Методика обеспечения своевременности и полноты обмена информационными ресурсами в корпоративных сетях с распределенным реестром

И.Б. Саенко[1], И.Н. Фабияновский[1], А.М. Старков[1], Е.Г. Баленко[2]

[1]Военная академия связи имени Маршала Советского Союза С.М. Буденного, Санкт-Петербург

[2]Донской государственный аграрный университет, Персиановский

Аннотация: В статье рассматривается методика обеспечения своевременности и полноты обмена информационными ресурсами в корпоративных сетях, построенных на основе технологии распределенного реестра, которая учитывает вариативность стратегии поведения системы распределенного реестра при информационном обмене. Методика учитывает нештатные функции, такие, как формирование ответвления обрабатываемых данных и воздействия злоумышленника, а также позволяет определить среднее значение времени задержки генерации блока путем корректировки числа операций, необходимых для решения блока. Применение данной методики позволяет повысить значение показателя своевременности и полноты обмена информационными ресурсами в корпоративной сети на 30% по сравнению с существующей системой информационного обмена.

Ключевые слова: технология распределенного реестра, корпоративная сеть, информационные ресурсы.

Система с технологией распределенного реестра (ТРР) – система баз данных, распределенных между сетевыми узлами, не контролируемая единым органом [1].

Основными качественными свойствами, определяющими эффективность такой системы при применении ее в корпоративных сетях (КС) для обеспечения безопасного обмена информационными ресурсами (ИР), являются полнота и своевременность обмена ИР, которые зависят от множества характеристик такой системы, например, возможность изменения системы распределенного реестра (СРР), среднего времени задержки генерации блока, возможности формирования ответвления обрабатываемых данных, изменений системы из-за воздействия противника и др. [2-4].

С целью обеспечения своевременности и полноты обмена ИР в КС с учетом изменяющихся характеристик СРР разработана соответствующая методика (рис. 1), которая включает в себя 5 этапов.

На 1 этапе происходит сбор информации. В зависимости от условий функционирования КС времена создания блоков будут различными.

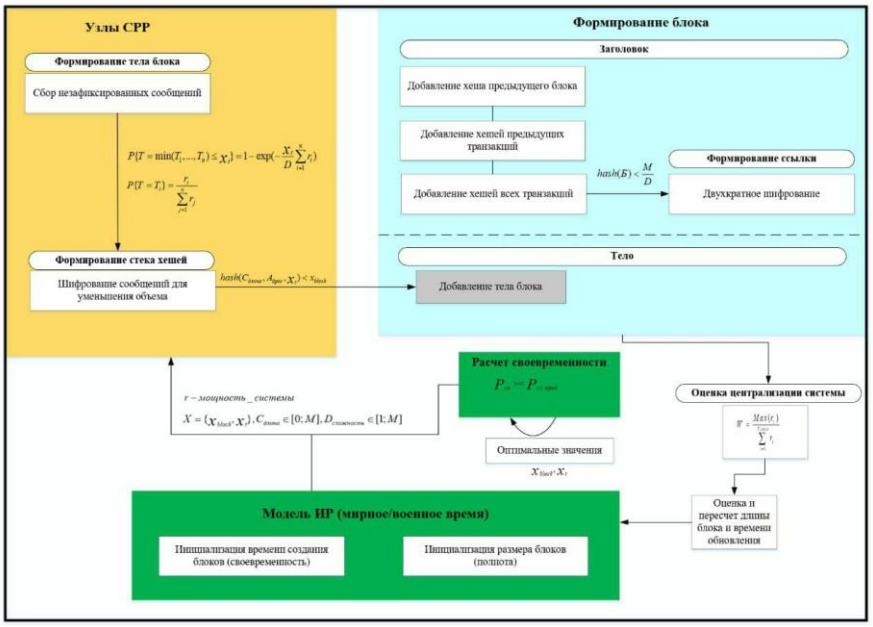

Рис. 1 – Структура методики

На 2 этапе производится сбор незафиксированных сообщений. Время $T(r)$, которое необходимо для выполнения r операций, связанных с вычислением блока, имеет экспоненциальное распределение с параметром r/D:

$$P\{T = \min(T_1,...,T_n) \le t\} = 1 - \exp(-\frac{t}{D}\sum_{i=1}^{n} r_i) \ .$$

Рассмотрим n узлов с хеширующими мощностями r_1, r_2, …, r_n. Время T для нахождения блока равно минимуму из случайных величин $T(r_i)$ и, согласно свойствам экспоненциального распределения, также распределено экспоненциально:

$$P\{T = T_i\} = \frac{r_i}{\sum_{j=1}^{n} r_j}$$

На 3 этапе производится шифрование сообщений для уменьшения объема. Чтобы блок считался действительным, его ссылка не должна превышать размера блока:

$$hash(C_{\text{длина}}, A_{\text{адрес}}, t) < Б_{\text{размер}}$$

На 4 этапе производится формирование блока. Для этого в начале процесса осуществляется перерасчет длины и времени обновления цепочки блоков (ЦБ) СРР.

Каждый блок состоит из заголовка блока с ключевыми параметрами и списка транзакций. Для того чтобы было возможно ссылаться на конкретный блок, его заголовок хешируется дважды с помощью функции *SHA*-256. Блок считается действительным, если его ссылка превышает определенный порог:

$$hash(Б) < \frac{M}{D}$$

Чем больше величина D, тем больше итераций необходимо произвести для нахождения действующего блока, причем ожидаемое число операций равняется D.

На 5 этапе производится оценка централизации системы, которая показывает отношение максимального значения ресурсной мощности r_z, сосредоточенной в одном узле КС, к общему значению ресурсной мощности всей сети.

Для оценки полноты обмена ИР между конечными устройствами СРР применяется подход, основанный на кластеризации сообщений путем их передачи между точками данных [5]. Предложенный алгоритм принимает на вход матрицу схожести между элементами набора данных и возвращает набор меток, присвоенных этим элементам. Обмен сообщениями между точками данных происходит до тех пор, пока не будет получен набор образцов высокого качества.

Определим метрику подобия $s(x_i, x_j) > s(x_i, x_k)$, если наблюдение x_i больше похоже на наблюдение x_j и меньше похоже на наблюдение x_k. Простым примером такой метрики подобия является: $s(x_i, x_j) = -\| x_i - x_j \|^2$. Сходство определяется как отрицательное значение евклидова расстояния между двумя экземплярами. Чем больше расстояние между любыми двумя экземплярами, тем меньше сходство между ними.

В итоге получаем матрицу подобия, представленную на рис. 2.

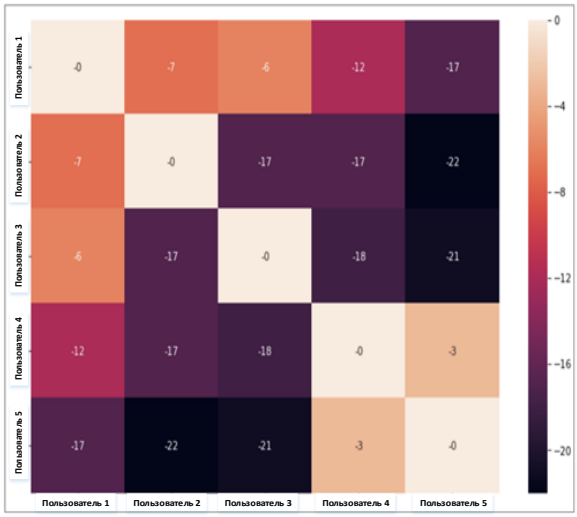

Рис. 2 – Матрица подобия

Значения недиагональных элементов будут определять количество сформированных кластеров. Чем меньше это значение ($value \leq 0$), тем меньше количество кластеров.

Опишем «соответствие», составив две нулевые матрицы. Одна из них, $r_{i,k}$, которая определяет, насколько наблюдение k_i является «образцом для подражания» для *i*-го наблюдения по отношению ко всем другим возможным «образцам для подражания». Другая матрица, $a_{i,k}$, определяет, насколько уместно для *i*-го наблюдения использовать k_i наблюдение в качестве «эталонной модели».

Матрицы обновляются последовательно по следующим правилам:

$$r_{i,k} \leftarrow s(x_i, x_k) - \max_{k' \neq k}\{a_{i,k'} + s(x_i, x_{k'})\},$$

$$a_{i,k} \leftarrow \min\left(0, r_{k,k} + \sum_{i' \notin \{i,k\}} \max(0, r_{i',k})\right), i \neq k,$$

$$a_{k,k} \leftarrow \sum_{i' \neq k} \max(0, r_{i',k}).$$

Так как своевременность, наравне с полнотой, является критически важным параметром, для её достижения предлагается алгоритм, псевдокод которого представлен ниже.

```
MAIN(D, eps, MinPts) {
        C = 0
  for each point P in dataset D {
    if P is visited
      continue next point
    mark P as visited
    NeighborPts = regionQuery(P, eps)
    if sizeof(NeighborPts) < MinPts
      mark P as NOISE
    else {
      C = next cluster
      expandCluster(P, NeighborPts, C, eps, MinPts)
    }
  }
}
expandCluster(P, NeighborPts, C, eps, MinPts) {
  add P to cluster C
  for each point Q in NeighborPts {
    if Q is not visited {
      mark Q as visited
      QNeighborPts = regionQuery(Q, eps)
      if sizeof(QNeighborPts) >= MinPts
        NeighborPts = NeighborPts joined with QNeighborPts
    }
    if Q is not yet member of any cluster
      add Q to cluster C
```

```
    }
  }
  regionQuery(P, eps)
    return all points within P eps-neighborhood (including P)
```

Введем несколько определений. Пусть задана некоторая симметричная функция расстояния $p(x,y)$ и константы ε и m. Обозначим $E(x)$ – область, для которой $\forall y: p(x,y) \le \varepsilon$, ε – окрестность объекта x. Корневым объектом степени m называется объект, ε – в окрестности которого содержит не менее m объектов: $|E(x)| \ge m$. Объект p непосредственно плотно достижим из объекта q, если $p \in E(q)$ и q – корневой объект. Объект p плотно достижим из объекта q, если $\exists p_1, p_2, ..., p_n$, $p_1 = q$, $p_n = p$, такие что $\forall i \in 1...n-1: p_{i+1}$ непосредственно плотно достижим из p_i.

Выберем корневой объект из набора данных, пометим его и поместим всех его соседей в список обхода. Теперь для каждой точки из списка, если она тоже корневая, добавим всех ее соседей в список обхода. Кластеры помеченных точек, сформированные в ходе этого алгоритма, максимальны. Их нельзя расширить еще одной точкой, чтобы удовлетворялись все условия. Отсюда следует, что если обошли не все точки, можно перезапустить обход из другого корневого объекта. Тогда новый кластер не поглотит предыдущий.

Сложность данного алгоритма может равняться $O(N)$, что полностью удовлетворяет условию своевременности. Также стоит отметить, что для оптимальной работы этого алгоритма необходим выбор оптимального количества кластеров (рис. 3, 4) [6]:

$$J(C) = \sum_{k=1}^{K} \sum_{i \in C_k} \| x_i - \mu_k \| \to \min_C ,$$

где C – набор кластеров мощностью K, μ_k - центр кластера C_k.

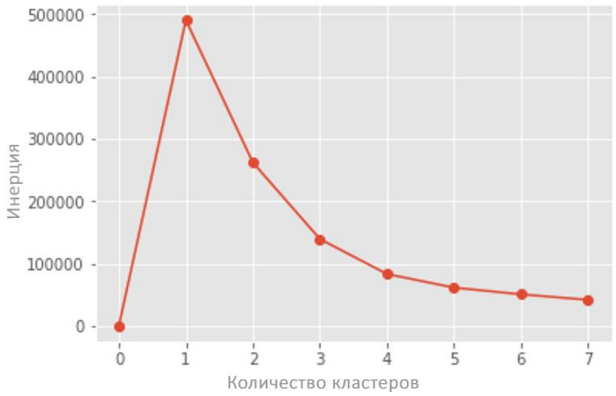

Рис. 3 – Зависимость суммы квадратов внутри кластеров от их количества

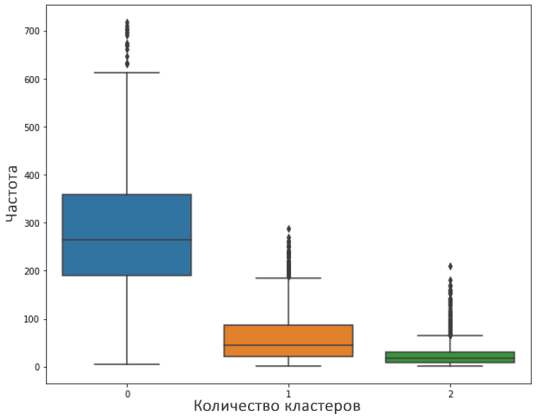

Рис. 4 – Зависимость количества кластеров от частоты

Что касается обеспечения полноты обмена, то следует отметить, что данный подход отлично справляется с плотными, хорошо отделенными друг от друга кластерами, как это показано на рис. 5.

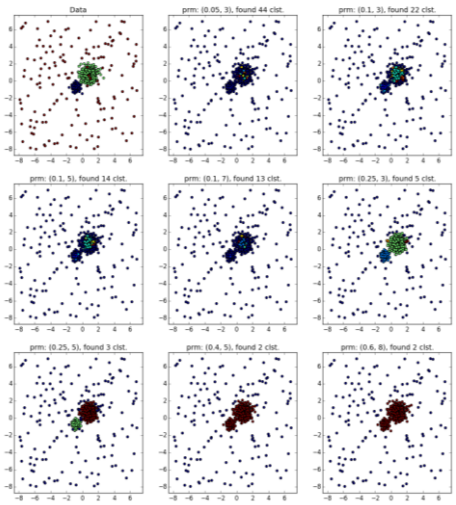

Рис. 5 – Распределение конечных устройств СРР

В качестве метрик оценки точности предложенного подхода предлагается использовать следующие показатели.

1. Кластерный индекс достоверности. Эта метрика позволяет оценить качество кластеризации, используя только начальные данные и результат кластеризации, следующим образом:

$$s = \frac{p - q}{\max(p, q)} \ ,$$

где p –среднее расстояние до точек в ближайшем кластере, q – среднее внутрикластерное расстояние до всех точек в собственном кластере. Значения этой оценки для каждого кластера лежат в диапазоне от –1 до 1. Если эта оценка ближе к 1, то это указывает на то, что точка данных очень похожа на другие точки данных в кластере. Если оценка ближе к –1, то это указывает на то, что точка данных не похожа на остальные точки данных в его кластере.

2. Однородность и полнота. Данные метрики базируются на функциях энтропии и условной энтропии:

$$h = 1 - \frac{H(C \mid K)}{H(C)} \ ,$$

$$c = 1 - \frac{H(K \mid C)}{H(K)} \ ,$$

где K – результат кластеризации, а C – начальное разделение. Метрики не симметричны. Оценки лежат в диапазоне [0,1]. Значения ближе к 1 указывают на более точные результаты кластеризации.

На рис. 6 представлена зависимость коэффициента централизации СРР от времени отправки ЦБ СРР [7].

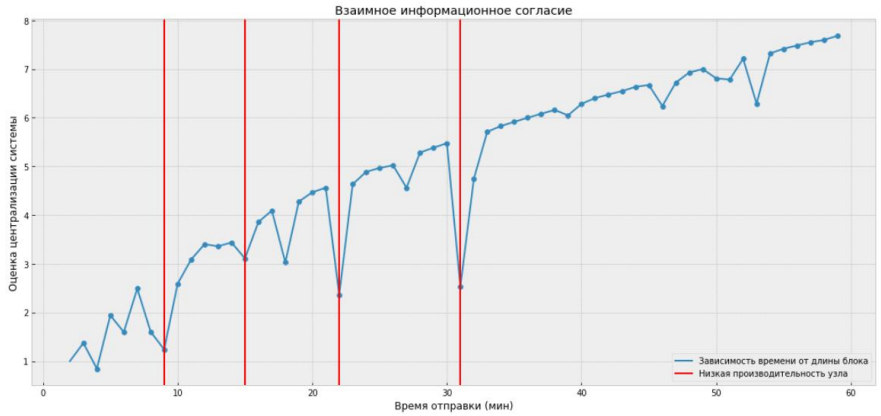

Рис. 6 – Зависимость коэффициента централизации СРР от времени отправки ЦБ СРР

Прокомментируем характер изменения оценки степени централизации системы. Легко заметить, что на отдельных участках графика идет падение эффективности системы. Причиной этого является тот факт, что у каждого узла КС имеется разная вычислительная мощность, которая измеряется скоростью создания хешей. На отдельных итерациях слабые машины не справляются с просчетом больших хешей, система реконфигурируется [8-10]. Однако в целом можно отметить, что с увеличением длины блока растет время отправки.

Зависимость размера блока в СРР от времени отправки ЦБ СРР представлена на рис. 7.

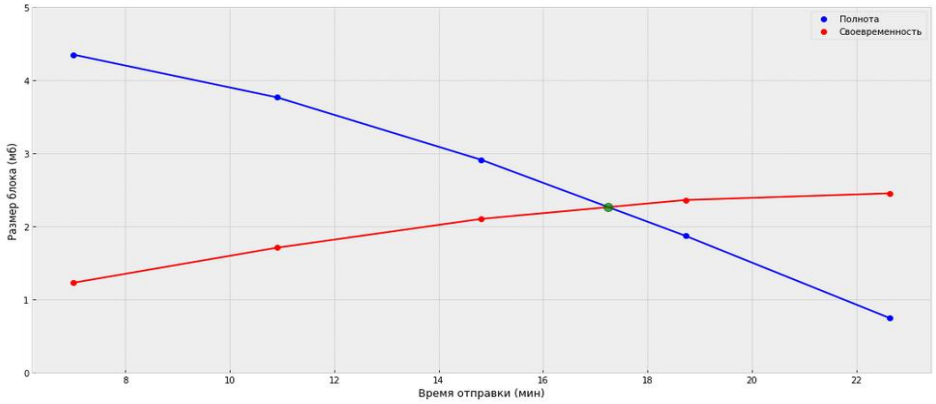

Рис. 7 – Зависимость размера блока в СРР от времени отправки ЦБ СРР

Анализ показал, что для выполнения требований по своевременности и полноте обмена ИР СРР ЕИП время отправки блоков должно быть 17,5 минут, а размер блока – 2,2 Мб.

Таким образом, разработанная методика показывает, что показатели своевременности и полноты обмена ИР в КС при использовании ТРР повышаются на 30% по сравнению с показателями в системе обмена без ТРР.

Литература

1. Brendan J. Frey, Delbert Dueck Clustering by Passing Messages between Data Points. URL: warwick.ac.uk/fac/sci/dcs/research/combi/seminars/freydueck_affinitypropagation_science2007.pdf.

2. Саенко И.Б., Фабияновский И.Н., Николаев В.В., Ясинский С.А. Построение модели функционирования распределенной информационной системы на основе блокчейн-технологии // Информация и космос. 2020. №4. с. 73-78.

3. Козленко А.В., Авраменко В.С., Саенко И.Б., Кий А.В. Метод оценки уровня защиты информации от НСД в компьютерных сетях на основе графа защищенности // Труды СПИИРАН. 2012. №2 (21). С. 41-55.

4. Rabin M. Rondamized byzantine generals // Proc. 24th Symp. on Foundations of Computer Sci. USA. 1983. pp. 393-402.

5. Rabin M. Probabilistic algorithms // Algorithms and Complexity. N. Y.-London: Acad. Press. 1976. pp. 21-39.

6. Kotenko, I.V., Saenko, I.B., Kotsynyak, M.A., Lauta, O.S. Assessment Of Cyber-Resilience Of Computer Networks Based On Simulation Of Cyber Attacks By The Stochastic Networks Conversion Method // SPIIRAS Proceedings. 2017. 6(55). pp. 160–184.

7. Robinson P. Hyland-Wood D., Saltini R., Johnson S., Brainard J. Atomic Crosschain Transactions for Ethereum Private Sidechains.

URL:semanticscholar.org/paper/Atomic-Crosschain-Transactions-for-Ethereum-Private-Robinson-Hyland-Wood/9a0889a3fd116595a697a180c852817de5caaa50.

8. Rosenfeld M. Analysis of bitcoin pooled mining reward systems // arXiv preprint. arXiv: 1112.4980. 2011.

9. Sapirshtein A., Sompolinsky Y., Zohar A. Optimal selfish mining strategies in Bitcoin // arXiv preprint. arXiv: 1507.06183. 2015.

10. Schlichting R., Schneider F. Fault-stop processes: an approach to designing fault tolerant computing systems // ACM Trans. Comput. Syst. 1983. V. 1. № 3. pp. 222-238.

References

1. Brendan J. Frey, Delbert Dueck Clustering by Passing Messages between Data Points. URL: warwick.ac.uk/fac/sci/dcs/research/combi/seminars/freydueck_affinitypropagation_science2007.pdf.

2. Saenko I.B., Fabiyanovskij I.N., Nikolaev V.V., Yasinskij S.A. Informaciya i kosmos. 2020. №4. pp. 73-78.

3. Kozlenko A.V., Avramenko V.S., Saenko I.B., Kij A.V. Metod ocenki urovnja zashhity informacii ot NSD v komp'juternyh setjah na osnove grafa zashhishhennosti [A method for assessing the level of information protection against tampering in computer networks based on the security graph]. Trudy SPIIRAN, 2012, №2(21). pp. 41-55.

4. Rabin M. Rondamized byzantine generals Proc. 24th Symp. on Foundations of Computer Sci. USA. 1983. pp. 393-402.

5. Rabin M. Probabilistic algorithms. Algorithms and Complexity. N. Y.-London: Acad. Press. 1976. pp. 21-39.

6. Kotenko, I.V., Saenko, I.B., Kotsynyak, M.A., Lauta, O.S. Assessment Of Cyber-Resilience Of Computer Networks Based On Simulation Of Cyber Attacks By The Stochastic Networks Conversion Method SPIIRAS Proceedings. 2017. 6(55). pp. 160–184.

7. Robinson P. Hyland-Wood D., Saltini R., Johnson S., Brainard J. Atomic Crosschain Transactions for Ethereum Private Sidechains. URL:semanticscholar.org/paper/Atomic-Crosschain-Transactions-for-Ethereum-Private-Robinson-Hyland-Wood/9a0889a3fd116595a697a180c852817de5caaa50.

8. Rosenfeld M. Analysis of bitcoin pooled mining reward systems. arXiv preprint arXiv: 1112.4980. 2011.

9. Sapirshtein A., Sompolinsky Y. and Zohar A. Optimal selfish mining strategies in Bitcoin. arXiv preprint arXiv: 1507.06183. 2015.

10. Schlichting R., Schneider F. Fault-stop processes: an approach to designing fault tolerant computing systems ACM Trans. Comput. Syst. 1983. V. 1. № 3. pp. 222-238.