УДК 004.22:004.421

МЕТОД ВЫЯВЛЕНИЯ ИЗМЕНЕННЫХ ФАЙЛОВ В ОБЛАЧНОЙ СИСТЕМЕ РЕЗЕРВНОГО КОПИРОВАНИЯ

Б.Г. Атаян

Национальный политехнический университет Армении

Представлен метод облачного резервного копирования большого количества данных, который позволяет оптимизировать фазу обнаружения изменений данных в процессе ступенчатого резервного копирования. Метод оптимизации процесса резервного копирования большого количества данных, в частности веб-сайтов, основан на извлечении и обработке зависимостей программного кода. Процесс резервного копирования организовывается на основе двух основных подходов: полного резервного копирования и ступенчатого резервного копирования. При полном резервном копировании затрачивается довольно много времени и ресурсов для создания резервной копии и ее загрузки на облачный резервный сервер, а также ее сохранения там. Для устранения этого основного недостатка в рамках данной работы предлагается применение ступенчатого резервного копирования, достоинством которого является быстрое создание резервных копий, так как во время ступенчатого резервного копирования в созданную копию включаются только те файлы, которые передаются на момент создания резервной копии. В процессе резервного копирования данных возникает необходимость проверить изменения всех файлов, например, произвести расчет проверочного значения содержимого файла, после чего, сравнив полученные значения с проверочными значениями данных, принять решение об осуществлении резервного копирования. Для выявления измененных файлов необходимо организовать цикл, который будет проверять изменения каждого файла в системе.

С учетом вышеизложенного предложены метод получения графа зависимостей программного кода и алгоритм анализа этого графа, что приводит к списку файлов, которые необходимо проверить на изменения во время операции инкрементального резервного копирования.

Ключевые слова: облачное резервное копирование, программный код, обнаружение изменений, ориентированный граф зависимостей.

Введение. В мире цифровой информации резервное копирование, или резервирование – это процесс копирования и сохранения данных, в результате чего полученная резервная копия может быть использована впоследствии для восстановления первоначальных оригинальных данных в случае их потери или повреждения. Резервное копирование данных может применяться в двух случаях. В первом случае его предназначением является восстановление данных после их потери, будь то их удаление или повреждение. Потеря данных очень

распространена среди пользователей. Во втором случае предназначением резервного копирования является восстановление более ранней информации, то есть восстановление более старой версии данных, причем тот период времени или то количество данных, для которых хранится история резервного копирования, настраивается пользователем системы резервирования.

Значительная часть катастроф происходит не только в результате воздействия человеческого фактора, срыва программного обеспечения и оборудования, но также и по причине чрезвычайных ситуаций [1]. В представленной ниже таблице показаны основные причины потери данных в течение последних 5 лет.

Таблица

Основные причины потери данных в течение последних 5 лет

Причина	**Процентное соотношение, %**
Обновление системы	72
Выключение/сбой электроэнергии	70
Конфигурационные изменения	69
Кибератаки	64
Недобросовестные работники	63
Утечка/потеря данных	63
Наводнение	48
Ураган	47
Землетрясение	46
Торнадо	46
Террористический акт	45
Цунами	44
Извержение вулкана	42
Война	42
Прочее	1

Несмотря на то, что резервное копирование является простейшим способом восстановления от катастроф и должно быть частью плана по восстановлению, резервирование само по себе не должно считаться целостным планом для восстановления от катастрофы, так как не все системы резервного копирования могут в полной мере восстановить и привести в порядок такие сложные комплексные конфигурации, каковыми являются кластерные системы (active directory Server) или многочисленные серверы хранения данных.

Типы резервных копий данных. Процесс резервного копирования организовывается на основе двух основных подходов [2, 3]:

- полное резервное копирование;
- ступенчатое резервное копирование.

Полное резервное копирование, как вытекает из названия, - это создание резервной копии резервируемых данных в полном объеме, которая содержит в себе все файлы, имеющиеся в системе, включая базы данных. Важнейшим недостатком полного резервного копирования является ресурсная затратность данного процесса. Затрачивается довольно много времени и ресурсов для создания резервной копии, ее загрузки на облачный резервный сервер, а также ее сохранения там. Для устранения данного основного недостатка в рамках данной работы предлагается применение ступенчатого резервного копирования, достоинством которого является быстрое создание резервных копий, так как во время ступенчатого резервного копирования в созданную копию включаются только те файлы, которые передаются на момент создания резервной копии. На рис. 1 показан пример расписания резервного копирования.

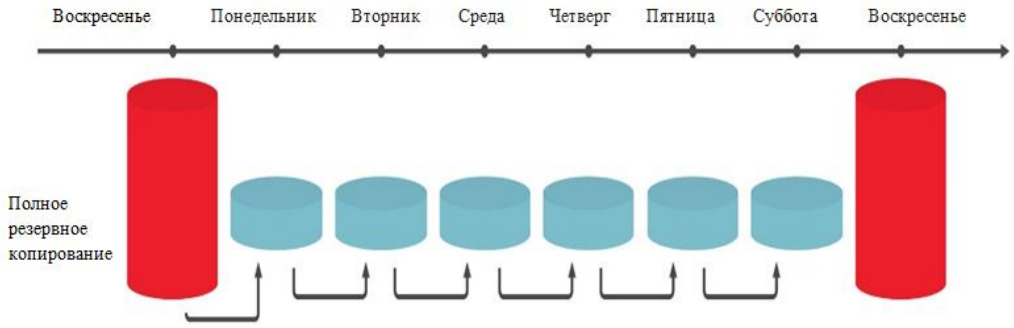

Рис. 1. Схема расписания ступенчатого резервирования данных

Ступенчатое резервное копирование обеспечивает более быстрое резервирование [4], чем несколько раз повторяющееся полное резервное копирование. Но этот метод также имеет недостаток, а именно - данные медленно восстанавливаются из резервной копии. Описанные методы могут применяться также совместно, например, как показано на рис. 1. Предположим, необходимо восстановить резервную копию за четверг. Сначала необходимо восстановить полную резервную копию за понедельник, далее ступенчатые резервные копии за вторник и четверг. Может случиться так, что одна из ступенчатых резервных копий будет недоступна на момент восстановления по определенным причинам (например, сервер, хранящий резервную копию, может быть недоступен по причине технического сбоя). В данном случае процесс полного восстановления не будет осуществлен. Можно заметить, что продолжительность процесса восстановления резервной копии зависит от

количества действий полного и ступенчатого резервного копирования. Например, если полное резервирование осуществляется один раз за пять дней, то в процессе восстановления можно использовать созданную за последние пять дней полную резервную копию. Очевидно, что в системе полного резервного копирования временный интервал осуществления действий полного и ступенчатого резервирования играет большую роль в оптимизации данного процесса.

Особенности резервного копирования веб-сайтов. В случае резервного копирования большого количества и большого объема данных использование ресурсов становится более значительным. В процессе резервного копирования данных возникает необходимость проверить изменения всех файлов, например, произвести расчет проверочного значения содержимого файла, после чего, сравнив полученные значения с проверочными значениями данных, резервированных на сервере, принять решение об осуществлении резервного копирования. Возвращаясь к резервному копированию большого количества данных, однозначно можно отметить, что одним из самых длительных действий данного процесса является выявление измененных файлов. Для выявления измененных файлов необходимо организовать цикл, который будет проверять изменения каждого файла в системе. Ярким примером большого количества данных является одна из наиболее известных систем создания веб-сайтов – WordPress, которая содержит более 1542 файлов. В данном случае для выявления измененных файлов необходимо проверить и сравнить 1542 файла, что может занять достаточно много времени.

Другим процессом, требующим много времени, является расчет проверочного значения содержимого файла (например, хеш-значение). В зависимости от применяемого алгоритма и размеров файла данное время может сокращаться или увеличиваться. Для расчета проверочного значения можно применять алгоритм быстрого хеширования, каковым является, например, CRC32 [5], но подобные алгоритмы имеют недостаток – высокая вероятность коллизии (получение одинакового хеш-значения для различных файлов). Таким образом, на практике более целесообразно применять алгоритмы с низкой вероятностью коллизии.

На сегодняшний день веб-сайты представляют собой большой объем хранящейся в интернете и обрабатываемой информации:

- количество веб-сайтов, имеющихся в интернете, составляло 1.734.290.608 [6] по данным на декабрь 2017 года;
- язык PHP используется в 83,1% всех сайтов [7], в серверной части которых известна примененная технология;

- каждый день приблизительно 70.000 [8] сайтов подвергаются хакерским атакам или бывают недоступны по другим причинам.

Веб-сайты создаются с различными целями, начиная с самых простых и ясных и кончая онлайн-банком или электронной торговлей. В наши дни сложно представить бизнес, который не имел бы веб-сайта. Сайт может быть использован для роста бизнеса, а также для осуществления различных маркетинговых стратегий.

Резервное копирование сайта — это довольно актуальная задача, которую с учетом особенностей области применения можно решить с помощью средств, предоставляющих облачные услуги. Независимо от применяемой технологии и метода создания, основным видом содержания, свойственным для веб-сайтов, является программный код, который имеет следующие особенности [9]:

- каждый файл на сайте может иметь зависимость от других кодовых файлов;
- изменение одного файла может привести к изменению другого файла/ов;
- выявление измененных файлов может быть организовано посредством выявления и анализа связей между файлами.

На рис. 2 описывается, какая связь может быть между различными частями программного кода. В данном случае изменение одной рабочей части программного кода может привести к последовательным изменениям в других частях кода сайта. Эту особенность можно эффективно использовать для оптимизации процесса облачного резервного копирования веб-сайта, содержащего большое количество файлов.

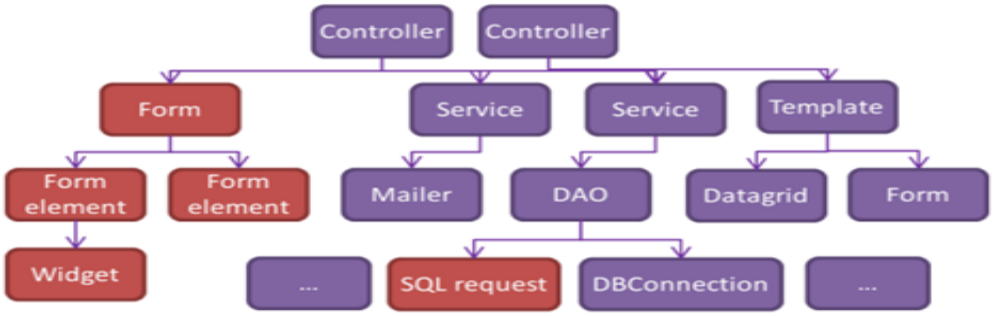

Рис. 2. Пример зависимости файлов

Предлагаемый алгоритм для выявления измененных файлов веб-сайта. Принимая во внимание вышеописанное, а также тот факт, что основной задачей резервного копирования большого количества данных является затратность

действий, связанных с выявлением измененных файлов, последнее можно организовать посредством обнаружения связей между файлами, принадлежащими резервируемому сайту, а также анализа графа, характеризующего данные связи. Предлагаемый метод резервного копирования веб-сайта заключается в следующем.

Если общее количество файлов, имеющихся в системе, обозначим через $\boldsymbol{N}$, а количество измененных файлов за i-й отрезок времени резервного копирования - $\boldsymbol{K_i}$, где $\boldsymbol{K_i \leq N}$, то количество тех файлов, изменения которых необходимо будет проверить, составит $\boldsymbol{M}$, и так как все эти файлы нуждаются в проверке, то $\boldsymbol{M = N}$.

Время действия проверки $\boldsymbol{F_j}$ – го файла можно рассчитать по формуле

$$\boldsymbol{c_j = C(F_j)}, \text{ где } \boldsymbol{F_j, j \in N},$$

где $\boldsymbol{C}$ – функция зависимости значения времени действия проверки от файла. Во время $\boldsymbol{i}$-го резервирования время проверки всех измененных файлов рассчитывается по формуле

$$\boldsymbol{c_i} = \sum_{\boldsymbol{j}}^{\boldsymbol{M}} \boldsymbol{c_j}.$$

Задача заключается в минимизации значения $\boldsymbol{M}$ так, чтобы $\boldsymbol{M \leq K_i}$, в результате чего снизится значение $\boldsymbol{c_i}$. Необходимую информацию для выявления файлов, нуждающихся в проверке изменений, можно получить посредством анализа графа зависимостей. Вектор, содержащий наименования файлов, нуждающихся в проверке изменений, обозначим $\boldsymbol{L_{check}}$.

Предлагаемый алгоритм анализа включает следующую последовательность шагов:

Шаг 1. Создать ориентированный граф зависимостей файлов $G(U, V)$, где вершина $v \in V$ соответствует файлу, а ребро $u \in U$ - зависимости между двумя файлами. Каждое ребро имеет свой вес, который показывает уровень зависимости файлов, соответствующих вершинам, соединенным с данным ребром. Вес определяется по следующей формуле:

$$\boldsymbol{D_{v_i v_j} = \frac{N_{F_i}}{N_{F_{ij}}}},$$

где $\boldsymbol{N_{F_i}}$- количество всех элементов (class, static, global, function), которые имеются в файле $\boldsymbol{F_i}$, а $\boldsymbol{N_{F_{ij}}}$ - количество из данных элементов, которые имеются в файле $\boldsymbol{F_j}$.

Шаг 2. Получение файлов, требующих изменений, из ориентированного графа зависимостей файлов $G(U, V)$.

Шаг 3. Выполнение проверки изменений для полученных файлов. Проверка осуществляется для всех файлов, имеющихся на векторе $\boldsymbol{L_{check}}$, полученных в результате анализа, после чего все файлы, нуждающиеся в резервном копировании, резервируются.

Алгоритм получения всех файлов, требующих изменений, из ориентированного графа зависимостей файлов $\boldsymbol{G(U, V)}$ следующий (Шаг 2):

1. Найти ту вершину, которая имеет наибольшее количество выходящих из нее ребер. Если таких вершин несколько, то выбирается та вершина, сумма весов ребер которой минимальна.
2. Проверить, действительно ли файл, соответствующий выбранной вершине, был изменен. Если да, то продолжить, в противном случае - удалить данную вершину из графа и вернуться к пункту 1.
3. Для найденной вершины рассчитать среднее значение весов всех ребер, выходящих из нее. Если ребер нет, удалить данную вершину из графа и вернуться к пункту 1.
4. Выбрать те вершины, весы ребер которых больше рассчитанного значения и количество этих ребер ≤ 1. Добавить наименования файлов, соответствующих данным вершинам, к вектору $\boldsymbol{L_{check}}$.
5. Те из вершин, которые соединены между собой ребрами, имеющими минимальный вес и которые более не имеют связей, удалить из графа, а для остальных вершин вернуться к пункту 1.

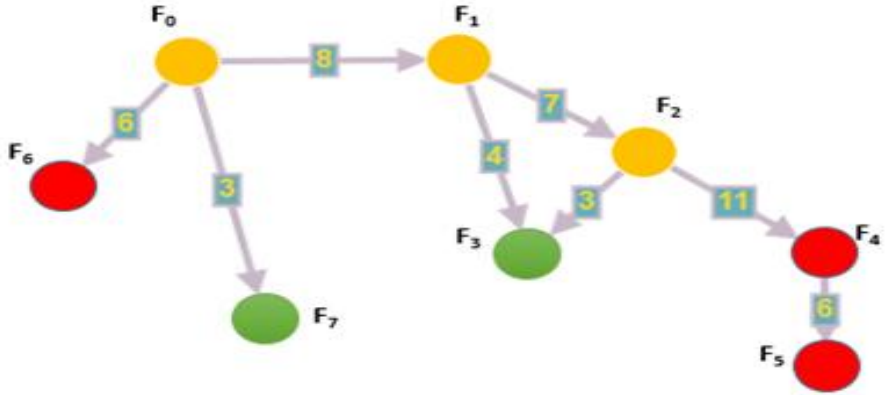

Рис. 3. Пример графа зависимостей файлов

Опишем пример работы алгоритма, принимая за основу представленный на рис. 3 граф зависимостей. Предположим, что из вершин, имеющих максимальное количество ребер, были изменены вершины F0 и F2. Вершина, которая имеет наибольшее количество выходящих из нее ребер, — это F0, которая в данном примере соответствует измененному файлу. Далее для данной

вершины рассчитываем среднее значение весов всех ребер, выходящих из нее, которое составляет 5,6. Вершинами, весы ребер которых больше рассчитанного значения, являются F6 и F1. Количество ребер, выходящих из вершины F6, будет $0 \leq 1$, следовательно, данная вершина прибавляется к вектору $\boldsymbol{L_{check}}$. Вершина F7 была соединена с ребром, имеющим минимальный вес, следовательно, она игнорируется, не нуждается в проверке. Следующими вершинами, имеющими наибольшее количество выходящих ребер, являются вершины F1 и F2. В данном примере предполагалось, что вершина F2 была изменена, следовательно, вершина F1 нуждается в проверке и прибавляется к вектору $\boldsymbol{L_{check}}$. Произведя аналогичные действия и расчёты для вершины F2, получим следующий вектор $\boldsymbol{L_{check}}$:

$$\boldsymbol{L_{check}} = \{\mathbf{F_6}, \mathbf{F_1}, \mathbf{F_4}, \mathbf{F_5}\}.$$

В результате в описанном маленьком примере удалось сэкономить действия проверки изменений двух файлов, что составляет 25% от общих файлов.

Заключение. Таким образом, в процессе ступенчатого резервного копирования веб-сайтов на этапе выявления изменений рекомендуется проводить анализ ориентированного графа зависимостей файлов, в результате которого получается список файлов, нуждающихся в проверке во время действия резервного копирования. Далее необходимо осуществить проверку файлов и создать единый резервный файл. Резервная копия должна содержать в себе не только резервируемые файлы, но также и определенные мета-данные о них. Применение единых файлов резервных копий наиболее актуально во время облачного резервного копирования, поскольку в этом случае данные собираются в одном файле, т.е. становится возможным осуществить более легкое управление, сохранение и передачу резервных копий в облачных серверах.

Литература

1. **Symantec Corporation**, Symantec Disaster Recovery Study, Global results. http://www.symantec.com/content/en/us/about/media/pdfs/Symc_Survey_SAMGDisasterRecovery_Global_2010.pdf, 2010.
2. Performance and Availability Modeling of ITSystems with Data Backup and Restore / **R. Xia, X. Yin, Lopez J. Alonso, et al** // IEEE Transactions on Dependable and Secure Computing.- 2014.- P. 375-389.
3. **Chervenak A., Vellanki V., and Kurmas Z.** Protecting File Systems: A Survey of Backup Techniques // Joint NASA and IEEE Mass Storage Conf.- 1998. - P. 17-32.

4. Availability Modeling and Analysis for Data Backup and Restore Operations / **X. Yin, J. Alonso, F. Machida, et al** // IEEE 31st Symposium on Reliable Distributed Systems (SRDS).- 2012.- P.141-150.
5. **Peterson W., Brown T.** Cyclic Codes for Error Detection // Proceedings of the IRE.-1961. - P. 228-235.
6. **InternetLiveStats**, http://www.internetlivestats.com/total-number- of-websites/
7. **Netcraft Web Server Survey,** https://news.netcraft.com/archives/2013/01/31/php-just-grows-grows.html
8. **InternetLiveStats**, http://www.internetlivestats.com/watch/websites-hacked/
9. **Egyed A.** A Scenario-Driven Approach to Trace Dependency Analysis // IEEE transactions on software engineering.- 2003. - P.116-132.

Поступила в редакцию 11.02.2018.
Принята к опубликованию 05.06.2018.

ԱՄՊԱՅԻՆ ՊԱՀՈՒՍՏԱՎՈՐՄԱՆ ՀԱՄԱԿԱՐԳՈՒՄ ՓՈՓՈԽՎԱԾ ՏՎՅԱԼՆԵՐԻ ՀԱՅՏՆԱԲԵՐՄԱՆ ՄԵԹՈԴ

Բ.Գ. Աթայան

Ներկայացվում է մեծաքանակ տվյալների ամպային պահուստավորման մեթոդ, որը թույլ է տալիս աստիճանական պահուստավորման գործընթացի տվյալների փոփոխությունների հայտնաբերման փուլում հասնել աշխատանքի լավարկման: Աստիճանական պահուստավորման գործընթացի լավարկման մեթոդը հիմնված է մեծաքանակ տվյալների, մասնավորապես՝ վեբ կայքերի, ծրագրային կոդում կախվածությունների ստացման և մշակման վրա: Ամպային պահուստավորման գործընթացը կարող է կազմակերպվել երկու եղանակով՝ ամբողջական պահուստավորում և աստիճանական պահուստավորում: Ամբողջական պահուստավորման դեպքում ծախսվում են մեծ քանակությամբ հաշվողական ռեսուրսներ՝ պահուստային օրինակի ստեղծման և պահուստային սերվեր վերբեռնման համար: Այս թերությունը նկարագրված համակարգում լուծվում է աստիճանական պահուստավորման միջոցով, որի առավելություններից է պահուստների արագ ստեղծումը, քանի որ պահուստավորվում են միայն փոփոխված տվյալները: Արդյունքում՝ առաջանում է պահուստավորման գործողության ժամանակ փոփոխված ֆայլերի հայտնաբերման խնդիրը: Փոփոխված ֆայլերի հայտնաբերումը կատարվում է ֆայլերի պարունակության ստուգիչ արժեքների ստուգման միջոցով, այսինքն, համակարգում առկա բոլոր ֆայլերի համար կատարվում է ստուգիչ արժեքների հաշվարկ, որից հետո, պահուստավորման գործողության առաջին փուլում բոլոր ֆայլերի համար կատարվում է ստուգիչ արժեքի վերահաշվարկ և համեմատություն նախորդ արժեքների հետ: Ակնհայտ է, որ եթե համակարգում առկա ֆայլերի քանակը մեծ է, ապա այս

գործողությունը կարող է լինել բավականին ռեսուրսատար: Առաջարկվում է այս խնդիրը լուծել փոփոխված ֆայլերի հայտնաբերման ալգորիթմի միջոցով, որը հիմնված է համակարգում առկա տվյալների կախվածությունների վրա:

Առաջարկվում է ֆայլերի կախվածությունների ուղղորդված գրաֆի ստացման մեթոդ և վերլուծության ալգորիթմ, և արդյունքում ստացվում է տվյալ պահուստավորման գործողության ժամանակ փոփոխությունների ստուգման կարիք ունեցող ֆայլերի ցուցակ:

Առանցքային բառեր. ամպային պահուստավորում, ծրագրային կոդ, փոփոխությունների հայտնաբերում, կախվածությունների ուղղորդված գրաֆ:

A METHOD FOR DETECTING THE CHANGED FILES IN A CLOUD-BASED BACKUP SYSTEM

B.G. Atayan

The cloud backup method for large amounts of data is presented which optimizes the change detection phase of incremental backup process. The incremental backup process optimization method is based on the dependency detection and analysis in large amounts of data, particularly, in the source code of websites. The cloud backup process can be organized with two different approaches. The first one is the full backup. The resource consumption, while performing full backup can be significant, particularly, when the amount of data in the system is large. Also, large full backups should be uploaded to remote backup servers, which is also resource-consuming. To overcome these disadvantages, the incremental backup approach is used in the Cloud Backup System. The advantage of incremental backup is that the only data that have been changed after the last backup are backed up, which means that the size of backup archive is smaller and it will take fewer resources to create and upload it to the destination backup server. As a result, the problem of detection of changed files arises in the first phase of incremental backup process. The change detection is usually done by creating check values of data in the system and comparing them to the newly computed values while processing the backup. It is obvious, that if the number of files in the system is relatively large, then, the process of change detection will take significant amount of resources. The method of solving this issue is presented, which is based on dependencies between the files that are being backed up in the system.

The oriented dependency graph construction and the analysis algorithm are presented, the output of which is the list of files from the data that requires change detection.

Keywords: cloud backup, source code, change detection, oriented dependency graph.