Black box replication: Breaking the latency limits

Assaf Natanzon * Alex Winokur † Eitan Bachmat ‡

Abstract

Synchronous replication is critical for today's enterprise IT organization. Synchronous replication is mandatory by regulation in several countries for some types of organizations, including banks and insurance companies. Synchronous replication technologies have been available for a long period of time, but due to speed of light and maximal latency requirements, synchronous replication is usually limited to a distance of 50 miles. Flight data recorders also known as black boxes, have long been used to record the last actions which happened in airplanes at times of disasters. We present an integration between an Enterprise Data Recorder and an asynchronous replication mechanism, which allows breaking the limits that light speed imposes on synchronous replication.

1. Introduction

Synchronous replication is mandatory for some organization, due to rules imposed by regulations and also due to the desire to have no data loss in case of a disaster. Even a small amount of data loss can have significant monetary effects on an IT organization. In *synchronous replication* every write is acknowledged before more data is replicated. Due to the speed of light limitation on information transfer and the typical two round trip SCSI write protocol, synchronous replication is usually limited to a 50 mile distance. Beyond such a distance the performance slowdown incurred by synchronous writing becomes prohibitive. The distance limitations, force enterprises to build data centers within a 50 miles distance, thus, even in cases where the enterprise already has several data centers with larger distances, the enterprise is forced to build new data centers. Because Synchronous replication is so limited by distance, it may in some cases be impossible to protect the systems from a regional disaster like a hurricane, or electricity shutdown like the northeast blackout of 2003 which affected an estimated 10 million people in Ontario and 45 million people in eight U.S. states.

Because of such limitations large enterprises may create two separate remote copies of the data, a Synchronous copy at a 50 mile distance, and another asynchronous copy at a larger distance. Thus the enterprise will have no data loss when a local disaster occurs, but will lose data when a regional disaster happens. The problem of solving the loss of communication during a disaster is long solved for airplane communication. *flight data recorder* Popularly referred to as a "black box", is an electronic device employed to record any instructions sent to any electronic system on the aircraft. The data recorded by the FDR is used for accident investigation, as well as for analyzing air safety issues, material degradation and engine performance. Due to their importance in investigating accidents, these ICAO-regulated devices are carefully engineered and stoutly constructed to withstand the force of a high speed impact and the heat of an intense fire [25].

Efforts to require crash-protected flight recorders date back to the 1940s. The introduction of Flight Data Recorders (FDR),however, experienced many delays.

* Assaf Natanzon, EMC Corp., Ramat Gan, Israel and department of Computer Science, Ben-Gurion University, Beer-Sheva, Israel, 84105. assaf.natanzon@emc.com

† Alex Winokur, Axxana, Tel Aviv, Israel. Alex.Winokur@axxana.com

‡ Eitan Bachmat, Department of Computer science, Ben-Gurion University, Beer-Sheva, Israel, 84105. ebachmat@cs.bgu.ac.il

That is because technology could not match the design requirements of a unit that could survive the forces of an aircraft crash and the resulting fire exposure until 1958, when the world authorities approved a minimum operating requirements for an FDR. Early devices had very limited recording capabilities, only five analog parameters describing flight conditions of the aircraft, i.e., heading, altitude, airspeed, vertical accelerations, and time were embossed onto a metal foil (Incanol Steel), which was used only once. Second generation of recorders that used magnetic tape as the recording medium were introduced at the 60s. The first product to use this new technology was the cockpit voice recorder (CVR). In 1965, all commercial operators were mandated to install a CVR, which would retain the last 30 minutes of crew voice communications and noise within the cockpit environment.This same tape technology (recording data in a digital format) was expanded to the FDR. The Solid State Flight Data Recorder (SSFDR) became commercially practical in 1990. Solid State refers to storage of data in semiconductor memories or integrated circuits, rather than using the older technology of electromechanical methods of data retention. More on the history of FDRs can be found in [23]

In this paper we present a method for breaking the limitations imposed by the speed of light, by integrating an *Enterprise Data Recorder* (EDR) and an asynchronous replication mechanism. An Enterprise data recorder is a disaster proof storage system, which is capable of surviving extreme conditions, similar to the conditions an FDR can survive. Leveraging the EDR capabilities with asynchronous replication, the system guaranties that the user can replicate to a remote site at any distance, asynchronously, adding no additional latency to its production application, and having no data loss in case of a disaster. The solution which is described in this paper, the Phoenix System RP, is generally available, in conjunction with an EMC VNX, CLARiiON.

The EDR is equipped with cellular transmission and allows recovery of the data even at disaster times. Newer wireless networks are following the general trend towards more decentralized structures. The cellular telephone network is centralized at a smaller neighborhood scale in major cities. Thus the destruction of antenna sites typically only reduces service in a limited area. For example, McCaw Cellular lost 2 of its 400 cell sites in the Northridge earthquake, resulting in only isolated service disruptions.15 Emerging wireless technologies such as Wi-Fi serve even smaller areas only a few hundred feet in diameter [27]

Users may also achieve benefit from an EDR based solution when replicating at short distances which allow in principal synchronous replication. When replication is synchronous the production storage array response time is dependent on the replica storage array response time. A large load imposed by an application running on the replica site may have significant impact on an application running on a local storage array which is being synchronously replicated.

The paper is divided as follows:

The next section presents some background on RecoverPoint, a replication solution which is capable (among other options) of asynchronous replication.

Section 3 describes the EDR solution, its structure and compliance tests.

Section 4 discusses the integration of the EDR with the RecoverPoint based asynchronous replication, describing several modes of operation.

In section 5 we analyze the performance of the combined product.

Section 6 describes related work,

Section 7 summarizes our main conclusions and discusses future work.

2. Description of a network based, continuous data replication solution

In this section we describe Recover-Point, which is an example of a network based continuous data protection and replication scheme. The replication system whose features we integrated with EDR is composed of two pieces, A *splitter* and a *data protection appliance*.

The splitter, intercepts the I/O arriving at the data path, and mirrors the I/O to both the network data protection appliance, and the original storage target. The splitter is installed either at the replicated host, or inside a fabric switch, or in a storage array.

The splitter is a fairly simple and portable piece of code. The splitter can intercept data operations at several logical levels. The system implements a splitter for

replication of block level devices which intercepts I/O at the SCSI layer.

The data protection appliance is a physical or virtual appliance, which exposes a SCSI target. It receives I/O coming from a splitter and replicates them over a WAN to a data protection appliance at the replica site. The replication appliance at the replica site manages the remote copy.

The replication appliance is a complex piece of code. The protocol between the replication appliances utilizes data compression and data de-duplication to optimize WAN traffic. For more on de-duplication methods, see [2, 1, 22]. The replica data protection appliance manages a *journal* of the data at the replica site as well as a full copy of the data.

The network based replication system, allows creation of consistency groups consisting of several volumes. Since several splitters can send the I/O operations to the same data protection appliance, write order fidelity can be achieved between volumes from separate storage arrays at a single data protection appliance.

The journal is a log containing the sequence of changes that happened to each of the volumes in the consistency group. There are two streams of data in the journal, a *redo log*, containing changes which occurred to the production volume and not yet applied to the replica volume, and an *undo log* containing a list of changes which undoes the latest changes in the replica volumes.

Each entry in a journal contains, apart from the I/O data itself, the following I/O meta data:

1) volume ID.

2) The I/O block offset within the volume.

3) The I/O length.

The meta data and the data of the journal may be stored in different streams. The meta data in the journal also contains information whether the point in time after the write, described by the meta data, is consistent. It may also contain some user created strings describing the point in time, e.g., a bookmark. The user may create a bookmark whenever desired, each bookmark is a consistent point in time. If the user desires to create an application consistent point in time, the user needs to put the application in a consistent mode like Oracle hot backup mode or Microsoft VSS, and request the system to create a bookmark while the application is in the consistent mode. Creation of a bookmark is just a logical procedure which adds messages with the bookmark info into the I/O stream, and thus bookmark creation typically takes less than a millisecond.

natanzon-RPDiagram.jpg

Figure 1. Description of the system set-up.

The full replica copy, together with the journal provides a Continuous Data Protection (CDP) system, allowing the user to access any point in time. Another example of a CDP system, which differs substantially from the one we implemented, is presented in [16]. A system which offers multiple point in time views can also be found in [17]. We now describe the operation of the system upon the arrival of a write I/O. The system is described in figure 1.

The I/O flow is as follows:

1) A host application generates an I/O.

2)The splitter, which is located somewhere in the data path, intercept the I/O. The Splitter checks whether the I/O belongs to a volume which needs to be replicated.

3) If the I/O should be replicated, the splitter mirrors the I/O to both the data protection appliance and the original I/O target, i.e., the production storage array. The splitter works at the SCSI protocol level, and thus it intercepts SCSI write commands and sends the SCSI write command to a target exposed by the data protection appliance at the production site, and the original SCSI target of the command.

4) The splitter returns the SCSI status of the write only after receiving status from both the original I/O target and the data protection appliance target.

The protocol described above between the splitter and the data protection appliance basically states that the splitter mirrors the I/O synchronously between the production volume and the production site data protection appliance. The data protection appliance controls the I/O termination status. This allows the data protection appliance to implement any replication strategy desired.

A synchronous replication strategy is implemented when the production appliance acknowledges the splitter I/O (by returning a SCSI status) after the I/O is sent to the replica site. We implemented two options for synchronous replication strategies. In one implementation the production appliance acknowledges the I/O to the splitter immediately after the I/O reaches the cache of the replica appliance. The I/O in this case is not yet safely written at the storage array at the replica site, but any disaster at the production site will induce no data loss as all the data is cached at the replication data protection appliance.

The second implementation is synchronous replication to the replica site storage array. In this implementation the production appliance acknowledges the splitter only after the I/O is persistently stored in the journal at the replica site at the storage array.

An Asynchronous replication strategy is implemented by decoupling the protocol between the local and remote data protection appliances from the acknowledgment to the splitter. The local data protection appliance immediately acknowledges the I/O to the splitter and only then the appliance asynchronously sends the I/O to the replica data protection appliance.

The asynchronous replication may be semi-synchronous, i.e., all I/O are sent to the replica site, or it can reduce to snapshot shipping methodology, grouping large sets of changes together and removing *hot spots* from each changed set sent to the replica site. Hot spots are locations on a volume being replicated which are frequently accessed. When shipping a snapshot to a replica site we only need to transfer the latest data version which was written to a specific location on the storage device. If data was written to the same location on the storage device 10 times before the snapshot is created, the replication engine will only need to transfer one piece of data, saving significant amount of WAN traffic and also significant amount of traffic to the replica storage array, [13].

3. The Enterprise Data Recorder (EDR)

Until recently creating an EDR for a storage array was virtually impossible, storage spindles moving parts,are not be able to sustain shock or pressure. SSD drives on the other hand have no moving parts and thus can sustain severe conditions. Until recently solid state devices where not large enough and reliable enough to store the amount of data needed to backup the last minutes of a storage device. Another barrier of building an EDR is the limited bandwidth of cellular networks which only recently evolved to usable bandwidth. The EDR system created consists of an electronic box with an SSD drive, a battery and an electronic board including cellular transmission. The metal cylinder enclosure protects the box against catastrophes such as a building collapse and so on. The enclosure contains cell antennas and a water cooling pipe to cool the system. the exterior of the EDR is coated with heat-resistant bright orange paint for high visibility in wreckage, the system is also coated by a fire resistant fabric. The fabric is made of Basalt fiber a technology used by the Russian space program. The internal part is wrapped in this blanket a couple of times (the blanket wraps around a wooden box,which is a very good heat insulator and often used in fire protected safes). The pipe wraps between the blankets a number of time such that the temperature gradient of this long pipe brings it from the outside temperature to the inside temperature (most of the cables melt anyway on the external side of the blanket).

Table 1 describes the network categories, and the amount of data which can be recovered from an EDR using a celular transmision fom the relevant category within 6 hours, which is a reasonable RTO (Recover Time Objective). The figures above are in lab environment. In a real life network, at most one can get around 40-50 percent of the figures mentioned. For example, in Category 5 network, we should assume not more than 15-20GB in 6 hours. In practice, we see most of the time much less. in the EDR creates we used an (MC8790) modem which can upload today maximum of 2Mbps which is the equivalent of Category 5 above, what usually exists in good networks. There is a newer Firmware version for the above modem, which can bring us to an upload rate of 5.76 Mb/s, but it was not stable enough for our testing. Networks with category higher than 5 are very rare these days. Since the solution is integrated which a nearly synchronous continuous replication so-

Figure 2. EDR system

Figure 3. EDR before and after extreme fire

lution, the average data which will need recovery is data from the last copule of minutes.

Table 1. Network Categories

Cathegory	MB/Sec	Total GB in 6 hours
Category 1	0.73	15.40
Category 2,3	1.46	30.80
Category 4	2.93	61.80
Category 5	2	42.19
Category 6	5.76	121.50
Category 7,8	11.5	242.58
Category 9	23	485.16

The EDR system is photographed at figure 2
The EDR is capable of surviving:

1. Direct flames at 2000 F (1093 C) for one hour followed by up to 482 F (250 C) for an additional 6 hours.
2. Pressure of 5000 lb (2268 kg) of weight.
3. Pierce force of 500 lb (227 kg) rod with cross section of .042 dropped from a 10 ft (3 m) height.
4. 30FT of water pressure.
5. Shock of $40G$.

The resilience features were all lab tested. Figure 3 shows the EDR after one hour of 2000 F. The SSD was still operational after the application of the above forces and heat.

4. EDR integration with the asynchronous data replication scheme

We describe the design of the system which integrates the EDR with the asynchronous replication solution. The EDR is connected to a controller which manages it. the controller is a standard off the shelf x86 server with an HBA attached to it, which is connected via a SAN (storage area network) to the EDR box and to the replication appliance. In theory the controller could have been implemented as part of the appliance but in order to simplify the system and to reduce dependencies the controller was implemented as a separate appliance. The controller is configured to expose a SCSI LUN for each replicated storage device and is configured by the replication appliance.

The system architecture design is describes in figure 4.

The IO flow when no errors occur is as following:

1. An application server generates a write IO.
2. The splitter intercepts the IO and sends it to the replication appliance.
3. The replication appliance sends the IO to the EDR controller.
4. The EDR controller persistently stores the IO data in the EDR.
5. Once the EDR controller receives a SCSI status from the EDR, it returns the same status to the replication appliance.

Figure 4. system design

6. The replication appliance sends the status to the splitter.
7. The splitter sends the IO to its original target.
8. Asynchronously, the replication appliance sends the data to the replica appliance,which in turns journals it as described above.

4.1 configuration and control of the EDR

The replication appliance configures the EDR controller using a SCSI based protocol which we developed. The appliance configures which volumes the EDR controller has to expose and what are the configured consistency groups. For each consistency group the EDR controller creates a cyclic journal of a preset size configured by the replication appliance. For example for each consistency group a journal of 4GB is allocated. The IO write operations arriving to LUNs which are part of the consistency groups are written to the journal log of the relevant consistency group. Since the replication appliance is aware of the log size it allocated for each consistency group, the replication appliance will assure that the lag between the production site write IO and the last IO arrived to the replica site is less than the size allocated for the consistency group cyclic log allocated at the EDR. As a result, all the IO which have not yet arrived at the replica site are guaranteed to be stored in the EDR.

The user can control the behavior of the system in cases where the system cannot achieve the lag which is dictated by the size allocated for the consistency group at the EDR. This may happen due to limited WAN resources or various problems with the remote storage resources at the replica site.

1. The user can configure the system to delay IO when the maximal lag is reached. In such a case, the system starts delaying acknowledgment of IO operations and assures the lag remains lower than the bound set by the replication appliance. the IO are never delayed for more than a couple of seconds to assure that hosts will not get timeouts due to the system being un-responsive.
2. The user can configure the system not to delay the production application. In this case when the maximal lag is reached, the replication appliance configures the EDR controller not to enforce the limited journal size, and the EDR space starts to fill. Once the lag is less than the original configured size again, the replication appliance re-configures the EDR controller to enforce the limited log size, and the rest of the EDR storage is freed. In case an unlimited log size is set by the user, the EDR may also get full.

When EDR storage is full the system can either stop being synchronously protected, or it may stop allowing new IO to arrive causing applications to cease working. The protection level the user sets controls whether, the system will stop serving IO operations when EDR is full, or the system will stop being protected synchronously and just warn the user that the system is currently not protected.

4.2 Temporary disaster handling

With current synchronous replication technologies, the replication may cease to function in two type of disasters,

1. Transient failures like a WAN failure which may cause synchronous replication to stop functioning.
2. Permanent failures, which include real disasters, such as a flood or fire, which cause the whole production site or major parts of it to stop functioning.

The EDR solution allows the system to continue synchronous protection when a temporary WAN failure occurs. Today enterprises are usually configured to stop replicating synchronously in case of a WAN disaster. The reason is that WAN disasters are much more likely than local hardware disasters. Storage systems and networks are usually created with high redundancy, thus a full outage is not likely. However, WAN failures are

much more common, either due to maintenance or some other problems. When the system discovers a WAN failure, the replication appliances will automatically move all relevant consistency groups, to a mode where the log size is unbounded, and thus if the WAN failure ends before the EDR is full the user will remain protected synchronously throughout the whole process.

One of the main problems with current synchronous replication approaches is that since users do not want to have their systems stop functioning during a WAN disaster, they configure the synchronous replication systems to stop replicating when a WAN disaster occurs. Usually a WAN outage of 10-20 seconds will cause the system to stop replicating. There is not much more the users can do, because if after an outage of 10-20 seconds the IOs will not be acknowledged to the hosts and the applications will crash and stop functioning.

The assumption is that if there is a real disaster the systems will stop at around the same time as the WAN. The problem with this approach is that if the disaster is a rolling disaster and the WAN is lost say five minutes before the rest of the system is affected, data will be lost.

As discussed in the previous subsection, in the presence of an EDR, the user determines the behavior of the system in case the EDR storage is full, but the EDR storage is large enough to handle hours of WAN outage. For this reasons it may also be justified to use EDR based replication coupled with synchronous replication and thus increase data availability even further.

4.3 recovery from a full disaster

Once a full disaster causing the production site to be completely lost, the system can recover all the data to the replica site, for this reason the system has a module called the *EDR recoverer*, the EDR recoverer is also based on standard hardware. There are two possible recover modes:

1. Recovery directly from the EDR: In this mode the EDR itself is brought from the disaster zone. Previous large scale disasters show that it would take a couple of days to recover the black box from a disaster zone. The recoverer verifies that the asynchronous image available at the replica site can be completed to a synchronous image. The EDR may not have a synchronous image if the EDR got full and the user configured the system to continue working even if the EDR is full. In that case, the EDR will have an indication that a consistency group is out of sync. The recovered then indicates to the replication appliance to provide the latest asynchronous image it has for the consistency group, and replays all the writes from the log to the replica volumes.
2. The EDR system has cellular transmition capabilities. The EDR recoverer can connect to the EDR over cellular communication and download the relevant data changes which need to be applied to the non updated image that the asynchronous replication mechanism provided for each consistency group.

   Previous disasters showed that cellular networks tend to be more reliable than wired networks during disasters, thus the cellular transmission unit may allow recovery of all or part of the data before the EDR itself is recovered.

5. Experimental results

5.1 Experimental setup

For hardware, we used two Recover-Point gateways with 2 GEN4 data protection appliances one appliance at each site. The interconnect between the two sites we used was is a $1Gb$ TCP/IP connection over Ethernet when we replicated using the EDR and $1Gb$ fibre channel connection when we tested synchronous replication without an EDR . The appliances each have $8192MB$ of RAM. Each appliance has 2 quad core CPU a QLogic QLE2564 quad-port PCIe-to-8Gbps Fibre Channel Adapter.

The replication gateways are appliances which are connected to an Axxana collector (EDR gateway), which have identical hardware as the replicaiton appliances.

Each appliance in the pair is connected to a separate CLARiiON CX4-480 storage array with 30 FibreChannel-attached disks, configured as 6 separate RAID5, $4 + 1$ RAID groups. We configured a consistency group replicating 12 production volumes. The 12 production volumes were created on 4 separate RAID5 4+1 groups, 3 volumes were created on each RAID group. The journal was striped over two volumes from two separate RAID groups. We configured the replica site in an identical way. When simulated synchronous replication over distance we use an ANUE fiber channel emulator to emulate a 25 mile distance WAN with limited 1Gbit bandwidth.

Figure 5. EDR based replication response time

Figure 6. Synchronous replication over 25 mile distance

5.2 Performance metrics for synchronous replication over distance

We tested the system with EDR protection and compared it to synchronous replication over various distances. As expected, replication using an EDR can achieve performance similar to synchronous replication at zero (close proximity) distance.

Figure 5 exhibits the response time of an EDR based system replicating asynchronously to a distant location. We ran the system with 8KB IOs which simulates the behavior of online transaction processing . The figure describes the response times compared to the number of IO operations per second. The system utilized Lempel-Ziv based compression algorithm which yielded compression ration of over 2, allowing the system to support over 16000 IO operations per second on a 1Gb TCP/IP WAN. We simulated a write only environment, as read have less effect on replication. In real systems customer data shows that between 75-80% of the IO operations are read IOs, but since there are dependencies between read and write commands, high response time to write commands will reduce the overall performance of both reads and writes in the system.

Figure 6 describes behavior of a synchronous replication system and its response times when replicating over larger distances. The figure describes a distance of 25 miles over 1Gb fibre channel link, which is still considered a distance where synchronous replication can still be reasonably performed. We observe that the IO response time significantly increases and at 5000 IO/sec the latency reached a significant response time of 5 milliseconds, compared to less than 2 milliseconds at the same IO rate when using an EDR at zero distance as seen in figure 5.

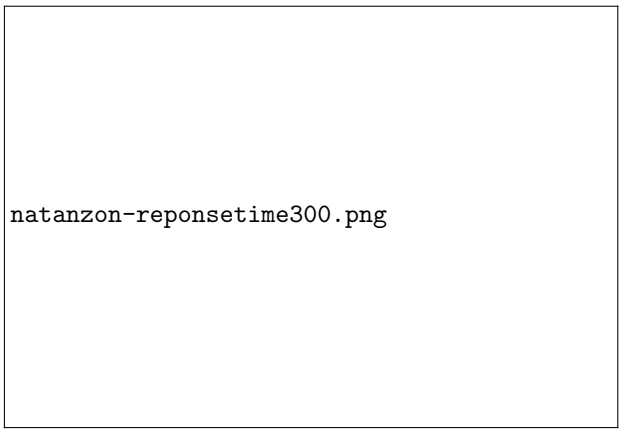

Figure 7. Synchronous replication over very large distance (300 mile)

Figure 7 describes the behavior of synchronous replication at a distance of 300 miles which is not considered relevant to synchronous replication using standard methods, since it typically causes serious application delays, as we can see even at less than 3000 IO/sec the IO response time is over 15 milliseconds which is not acceptable for most applications.

5.3 Synchronous replication when replica storage is loaded

Synchronous replication is heavily dependent on the replica storage response time, many customers have several data centers and replicate between them. a storage system which is a target for synchronous replication, may also be the production storage system for another volumes. Synchronous replication waits until replica storage acknowledges the IO before the production system acknowledges the IO operation to the application, a temporary heavy load at the replica site from an application like a data warehouse running at the replica site, may significantly increase the remote storage response time, causing the performance and response time of an application running at the local site and replicated synchronously to increase significantly. We tested storage response time when replicating to a temporarily loaded storage when replicating synchronously and replicating virtually synchronous by using asynchronous replication with an EDR.

Figure 8 describes a typical storage behavior, response VS. load. We tested a CLARiiON CX-4 with 120 FC drives. please note that the test results may be different depending on the model of the storage array,the amount of the back-end devices, the type of the back-end devices and the storage configuration. We tested the storage with TPC-C based load, TPC-C is an on-line transaction processing benchmark created by the Transaction Processing Performance Council which is a non-profit organization founded in 1988 to define transaction processing and database benchmarks [26]. TPC-C is an OLTP benchmark which simulates transactional data based behavior. The tests utilize the storage with the following transaction load:

Table 2. TPC-C storage workload

IO Type	percentage of load
Random read hit	20%
Random read miss	45%
Sequential read	10%
Random write	15%
Sequential write	10%

As we can observe as the storage load increases the average IO response time increases and at 40000 IOPS the response time reaches about 7 milliseconds.

natanzon-RPTPCC.png

Figure 8. Synchronous replication over distance

response_time_remote_loaded.png

Figure 9. Production storage response time when replicating synchronously to a loaded remote storage

Figure 9 describe response time of synchronous replication at zero distance, VS. remote storage load, the green line describes the IO response time of local storage when remote storage already runs a 15000 IO/sec TPCC load, the red line describes the load when remote storage is utilized by 30000 IO/sec TPC-C load and the blue line describes the load of the system when remote storage is utilized with 45000 IO/sec TPC-C load, as we can see the load on the remote storage has significant impact on the response time of the production storage.

When using an EDR based replication, the replication results will be identical to the results of figure 5. The replica storage has no impact on the latency of the production storage, the overall response of the replica storage of course increases as the load on the replica

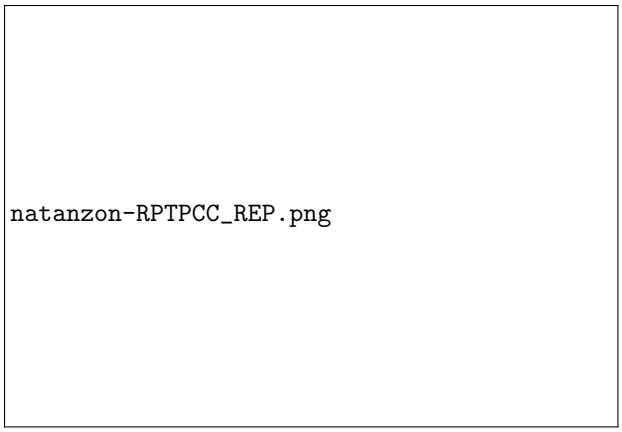

Figure 10. remote storage TPC-C response time when receiving a 10000 IOPS load from a production storage

storage is higher when replication is on, in this case the overall TPC-C IOPS of the replica storage are lower, figure 10 describes TPC-C response time when the storage running TPC-C is also a replica for a different storage array, where the source of the replication generate 10000 IO/sec.

5.4 recovery testing

in this section we tested the recovery using a cellular connection. We tested recovery in several use cases, since the asynchronous replication is continuous, in most cases the amount of data lost in case of disasters will be very small. Thus, we simulated the amount of time it would take to recover 10 seconds of data loss. We used typical customer loads to estimate the amount of data that will be lost during 10 second. [I still need to add a graph here, from customers]

6. Related work

There are several types of on-line data replication techniques. Data replication techniques typically come in one of two flavors, either host based or array based. The `Coda` system is one early example of a replicated file system [9]. In Coda, the clients of a file server are responsible for writing to multiple servers. This approach is essentially synchronous logical-level mirroring. By putting the responsibility for replication on the clients, Coda effectively off-loads the servers, and because clients are aware of the multiple servers, recovery from the loss of a server is essentially instantaneous. However, Coda is not designed for replication over a WAN. If the WAN connecting a client to a remote server is slow or congested, the client will feel a significant performance impact.

The infrastructure needed to handle disasters is discussed in [8].

Array based replication includes both synchronous replication and asynchronous replication technologies. For a nice introduction to remote mirroring techniques see, [7]. Efficient asynchronous replication schemes are described in [20, 21]. For descriptions of commonly available commercial implementations, which have both synchronous and asynchronous modes, see, [13, 4, 3]. However, all these systems do not dynamically move between the two modes.

7. Conclusion and future work

We presented the design and implementation of a solution to the problem of synchronous replication over large distances without severe performance impact. Due to physical speed of light limitations, the solution proposed is the only one possible. The solution does not slow down significantly, normal system operations, and has added benefits also in the case of synchronous replication at short distances. The solution incorporates a physical device similar to the ”black boxes” employed in aircrafts to save flight related data. the device is integrated into an existing framework for data replication, both synchronous and asynchronous.

FDR changed the ability to investigate aviation disasters, likewise, EDR based replication systems can significantly improve the recoverability of data after disasters. Today synchronous replication is limited significantly by the speed of light, and is much more exposed to transient disasters, like WAN disasters. EDR systems can significantly increase the availability and reliability of synchronous replication.

For future work, we may investigate different integration options between and EDR and the replication module. We can, for instance, have the splitter send the IO operations directly to the EDR and the replication appliance at the same time saving another hop and reducing the latency.

8. acknowledgments

The authors wish to thank Lev Ayzenberg for helpingwith test and coding.

References

[1] L. Aronovich, R. Asher, E. Bachmat, H. Bitner, M. Hirsch, S.T. Klein: The design of a similarity based deduplication system. SYSTOR 2009: 6.

[2] B. Zhu, K. Li and H. Patterson, Avoiding the Disk Bottleneck in the Data Domain Deduplication File System, Proc. of FAST 08, the 6th USENIX Conf. on File and Storage Technologies, 279-292, 2008.

[3] A. Azagury, M. Factor, and W. Micka, Advanced functions for storage subsystems: Supporting continuous availability. An IBM SYSTEM Journal, 2003.

[4] EMC Symmetrix Remote Data Facility. http://www.emc.com/.

[5] S. Ghemawat, H. Gobioff and S. Leung, The google file system. In Proc. of ACM SOSP (Oct. 2003), pp. 2943.

[6] D. Hitz, J. Lau, M.A. Malcolm. File System Design for an NFS File Server Appliance. Proceedings USENIX Winter 1994 Conference. pp. 235-246.

[7] M. Ji, A. Veitch and J. Wilkes Seneca: remote mirroring done write Proceedings of USENIX Technical Conference (San Antonio, TX), pages 253268, June 2003. USENIX, Berkeley, CA.

[8] K. Keeton, C. Santos, D. Beyer, J. Chase and J. Wilkes, Designing for disasters. In Proc. of USENIX FAST (Berkeley, CA, USA, 2004), USENIX Association, pp. 5962.

[9] J.J. Kistler, Disconnected Operation in a Distributed File System. Technical Report CMU-CS- 93-156. School of Computer Science, Carnegie Mellon University, 1993.

[10] S.A. Leung, J. Maccormick, S.E. Perl and L. Zhang, Myriad: Cost-effective disaster tolerance. In Proc. of USENIX FAST (Jan. 2002).

[11] B. Liskov, S. Ghemawat, R. Gruber, P. Johnson, L. Shrira, and M. Williams, Replication in the Harp file system. In Proc. 13th SOSP, published as Operating Systems Review 25(5):226238, Oct. 1991. ACM.

[12] Matthews , J., Roselli, D., Costell, A., Wang, R., and Anderson, T. Improving the performance of log-structured file systems with adaptive methods. In Proc. of ACM SOSP (1997).

[13] R.H. Patterson, S. Manley, M. Federwisch, D. Hitz, S. Kleiman and S. Owara, SnapMirror: File-System-Based Asynchronous Mirroring for Disaster Recovery, Proc. of FAST 02, the 1th USENIX Conf. on File and Storage Technologies,117-129, 2002.

[14] M. Rosenblum, J.K. Osterhout. The Design and Implementation of a Log-structured File System. ACM Transactions on Computer Systems, Vol.10, No.1 (Feb. 1992), pp. 26-52.

[15] S. Savage and J. Wilkes, AFRAID, A Frequently Redundant Array of Independent Disks. In Proc. Winter USENIX, (San Diego, CA) pages 2739, Jan. 1996. USENIX.

[16] R. Shaull, L. Shrira and X. Hao, Skippy: a New Indexing Method for Long-Lived Snapshots in the Storage Manager, *ACM SIGMOD Conference 2008*,Vancouver, Canada,2008.

[17] J. D. Strunk, G. R. Goodson, M. L. Scheinholtz, C. A. N. Soules and G. R. Ganger. Self-securing storage: protecting data in compromised systems. In Proc. of the 4th OSDI, pages 165180.

[18] A. Tridgell, P. Mackerras. The rsync algorithm. Department of Computer Science Australian National University. TR-CS-96-05

[19] EMC Celerra Replicator http://www.emc.com/.

[20] H. Weatherspoon, L. Ganesh, T. Marian, M. Balakrishnan, and K. Birman Smoke and Mirrors: Reflecting Files at a Geographically Remote Location Without Loss of Performance Proc. of FAST 09, the 7th USENIX Conf. on File and Storage Technologies, 211-224 , 2009.

[21] R. Yan, J. Shu, and D. Chan Wen, An implementation of semi-synchronous remote mirroring system for sans. In ACM Workshop of Grid and Cooperative Computing (GCC) (2004).

[22] M. Lillibridge, K. Eshghi, D. Bhagwat, V. Deolalikar, G. Trezise, and P. Camble, Sparse Indexing: Large Scale, Inline Deduplication Using Sampling and Locality, Proceedings of USENIX File And Storage Systems conference (FAST), 2009.

[23] History of Flight Recorders 1997 - 2005, L-3 Communications Corp., Aviation Recorders http://www.l-3ar.com/html/history.html

[24] IOMeter http://www.iometer.org/

[25] FDR http://en.wikipedia.org/wiki/Flight_data_recorder/

[26] TPC-C http://www.tpc.org/

[27] Anthony M. Townsend Mitchell L. Moss telecommunications infrastructure in disasters: Preparing Cities for Crisis Communications http://www.nyu.edu/ccpr/