them on measurement data from PlanetLab [10], a global research network consisting of about 1000 worldwide distributed nodes.

IV. ANALYTICAL VALUE OF THE VARIANCE IN A REACTIVE SYSTEM

We quantify the variability of the lifetime with its statistical variance. Knowing the variance of the model gives a sense of how precise the prediction of expected lifetime can be. While an analytical expression for the average lifetime in a replicated storage system can be found in [14], no such expression exists for the variance.

With the aid of theory developed in [3], we present an explicit formula for lifetime variance with the following parameters:

- n , the initial number of nodes
- γ defined as the ratio between the repair rate μ and the failure rate λ
- $E(L)$, the expected value of the lifetime, whose formula, as reported in [14], is given by:

$$E(L) = \frac{1}{n\lambda} \sum_{k=0}^{n-1} \sum_{j=0}^k \binom{n}{k-j} \gamma^{k-j}$$

As we detail in the appendix, the variance is given by:

$$Var(L) = E^2(L) + \frac{2E(L)}{\lambda(1+\gamma)} \sum_{r=1}^n \binom{n}{r} \frac{(-1)^r}{r} + \frac{2}{\lambda^2(1+\gamma)^2} \sum_{r=1}^n \binom{n}{r} \frac{\gamma^r - (-1)^r}{r^2}$$

Upon further analysis of this expression, regardless of the values of n , λ and μ , the standard deviation is very close to the expected value, as the first term in the variance dominates the remaining two terms. To be more precise, the standard deviation is always slightly smaller than the expected lifetime, as the sum of the second and third terms is always negative, but only slightly so. This is because the second term is always negative and slightly larger than the third term, which is always positive. Since the expected lifetime and the standard deviation are almost identical, this suggests that perhaps lifetime is exponentially distributed.

The fact that, for exponentially distributed failures and repairs, the standard deviation of the lifetime is very close to the expected lifetime implies that the dispersion of the lifetime itself can be high. Interestingly, both the expected value and standard deviation increase exponentially with n (this is referring to the form of the curves of the functions themselves, *not* be confused with the separate point that the lifetime may be exponentially distributed because the expected value and standard deviation are very similar). So, for large n , a prediction can be off both in relative terms and absolute terms. For example, even if the expected lifetime of a data object is predicted to be many years, this prediction can be off by the same amount, thus in the worst case scenario the actual lifetime can be close to zero, providing little comfort that the replication is achieving its goal.

V. REACTIVE VS. PROACTIVE: ANALYSIS RESULTS

A. Dispersion of the Lifetime

From a practical point of view, knowing how dispersed the values of the lifetime are, which can be measured by the

Fig. 1. Histograms for lifetime in reactive (top) and proactive (bottom) models, each based on 1000 simulations.

variance, is important as it tells us how much one can expect the average lifetime to vary when designing a distributed storage system with given parameters. Our basic reactive and proactive models generate lifetimes with high dispersions. Figure 1 shows the results of simulations for both reactive and proactive in the form of histograms. We can immediately see that, qualitatively, the majority of lifetimes are very small, but there are some that are much larger than the mean value.

We carried out Chi-Square Goodness of Fit tests comparing the histograms to the exponential distribution. This resulted in a value of 9.4436 for reactive and 10.1501 for proactive, where the requirement for a good fit with a stringent significance level of 0.001 is less than 37.70. Consequently, we can say with high confidence that lifetime for both reactive and proactive follows an exponential distribution. This empirically confirms what was surmised earlier when we showed analytically that the expected and standard deviation curves for lifetime were nearly identical.

B. Storage Requirements

A basic question when comparing reactive and proactive models is: How much storage do they require, on average and at the maximum, to achieve the same level of expected lifetime? Such a comparison is significant in any real world scenario, as in most cases there are constraints on the number of available nodes and on the cost of the storage itself.

We compared reactive and proactive through the analysis of two parameters: the maximum number of replicas reached at some point during the lifetime of the object and the average replica creation rate, defined as the total number of created replicas divided by the lifetime of the object. The two parameters vary as n , λ , μ and ρ vary, thus making the problem multidimensional. Therefore, we restricted our study to some real world scenarios with μ varying between 0.5λ and 4.5λ . While the scenarios are also small-scale, with n varying between 2 and 10 (as simulation times grow rapidly with larger values of n), they provide enough insight given the goals of this study.

Figure 2 reports the maximum number of replicas and the average replica creation rate averaged over 100,000 simulations, in the reactive and in the proactive models, as a function of n , with $\lambda=1$ and $\mu=2.756$. Given n , λ and μ , the proactive replication rate ρ is determined so that the mean lifetime in proactive is the same (up to the second decimal digit) as the mean lifetime in reactive. The replica creation rate was computed measuring the total created replicas in one simulation and dividing this quantity by the mean lifetime for that simulation. To achieve statistical significance, the simulations were repeated 100,000 times and each point in Figure 2 represents their average.

Figure 3 again shows the maximum number of replicas and the average replica creation rate, but this time as a function of μ/λ , for $n=6$. In a real system, the failure rate λ is typically a given, as hardware, software or network components have inherent average failure rates. For this reason, we assume λ as constant and equal to 2, and change the value of μ only. The choice of the value 2 for λ is based on data from PlanetLab [14], where the average failure rate is a little over 1 failure/week. We chose to round up to 2 to present a more aggressive scenario.

Plotting the maximum number of replicas and the average replica creation rate both as a function of n (Figure 2) and as a function of γ (Figure 3) shows that proactive uses considerably more storage than reactive, especially at the maximum, when the usage in proactive is multiple times that in reactive. How much more storage is used depends on the values of the parameters, as the storage is affected by each of them, namely n , λ , μ and ρ .

To further analyze the maximum storage requirements in proactive, in Figure 4 we show the ratio between the maximum number of concurrently active replicas and the initial number of replicas as a function of ρ , for $n=2$ and $\lambda=2$. As the repair rate μ grows in reactive, the replication rate ρ in proactive must also grow, in order to achieve the same value of mean lifetime. As the replication becomes faster, the number of concurrently active replicas also tends to grow. We show plots with a very low value of n because this is the case that maximizes the ratio between the maximum number of concurrently active replicas and the initial number of replicas in proactive.

Our simulations show that the proactive model produces a higher number of concurrently active replicas, which is almost always greater than the initial value of n , to which

(a) Maximum number of replicas. For reactive, the maximum number of replicas is the same as n , because replication occurs only to repair failures.

(b) Average replica creation rate, defined as the total number of created replicas divided by the lifetime of the object.

Fig. 2. Maximum number of replicas and average replica creation rate as functions of n , with $\lambda=1$, $\mu=2.756$, $\rho=1.32-8.49$. The plots show that proactive uses considerably more storage than reactive, especially at the maximum of the storage usage.

the reactive model is inherently capped. The ratio between the maximum number of replicas and the initial number of replicas progressively increases in proactive as the failure rate becomes smaller than the repair and replication rates. This is because in this case replication keeps occurring but failures happen much more slowly, thus accumulating replicas.

One could implement a system-wide check to cap the total number of replicas to a given number in a proactive system [1]. This comes at the price of either having at least one central authority that maintains knowledge of the entire system at any point in time and is able to communicate with each node, presenting a critical point of failure. Or, one could also implement a decentralized strategy to monitor the existing number of replicas in a system, but a frequent, system-wide communication may introduce significant network overhead.

The replica creation rate varies considerably according to the system parameters. For values of the failure rate λ being multiple times smaller than the repair rate μ , which is the most realistic scenario, proactive produces a higher average replica

(a) Maximum number of replicas. Since we keep n constant, reactive never reaches a maximum number of replicas that is larger than n . This is why the curve for reactive is a horizontal line.

(b) Average replica creation rate, defined as the total number of created replicas divided by the lifetime of the object.

Fig. 3. Maximum number of replicas and average replica creation rate as functions of μ/λ , with $\lambda=2$, $n=6$, $\rho=3.3-12.76$. The plots show that proactive uses considerably more storage than reactive, especially at the maximum of the storage usage.

creation rate than reactive. On the other hand, for λ larger than μ , the average replica creation rate tends to be lower in proactive than in reactive. This, however, is an unrealistic scenario, as in general the mean time to repair a component is shorter than its mean time to failure.

Overall, our study shows that in the most likely scenarios, opting for a proactive system will cost multiple times more storage compared to the same system implemented with a reactive replication strategy.

C. Bandwidth Requirements

Along with the storage requirements, the bandwidth requirements represent a fundamental aspect of the comparison between reactive and proactive models. It has been pointed out that one of the main benefits of using a proactive system is a smoother (i.e., more regular) usage of bandwidth, as the replications are more evenly distributed over time [1], [15], [6]. In the following, we present an analysis of the burstiness of reactive models, in order to quantify the difference with the

Fig. 4. Ratio between the maximum number of concurrently active replicas and initial number of replicas as a function of the replication rate ρ , in proactive, for $n=2$ and $\lambda=2$. As ρ increases, the ratio also increases.

bandwidth usage of proactive models.

Figure 5a shows the comparison between the replica creation over time, in reactive (left) vs. proactive (right). Figure 5b shows the corresponding histograms of replica creation every $1/\rho$. Each point in Figure 5a represents a reactive repair or a proactive replication and the two plots are zoomed in random points, representative of the entire curve. In the combinations of the parameters that are typical of real applications ($\mu \leq \lambda$), the maximum number of replicas per $1/\rho$ is about 5 and the most recurrent case is 1 replica created each interval, as the histogram has a wide spike around the value 1. The burstiness increases as n , λ and μ increase, because as the three parameters increase, there are more nodes that fail and get repaired per unit of time. In Figure 5a, a highly bursty pattern in the replica creation for the reactive model would correspond to many points connected by lines with very high slopes. Our data suggest that the distribution of replica creation over time shows a moderately bursty pattern. The burstiness occurs rarely, and even when it occurs, the reactive bandwidth usage has peaks of roughly 5 times the usage in reactive.

These results are most likely due to the fact that failures occur randomly and, since we are only considering uncorrelated failures, their distribution in time is random and relatively even. Since it is unlikely that bursts of uncorrelated failures occur within a short amount of time, bursts of repairs are not observed either. Additionally, when the failure rate λ increases, both μ and ρ must also be increased for the expected lifetime not to decrease. This implies that the time intervals $1/\rho$ become shorter. In other words, in proactive the replication must be more frequent and in reactive the probability of having many failures and repairs within each $1/\rho$ interval decreases, each interval being shorter. Thus, even when the failure rate increases, the reactive model does not appear to have a dramatically different bandwidth usage pattern compared with proactive.

VI. CONCLUSIONS

The optimal strategy of replication (reactive vs. proactive) to ensure durability in replicated storage systems is still an

open problem. In this work, we considered a basic reactive system in which both times between failures and times to repair are exponentially distributed, and a proactive system in which times between failures are exponentially distributed and times between replications are constant (i.e., replication occurs at a fixed periodic rate, independent of failures).

We derived the analytical formula for the variance of the lifetime for the reactive model. We found that the standard deviation of the lifetime is nearly the same as the expected value. From the simulations, this appears to be the case for proactive too. This led us to surmise that the lifetime itself is exponentially distributed. We confirmed this with Chi Square Goodness of Fit tests, at a stringent significance level of 0.001.

We also presented a quantitative comparison of reactive and proactive strategies in replicated storage systems, particularly in terms of maximum and average resource usage. This allows one to choose the best strategy when designing a system, considering the specific constraints existing in the system itself, in particular the available bandwidth and storage. In general, whenever a system cannot tolerate any spikes in bandwidth greater than about 5 times the average available bandwidth, then a proactive strategy is likely the more appropriate. On the other hand, whenever storage constraints prevent the commitment of multiple times the number of initial replicas, a reactive model is the better choice.

APPENDIX

We derive the expression for the variance of the lifetime for a reactive system with exponentially distributed failure times and repair times. We begin with an important result first derived in [3], which analyzed the first passage time distribution for a parallel exponential system with repair. This system is a generalization of the reactive system we analyze. The generic formula for the variance of a random variable D (which corresponds to our “lifetime”) in a such a system is given by:

$$Var(D) = E^2 + 2E \sum_{\alpha} a_{\alpha} b_{\alpha} + 2 \sum_{\alpha} (a_{\alpha}^* - a_{\alpha}) b_{\alpha}^2$$

- E is the expected value of the random variable D
- $a_{\alpha} = (-1)^r$
- $a_{\alpha}^* = \prod_{j=1}^r \frac{\mu_{i_j}}{\lambda_{i_j}}$
- $b_{\alpha} = \frac{1}{\sum_{j=1}^r \lambda_{i_j} + \mu_{i_j}}$
- $\sum_{\alpha}$ denotes the summation over the $\binom{n}{r}$ subsets of size r from 1 to n .
- $\lambda_1, \dots, \lambda_n$ are the parameters of the exponential failure time distributions and $\mu_1, \dots, \mu_n$ are the parameters of the exponential repair time distributions.

From this one can derive an expression that explicitly uses typical parameters of a replicated storage system. Expanding the expression, we have:

$$Var(D) = E^2 + 2E \sum_{r=1}^n \sum_{i_1 < i_2 < \dots < i_n} (-1)^r \frac{1}{\sum_{j=1}^r (\lambda_{i_j} + \mu_{i_j})} + 2 \sum_{r=1}^n \sum_{i_1 < i_2 < \dots < i_n} \left[\prod_{j=1}^r \frac{\lambda_{i_j}}{\mu_{i_j}} - (-1)^r \right] \frac{1}{\sum_{j=1}^r (\lambda_{i_j} + \mu_{i_j})^2}$$

In our case, we consider one value of λ and one value of μ for every state, as we assume the failure and repair rates do not vary with the number of replicas left in the system. Thus, instead of $\lambda_1, \dots, \lambda_n$ and $\mu_1, \dots, \mu_n$ we have λ and μ . We

(a) Reactive shows a moderately bursty pattern. For proactive, replications occur at a constant rate by design.

(b) Corresponding histograms of replica creation every $1/p$ intervals

Fig. 5. Replica creation in reactive (left) and proactive (right) models, with $n=6$, $\lambda=1$, $\mu=2.756$ and $\rho=5.05$. For proactive, by design exactly 1 replica is created every $1/p$ interval, while for reactive, this depends on the frequency of the failures. The most common case, however, is about 1 repair per $1/p$ interval.

also define $\gamma = \frac{\mu}{\lambda}$. Given these simplifications, the expression becomes:

$$\text{Var}(D) = E^2 + \frac{2E}{\lambda(1+\gamma)} \sum_{r=1}^n \sum_{i_1 < i_2 < \dots < i_n} \frac{(-1)^r}{r} + \frac{2}{\lambda^2(1+\gamma)^2} \sum_{r=1}^n \sum_{i_1 < i_2 < \dots < i_n} \frac{\gamma^r - (-1)^r}{r^2}$$

We use the formula for the expected value of D that was

$$\text{derived in [14]: } E(L) = \frac{1}{n\lambda} \sum_{k=0}^{n-1} \sum_{j=0}^k \binom{n}{j} \gamma^{k-j}$$

Substituting $E(L)$ for E and $\text{Var}(L)$ for $\text{Var}(D)$, we have

$$\text{Var}(L) = \frac{E^2(L)}{\lambda^2(1+\gamma)^2} + \frac{2E(L)}{\lambda(1+\gamma)} \sum_{r=1}^n \binom{n}{r} \frac{(-1)^r}{r} + \frac{2}{\lambda^2(1+\gamma)^2} \sum_{r=1}^n \binom{n}{r} \frac{\gamma^r - (-1)^r}{r^2} \quad \blacksquare$$

REFERENCES

- [1] E. S. Andreas, A. Haeberlen, F. Dabek, B. G. Chun, H. Weatherspoon, R. Morris, M. F. Kaashoek, and J. Kubiatowicz. Proactive replication for data durability. In *Proc. of the 5th Intl. Workshop on Peer-to-Peer Systems IPTPS*, 2006.
- [2] R. Bhagwan, K. Tati, Y. chung Cheng, S. Savage, and G. M. Voelker. Totalrecall: System support for automated availability management. In *Proc. of the 1st ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 337–350, 2004.
- [3] M. Brown. The first passage time distribution for a parallel exponential system with repair. In *Reliability and Fault Tree Analysis*, 1974.
- [4] A. Datta and K. Aberer. Internet-scale storage systems under churn - a study of the steady state using markov models. In *IEEE International Conference on Peer-to-Peer Computing (P2P)*, pages 133–144, 2006.
- [5] A. Duminuco, E. Biersack, and T. En-Najjary. Proactive replication in distributed storage systems using machine availability estimation. In *Proc. of the 2007 ACM CoNEXT Conference*, pages 27:1–27:12, 2007.
- [6] A. Duminuco, E. Biersack, and T. En-Najjary. Proactive replication in distributed storage systems using machine availability estimation. In *Proc. of the 2007 ACM CoNEXT Conference*, pages 27:1–27:12, 2007.
- [7] A. Gharaibeh and S. Al-Kiswani. Thriftstore: Finessing reliability trade-offs in replicated storage systems. In *IEEE Transactions on Parallel and Distributed Systems*, Vol. 22 No. 6, 2011.
- [8] B. gon Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, M. F. Kaashoek, J. Kubiatowicz, and R. Morris. Efficient replica maintenance for distributed storage systems. In *Proc. of the 3rd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 45–58, 2006.
- [9] A. Haeberlen, A. Mislove, and P. Druschel. Glacier: Highly durable, decentralized storage despite massive correlated failures. In *Proc. of the 2nd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2005.
- [10] S. Jain, R. Prinja, A. Chandra, and Z.-L. Zhang. Failure classification and inference in large-scale systems: A systematic study of failures in planetlab. Technical report, Department of Computer Science, University of Minnesota, 2008.
- [11] F. Junqueira, R. Bhagwan, R. Hevia, K. Marzullo, and G. M. Voelker. Surviving internet catastrophes. In *Proc. of the Usenix Annual Technical Conference*, 2005.
- [12] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *ACM SIGARCH Computer Architecture News*, volume 28, 2000.
- [13] Q. Lian, W. Chen, and Z. Zhang. On the impact of replica placement to the reliability of distributed brick storage systems. In *Proc. of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS)*, pages 187–196, June 2005.
- [14] S. Ramabhadran and J. Pasquale. Analysis of long-running replicated systems. In *Proc. of the 25th IEEE Annual Conference on Computer Communications INFOCOM*, 2006.
- [15] H. Weatherspoon, H. Miranda, K. Iwanicki, A. Ghodsi, and Y. Busnel. Gossip: Gossiping over storage systems is practical. In *ACM SIGOPS Operating Systems Review*, volume 41, 2007.