Для оценки трудоёмкости планируемого объёма работ применяется метод WBS (WorkBreakDownStructure) – иерархическая декомпозиция всего объёма работ до уровня, на котором задача может быть оценена и спланирована [1]. При проведении иерархического разбиения работ используется группировка задач по логическим доменам, где требуется привлечение разработчиков с однородным уровнем экспертизы и схожих технических навыков с учётом рекомендаций консорциума TMForum.

Формирование и учёт состава команды разработки происходит на основании ресурсного плана проекта, учитывающего не только доступность каждого ресурса (разработчика) с учётом выходных, праздничных дней, отпусков, но, также, и определённые показатели, характеризующие производительность и вовлеченность каждого разработчика:

Вовлеченность – характеристика, определяющая степень привлечения разработчика непосредственно к написанию программного кода. Разработчик может частично привлекаться на проект либо решать задачи в нескольких логических доменах, либо только частично заниматься непосредственно разработкой;

Производительность – коэффициент, определяющий способность разработчика решать задачи относительно оценки, определённой техническим лидером. Определяется на основании опыта разработчика в конкретном домене и в ходе проекта корректируется на основании накопленной статистики решения проектных задач.

Для оценки текущего состояния процесса разработки используется расширение метода освоенного объёма (EVT – EarnedValueTechnique) [1]. В разрезе количества задач и их трудоёмкости с учётом реализованного объёма работ определяется текущий статус процесса разработки в любой момент времени. На основании известных характеристик команды разработки и накопленных статистических данных производится расчёт планируемых дат окончания запланированных работ.

Система контроля и оценки состояния процесса разработки использует адаптацию каскадной модели разработки ПО, совмещая фазы разработки и тестирования. Привлечение команды тестирования для верификации реализованных функций на этапе разработки позволяет обнаружить большую часть дефектов на более ранних этапах. Введение количественного соотношения «входящих» задач (обнаруженные дефекты, улучшения либо необходимые дополнительные изменения) к «исходящим» задачам с учётом основного объёма работ и характеристик команды разработки, позволяет на базе экстраполяции оценить ожидаемую дату завершения работ и потенциальный сдвиг сроков разработки.

Список использованных источников:
1. Руководство к своду знаний по управлению проектами (Руководство PMBOK). – М.: Олимп-Бизнес, 2018 – 590с.
2. TM Forum. [Electronic resource] - https://www.tmforum.org/. Dateofaccess: 21.03.2018

АЛГОРИТМ DESX КАК СРЕДСТВО ПАРАЛЛЕЛЬНОГО ПОТОКОВОГО ШИФРОВАНИЯ БОЛЬШИХ ОБЪЁМОВ ДАННЫХ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Куцейко Р.Ю.

Ярмолик В.Н. – д.т.н., профессор

В настоящее время существует множество отраслей деятельности человека, которые сталкиваются с необходимостью хранения и обработки огромного количество цифровой информации на различных носителях. В случае необходимости защиты такого большого объёма данных с помощью криптографии, наибольшей проблемой становится время, необходимое на процесс шифрования и дешифрования. В связи с тем, что скорость процессорной обработки информации на сегодняшний день растёт медленно, для решения поставленной задачи необходимо использовать параллельные вычисления, обеспечивающие многократное ускорение обработки и алгоритмы, поддающиеся разбиению на параллельные процессы. В качестве такого алгоритма можно рассматривать алгоритм потокового шифрования DESX. Процесс его работы поддаётся разбиению на отдельные параллельные потоки, а относительно простая реализация позволяет производить преобразования на высокой скорости.

DESX – блочный потоковый алгоритм шифрования, разработанный на базе стандарта DES (DataEncryptionStandard). Основное отличие от других модификаций заключается в добавлении процедуры отбеливания ключа шифрования для увеличения криптографической стойкости [1]. В основе алгоритма шифрования лежит преобразование сетью Фейстеля, которое представлено на рисунке 1.

На схеме наглядно изображен один из последовательных раундов преобразования входного блока данных. Блок разбивается на старшую и младшую части, после чего они обрабатываются отдельно. Вычисляется образующая функция F и происходит суммирование по модулю два для определения частей блока, который будут участвовать в следующем раунде шифрования. В зависимости от избранной стратегии шифрования (электронная кодировочная книга, сцепление блоков шифрованного текста, обратная связь по шифрованному тексту и обратная связь по выходу) можно добиться различных результатов шифрования со своими уникальными особенностями [2].

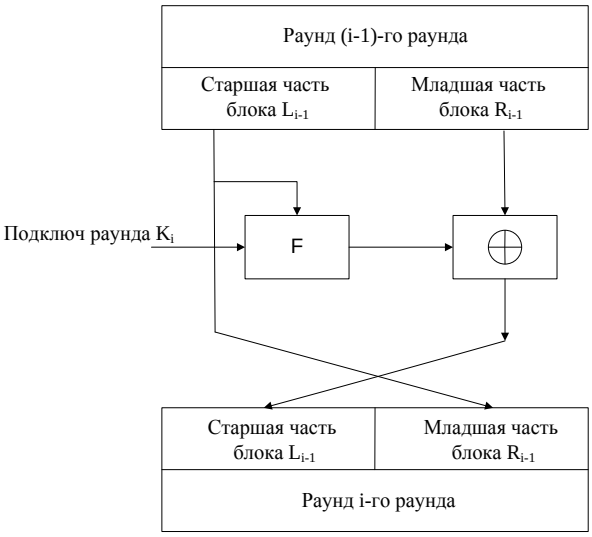

Рис. 1 –Раунд сети Фейстеля

Используя стратегию шифрования электронной кодировочной книги можно получить следующие преимущества:

- высокая скорость шифрования;
- легко поддаётся разбиению на параллельные процессы, что ещё больше повышает скорость шифрования [3];
- обладает приемлемыми показателями криптографической стойкости;

Основным недостатком можно считать более низкую, по сравнению с другими стратегиями шифрования, криптографическую стойкость. Три упомянутые стратегии шифрования (сцепление блоков шифрованного текста, обратная связь по шифрованному тексту и обратная связь по выходу) решают описанную проблему, однако гораздо хуже подаются разбиению на параллельные процессы, вследствие необходимости учитывать на текущем шаге результат предыдущих вычислений. По этой причине данные подходы сильно уступают в скорости работы электронной кодовой книге. Одной из возможностей решения описанной проблемы является разбиение данных на большие блоки, которые независимо будут шифроваться на основе предложенных стратегий. Однако, в таком случае, одинаковые блоки большого размера шифруются одинаковым образом.

Использование алгоритма шифрования DESX по стратегии электронной кодовой книги позволяет достичь очень высокой скорости шифрования больших объемов данных и легко поддаётся разбиению на параллельные процессы, что делает такой подход легко масштабируемым и настраиваемым.

Список использованных источников:

1. Аграновский А. В, Хади Р. А. Практическая криптография (серия «Аспекты защиты»), М.: Солон-Пресс, 2002. 254 с.
2. Деднев, М. А. Защита информации в банковском деле и электронном бизнесе / М.А. Деднев. – М.: Кудиц-образ, 2004. – 186с.
3. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Брюс Шнайер. – М.Триумф, 2003. – 816 с Урядов В.Н.

АЛГОРИТМ РАБОТЫ СИСТЕМЫ УЧЕТА ЗАДОЛЖЕННОСТИ ПО БАНКОВСКИМ КРЕДИТНЫМ КАРТАМ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Ламчановский А.Г.

Бахтизин В.В. – к.т.н., профессор

Требования банков к функциональности систем учета клиентской задолженности по кредитным картам постоянно повышаются. Банки выводят на рынок кредитные карты с новыми условиями, увеличивается количество клиентов. Для учета и взыскания задолженности банки используют специальные информационные автоматизированные системы.

Просроченная задолженность – это непогашенная в срок задолженность по основному долгу или плановым процентам за пользование кредитом, а также иным платежам по кредитному договору. С момента возникновения просрочки по выплате по кредиту банк начисляет штрафы и пени, размер и принцип расчета которых указываются в кредитном договоре.

До сих пор не существует оптимального алгоритма решения проблемы взимания просроченной