УЯЗВИМОСТИ ВЕБ-ПРИЛОЖЕНИЙ

2016

POSITIVE TECHNOLOGIES

Оглавление

Введение

Деятельность любой организации так или иначе связана с веб-технологиями. Широкое распространение получили веб-порталы различных услуг (в том числе государственных), интернет-магазины, торговые площадки, различные бизнес-приложения, системы дистанционного банковского обслуживания. Невозможно представить себе современную организацию, будь то крупная корпорация или небольшая частная фирма, у которой не было бы своего официального сайта или странички на каком-либо публичном веб-ресурсе. Корпоративные приложения, для которых необходимо устанавливать клиентское ПО и регулярно его обновлять, уходят в прошлое. Веб-технологии позволяют значительно упростить бизнес-процессы.

Для того чтобы максимально использовать преимущества веб-технологий, необходимо обеспечить доступность ресурсов для целевой аудитории, например из сети Интернет. Но доступ, следовательно, могут получить и злоумышленники. Это и недобросовестные конкуренты, и другие категории нарушителей, руководствующихся преступными намерениями, например с целью хищения денежных средств, нарушения доступности ресурса или получения чувствительной информации. Компрометация приложений может привести как к репутационным потерям, так и к финансовым, в том числе в виде упущенной прибыли (например, если будет потерян важный клиент или сорвется сделка).

При всем при этом разработчики не всегда уделяют достаточно внимания защите веб-ресурсов, сосредоточиваясь в первую очередь на функциональности приложения; а администраторы систем зачастую недостаточно осведомлены в вопросах информационной безопасности и могут совершать ошибки, которые делают приложения уязвимыми. Подавляющее большинство владельцев веб-ресурсов не следуют принципам обеспечения безопасности на всех этапах жизненного цикла приложений (secure software development lifecycle, SSDL), вследствие чего уязвимости не выявляются на ранних стадиях разработки, а остаются в приложениях даже после их приемки в эксплуатацию, что играет на руку злоумышленникам.

Ежегодно специалисты Positive Technologies изучают порядка 250–300 веб-приложений в рамках различных работ, начиная от инструментального сканирования и заканчивая анализом исходного кода. Данный отчет содержит статистику, собранную в ходе работ по анализу защищенности веб-приложений, проведенных специалистами компании в 2015 году. Представлен также сравнительный анализ этих данных с результатами аналогичных исследований 2013 и 2014 годов. Это дает возможность оценить динамику развития современных веб-приложений с точки зрения информационной безопасности.

1. Методика исследования

По итогам выполненных в 2015 году проектов было выделено 30 веб-приложений, для которых проводился углубленный анализ с наиболее полным покрытием проверок. Результаты анализа защищенности веб-приложений, полученные в ходе инструментального сканирования или тестов на проникновение, в данном исследовании не затрагиваются. В него включены проекты, в рамках которых проводился анализ защищенности приложений с помощью анализатора исходных кодов. При этом учитывались только те уязвимости, которые были подтверждены на тестовом стенде.

Оценка защищенности проводилась как ручным способом методами черного, серого и белого ящика с использованием вспомогательных автоматизированных средств, так и автоматизированно, с помощью анализатора исходных кодов. Метод черного ящика заключается в оценке защищенности информационной системы от лица внешнего атакующего без предварительного получения от владельца какой-либо дополнительной информации о ней. Метод серого ящика аналогичен методу черного ящика, но в качестве нарушителя рассматривается пользователь, обладающий определенными привилегиями в системе. В методе белого ящика для оценки защищенности информационной системы используются все необходимые данные о ней, включая исходный код приложений.

Обнаруженные уязвимости классифицировались согласно соответствующим угрозам по системе Web Application Security Consortium Threat Classification (WASC TC v. 2), за исключением категорий Improper Input Handling и Improper Output Handling, поскольку они реализуются в рамках множества других атак.

В настоящей статистике учитываются только уязвимости, связанные с ошибками в коде и конфигурации веб-приложений. Другие распространенные проблемы информационной безопасности (к примеру, недостатки процесса управления обновлениями ПО) не рассматриваются.

Степень риска уязвимостей оценивалась согласно системе Common Vulnerability Scoring System (CVSS v. 2); на основе этой оценки выделялись качественные оценки высокого, среднего и низкого уровней риска.

2. Резюме

В результате проведенных работ были сделаны следующие основные выводы:

Все веб-приложения уязвимы

Недостатки как минимум среднего уровня риска были обнаружены во всех исследованных приложениях. При этом в 70% систем были обнаружены критически опасные уязвимости. В течение последних трех лет доля систем, подверженных недостаткам высокой степени риска, постепенно повышается в пределах 60–70%, что свидетельствует о недостаточном внимании разработчиков к проблемам качества кода, а администраторов — к безопасности конфигурации приложений.

Пользователи приложений не защищены от атак

Большинство приложений позволяют проводить атаки на пользователей. Например, уязвимость «Межсайтовое выполнение сценариев» была обнаружена в 80% веб-приложений, а «Открытое перенаправление» — практически в каждой третьей системе (30%).

Разглашение информации все так же актуально

В 2015 году существенно снизилась доля приложений, в которых раскрывается информация об используемых версиях ПО. Однако при этом любой внешний нарушитель может получить другую, более чувствительную информацию (например, исходные коды приложений, пути установки, персональные данные пользователей). Уязвимость «Утечка данных» была обнаружена в каждом втором веб-приложении.

Веб-приложения на Java не менее уязвимы

Среди веб-приложений, рассмотренных в 2015 году, большинство были разработаны с помощью Java, и лишь 30% на языке PHP. Результаты предыдущих лет показали, что PHP-приложения, как правило, более уязвимы, чем системы, разработанные с помощью ASP.NET и Java. Однако на сегодняшний день картина изменилась: 69% Java-приложений подвержены критически опасным уязвимостям, а для PHP-систем этот показатель составил 56% (это ниже уровня 2013 года на 20%).

Конфигурации веб-серверов не уделяется должного внимания

Наиболее распространенные ошибки администрирования серверов практически не изменились. Как и прежде, это разглашение информации и недостаточная защита от подбора учетных данных. Эти недостатки характерны как для Apache Tomcat и WebLogic, так и для Nginx и Microsoft IIS.

Приложения банков и IT-компаний по-прежнему под угрозой

Как и в предыдущем году, веб-приложения банков оказались наиболее уязвимы, в каждой из таких систем были обнаружены критически опасные уязвимости. Аналогичная ситуация наблюдается в сфере IT. Положительная динамика была отмечена лишь для приложений промышленных и телекоммуникационных компаний.

Продуктивные системы недостаточно защищены

Доля уязвимых приложений, находящихся в эксплуатации, очень велика. Более половины таких ресурсов (63%) подвержены критически опасным уязвимостям. Эти недостатки могут позволить нарушителю установить полный контроль над системой (например, в случае загрузки произвольных файлов или выполнения команд), а также получать чувствительную информацию (например, в результате эксплуатации уязвимостей «Внедрение операторов SQL» или «Внедрение внешних сущностей XML»). Нарушитель может также

проводить успешные атаки, направленные на отказ в обслуживании. Для защиты систем, находящихся в эксплуатации, рекомендуется применять межсетевой экран уровня приложения (web application firewall).

Анализ исходного кода необходим

Анализ защищенности веб-приложений методом белого ящика позволяет выявить в среднем в 5 раз больше критически опасных уязвимостей в каждой системе. Для недостатков среднего уровня риска этот показатель выше вдвое. Была продемонстрирована высокая эффективность автоматизированных средств анализа исходных кодов приложений.

3. Портрет участников

Исследованные приложения принадлежали организациям, представляющим различные экономические отрасли и сферы деятельности, — промышленным, телекоммуникационным и IT-компаниям, государственным органам, средствам массовой информации, финансовым учреждениям.

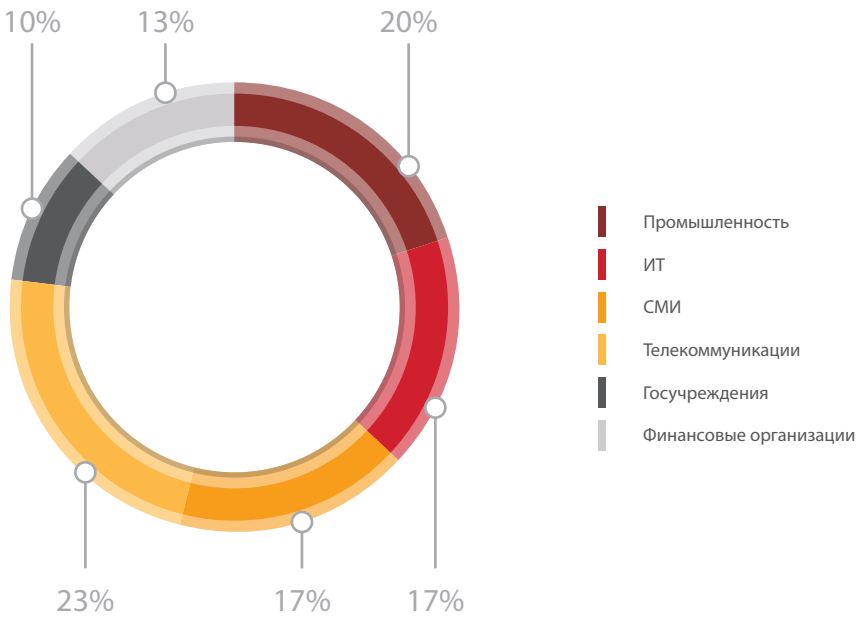

Рис. 1. Распределение систем по отраслям экономики

При разработке веб-приложений использовались различные технологии. Большинство приложений, вошедших в выборку, разработаны на базе Java и PHP. Также встречались приложения, созданные с использованием технологий ASP.NET, Perl, ABAP, 1C и других (их доля незначительна, и они были объединены в категорию «Другие»).

С точки зрения используемых веб-серверов были выделены четыре основные группы веб-приложений — приложения, функционирующие на базе Nginx, Microsoft Internet Information Services (IIS), Apache Tomcat, а также сайты, использующие веб-сервер WebLogic. Часть приложений функционировала под управлением серверов Apache и SAP NetWeaver Application Server, но их доля оказалась невелика. Наиболее распространенным веб-сервером в 2015 году стал Nginx: он обеспечивает работу 34% проанализированных веб-приложений.

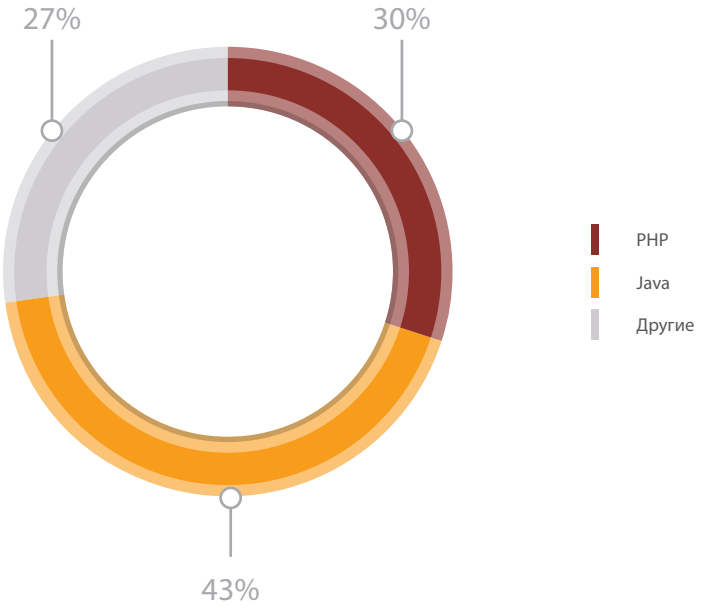

Рис. 2. Распределение систем по средствам разработки

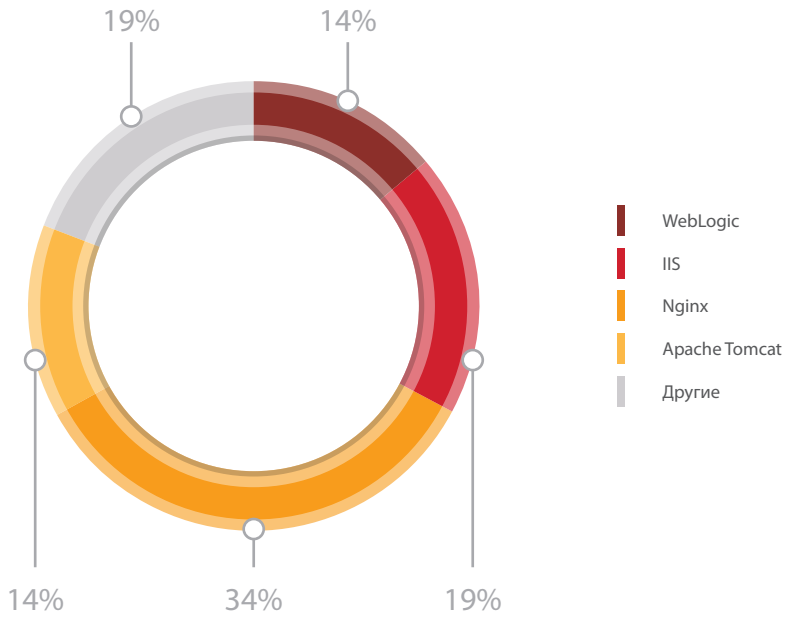

Рис. 3. Системы на базе различных веб-серверов

Исследуемые веб-приложения находились на различных стадиях разработки: продуктивные системы, доступные пользователям сети Интернет, а также тестовые площадки, находящиеся на финальной стадии разработки или в стадии приемки в эксплуатацию. Продуктивных систем было немного больше — 53%.

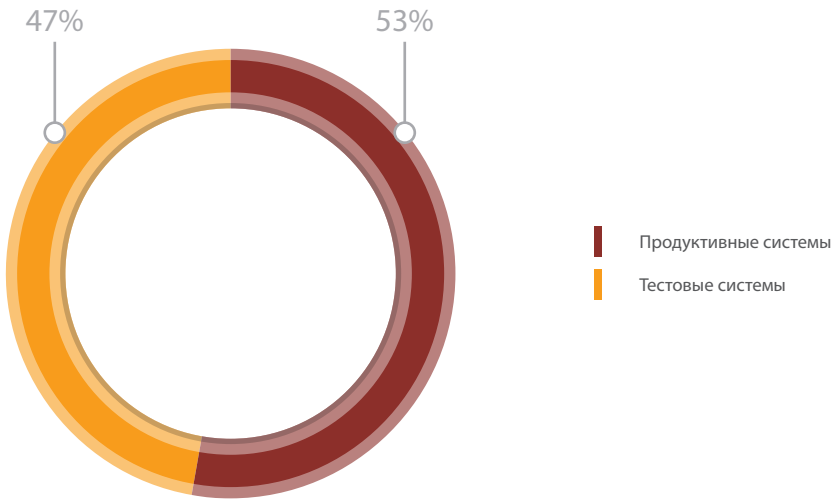

Рис. 4. Исследовались два вида систем

Как и в предыдущие годы, коммерческие системы управления содержимым (CMS) применялись для рассмотренных приложений крайне редко, в связи с чем статистические исследования уровня защищенности сайтов в зависимости от используемой системы CMS не проводились.

4. Статистика уязвимостей

Данная глава содержит анализ распространенности и уровней риска уязвимостей различных типов, классифицированных согласно угрозам, представленным в WASC TC v. 2.

4.1. Наиболее распространенные уязвимости

В результате анализа защищенности веб-приложений в 2015 году было установлено, что каждый третий выявленный недостаток критически опасен (34%). А большинство выявленных уязвимостей характеризуются средним уровнем риска (62%).

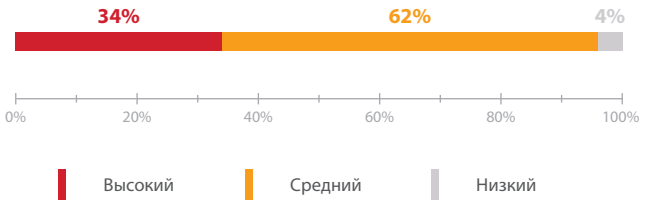

Рис. 5. Доля уязвимостей различной степени риска

Все исследованные веб-приложения оказались подвержены недостаткам как минимум среднего уровня риска. Полученные данные свидетельствуют о том, что с каждым годом увеличивается доля систем, в которых присутствуют критически опасные уязвимости. Веб-технологии с каждым годом набирают все большую популярность, вместе с

функциональностью возрастает и сложность реализации приложений. Разработчикам становится труднее при создании систем выявлять и исправлять ошибки, которые могут приводить к возникновению критически опасных уязвимостей.

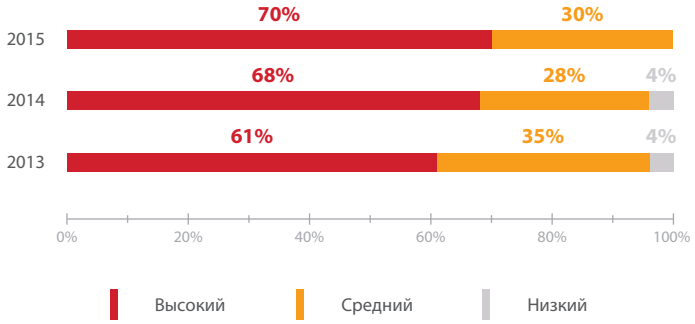

Рис. 6. Максимальная степень риска уязвимостей (доли сайтов)

Доля систем, содержащих недостатки высокой степени риска, в последние три года постепенно увеличивается в пределах 60–70%. Доля веб-приложений, подверженных уязвимостям средней степени риска, как и прежде велика, и в 2015 году достигла максимального значения. Процент веб-приложений, в ходе анализа которых были обнаружены уязвимости низкой степени риска, существенно снизился (с 80 до 50%). Предположительно, это объясняется тем, что подобные недостатки сравнительно просто выявить и устранить без существенного вмешательства в код приложения (например, уязвимость «Раскрытие информации об используемом ПО» стала встречаться вдвое реже).

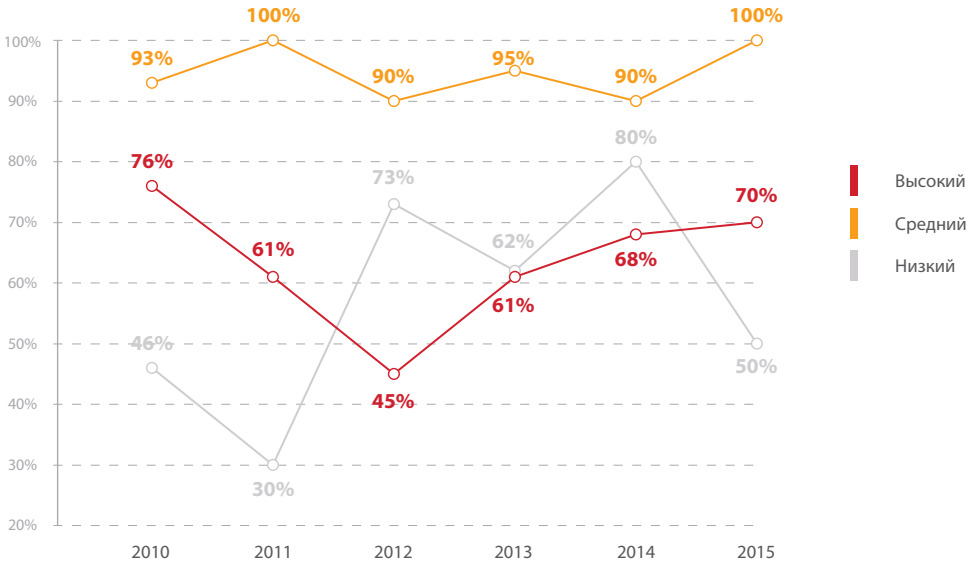

Рис. 7. Доля сайтов с уязвимостями различной степени риска

В 2015 году получила распространение уязвимость среднего уровня риска «Межсайтовое выполнение сценариев» (Cross-Site Scripting, XSS), занимавшая вторую строчку рейтинга в 2014 году. Данная ошибка была обнаружена в программном коде 80% исследованных

ресурсов. В результате ее эксплуатации злоумышленник может внедрить в браузер пользователя произвольные HTML-теги, включая сценарии на языке JavaScript и других языках, и таким образом получить значение идентификатора сессии атакуемого и совершить иные неправомерные действия, например фишинговые атаки.

Второе место занимает недостаток безопасности «Утечка информации» (Information Leakage): данная проблема была обнаружена в 50% приложений. Зачастую на страницах приложений разглашается такая чувствительная информация, как фрагменты исходных кодов приложения, персональные и учетные данные пользователей, конфигурация сервера. Нарушитель может использовать полученные данные при планировании и проведении атак. Около половины веб-сайтов содержат также уязвимости, связанные с отсутствием защиты от подбора учетных данных (Brute Force).

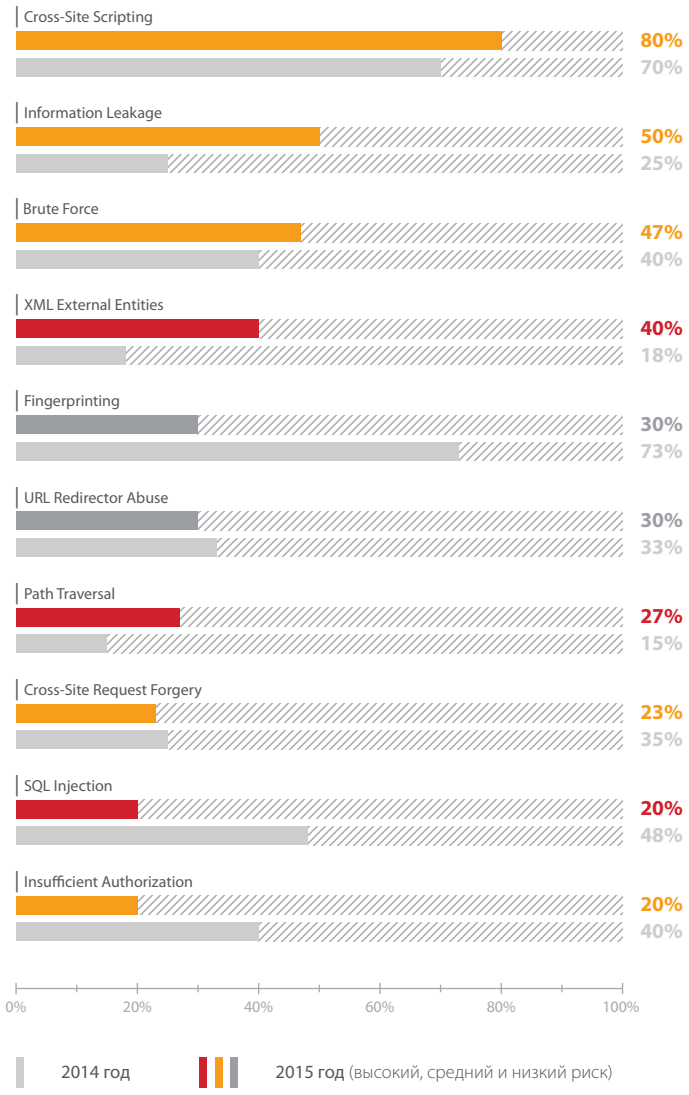

Рис. 8. Наиболее распространенные уязвимости (доля сайтов)

Критически опасная уязвимость «Внедрение операторов SQL» (SQL Injection) вновь вошла в топ-10, однако в 2015 году заняла лишь 9-ю строчку. Наиболее распространенным недостатком высокого уровня риска в 2015 году стала уязвимость «Внедрение внешних

сущностей XML» (XML External Entities): она занимает 4-е место по распространенности. Эта ошибка в программном коде позволяет злоумышленнику получить содержимое файлов, расположенных на атакуемом сервере, либо совершать запросы в локальную сеть от имени атакуемого сервера. Она обусловлена недостаточной проверкой приложением данных, поступающих от пользователя, что позволяет злоумышленнику осуществлять атаку, направленную на изменение логики запроса посредством внедрения произвольной DTD-схемы в тело XML-документа. Это может привести к разглашению важных данных, получению злоумышленником исходных кодов приложения, файлов конфигурации и другой чувствительной информации о системе, а также к отказу в обслуживании приложения.

Значительно снизилась доля веб-приложений, в которых было выявлено раскрытие информации о версиях используемого ПО (Fingerprinting); данный недостаток занимал первую строчку рейтинга в 2014 году.

Большинство уязвимостей выявлены в серверной части (67%). Среди них наиболее распространены утечка информации (Information Leakage) и отсутствие защиты от подбора учетных данных (Brute Force). Вдвое меньше оказалась доля недостатков, позволяющих осуществлять атаки на клиентов приложения. В данной категории следует выделить межсайтовое выполнение сценариев (Cross-Site Scripting, XSS), открытое перенаправление (URL Redirector Abuse) и подделку межсайтовых запросов (Cross-Site Request Forgery).

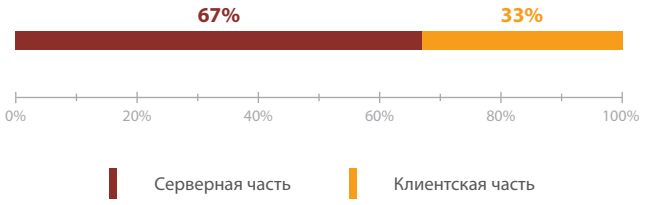

Рис. 9. Распределение уязвимостей по атакуемым частям приложения

Как и в предыдущие годы, большинство уязвимостей веб-приложений вызваны ошибками в программном коде, допущенными разработчиками (69%). Оставшийся 31% недостатков связан с некорректной конфигурацией.

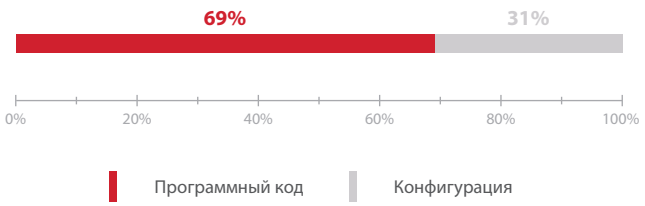

Рис. 10. Происхождение уязвимостей

4.2. Уязвимости, характерные для различных средств разработки

Более половины исследованных веб-сайтов, разработанных на PHP, содержит критически опасные уязвимости (56%). Соответствующее значение для веб-ресурсов, разработанных с использованием технологии Java, оказалось выше и составило 69%. Практически все сайты, созданные с помощью других языков программирования, подвержены уязвимостям высокой степени риска (88%).

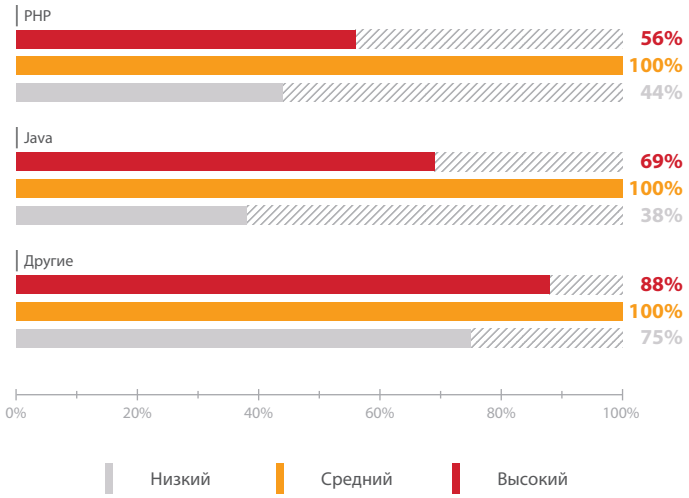

Рис. 11. Доли систем с уязвимостями разной степени риска (по средствам разработки)

Вне зависимости от языка разработки все ресурсы, вошедшие в исследование, содержат уязвимости как минимум средней степени опасности.

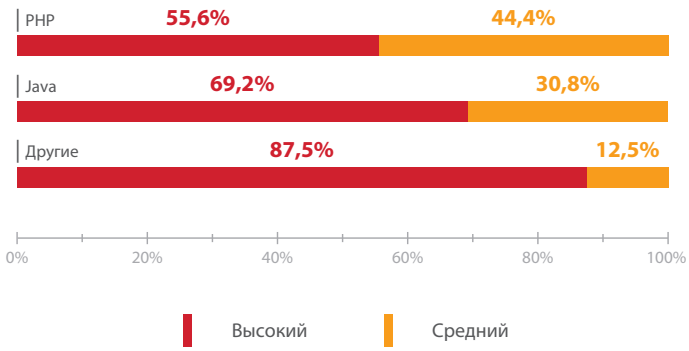

Рис. 12. Доля веб-приложений по максимальному уровню риска уязвимостей

Каждое веб-приложение, разработанное на языке программирования PHP, в среднем содержит 9,1 критически опасной уязвимости, что близко к показателю 2014 года (11 уязвимостей). Для Java данный показатель оказался несколько выше — 10,5. Для всех других языков программирования и средств разработки в среднем на каждую систему приходится лишь 2 критически опасные уязвимости.

Вне зависимости от языка разработки на каждую систему в среднем приходится более 10 уязвимостей среднего уровня риска. Для PHP данный показатель оказался лучше значения предыдущего года примерно вдвое.

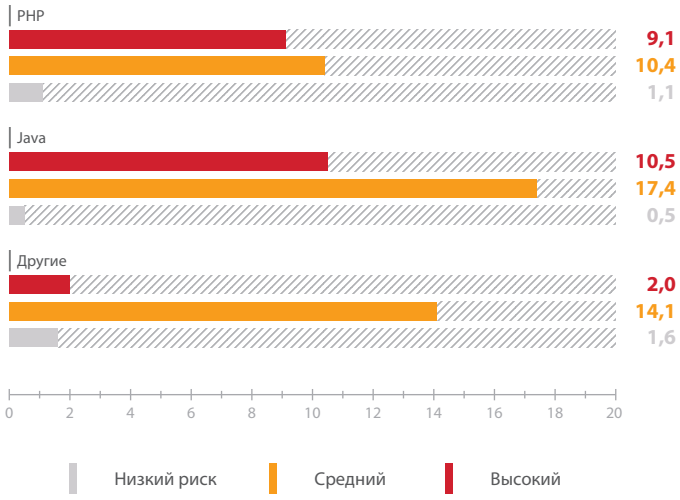

Рис. 13. Среднее число уязвимостей на одну систему

В таблице представлены данные о распространенности уязвимостей веб-приложений для различных средств разработки.

Таблица 1. Наиболее распространенные уязвимости (по средствам разработки)

PHP	Доля сайтов	Java	Доля сайтов	Другие	Доля сайтов
Cross-Site Scripting	89%	Cross-Site Scripting	77%	Cross-Site Scripting	75%
Information Leakage	56%	XML External Entities	54%	Information Leakage	75%
Brute Force	33%	Brute Force	46%	Brute Force	63%
OS Commanding	22%	Path Traversal	31%	Fingerprinting	60%
SQL Injection	22%	Information Leakage	31%	XML External Entities	50%
Path Traversal	22%	URL Redirector Abuse	31%	Cross-Site Request Forgery	38%
Insufficient Authorization	22%	SQL Injection	23%	Insufficient Transport Layer Protection	38%
Fingerprinting	22%	Cross-Site Request Forgery	23%	URL Redirector Abuse	38%
URL Redirector Abuse	22%	Application Misconfiguration	23%	Path Traversal	25%
XML External Entities	11%	HTTP Response Splitting	23%	Insufficient Authorization	25%

Уязвимость «Межсайтовое выполнение сценариев» оказалась наиболее распространенной для всех языков программирования. Ей подвержены от 75 до 89% ресурсов в соответствующих категориях. Также важно отметить, что в десятку наиболее распространенных уязвимостей для всех средств разработки вошли такие критически опасные уязвимости, как внедрение внешних сущностей XML (XML External Entities) и выход за пределы назначенного каталога (Path Traversal).

Доля приложений, подверженных уязвимости «Внедрение операторов SQL», сократилась по сравнению с 2014 годом. Данный недостаток встречается в 22% сайтов, разработанных на языке PHP, и в 23% приложений, созданных с помощью Java. В предыдущем году эта уязвимость была выявлена в 67% веб-ресурсов, разработанных на PHP.

Данные о долях веб-ресурсов, которые подвержены наиболее распространенным в 2015 году уязвимостям, для различных средств разработки представлены на графиках ниже.

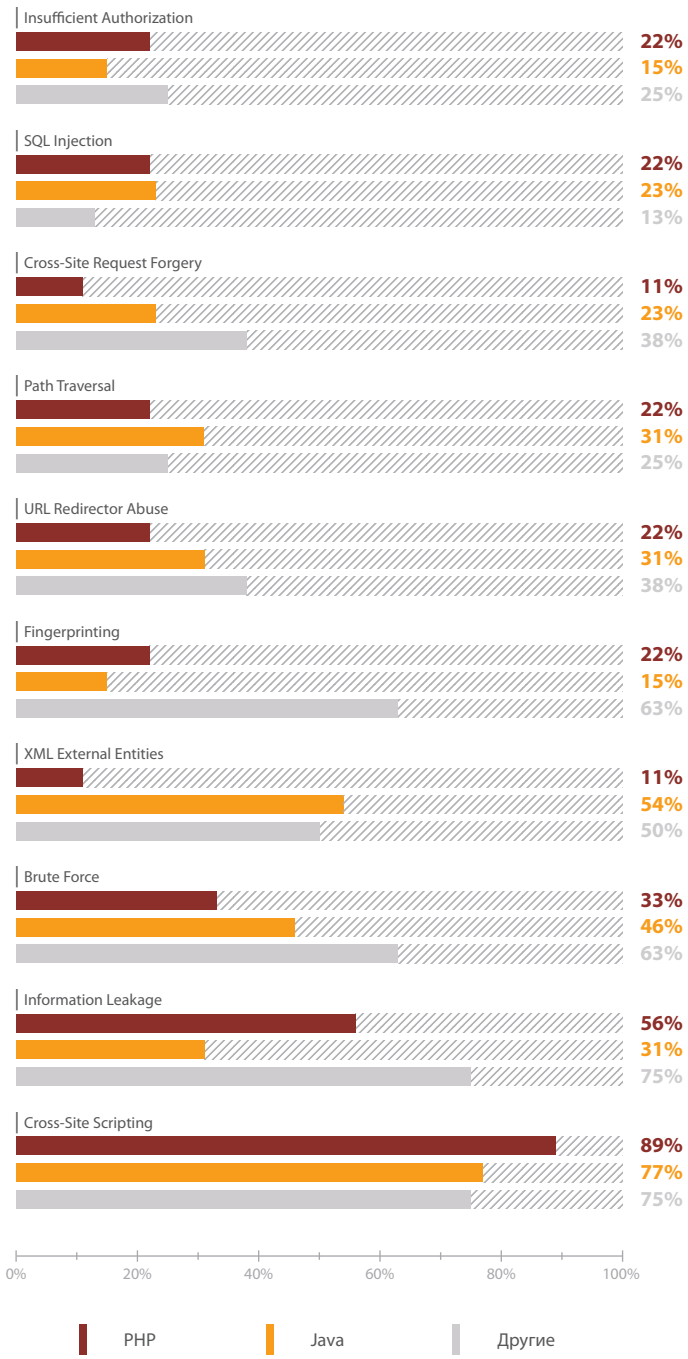

Рис. 14. Доли приложений с наиболее распространенными уязвимостями

Таким образом, во всех исследованных системах, независимо от средства разработки, была выявлена каждая из наиболее распространенных уязвимостей веб-приложений.

4.3. Уязвимости, характерные для различных веб-серверов

В результате анализа защищенности было установлено, что каждое второе веб-приложение под управлением сервера Nginx (57%) содержит уязвимости высокого уровня риска, что существенно меньше аналогичного показателя 2014 года (86%) и в точности совпадает с соответствующим значением 2013 года. Доля уязвимых ресурсов, функционирующих на базе Microsoft IIS, значительно возросла по сравнению с предыдущими периодами исследований и достигла максимального значения. При этом доля уязвимых сайтов под управлением веб-сервера Apache Tomcat снизилась почти вдвое и составила 33%. В предыдущие годы приложения под управлением сервера WebLogic не анализировались, а в 2015 году в 67% таких приложений были выявлены критически опасные уязвимости.

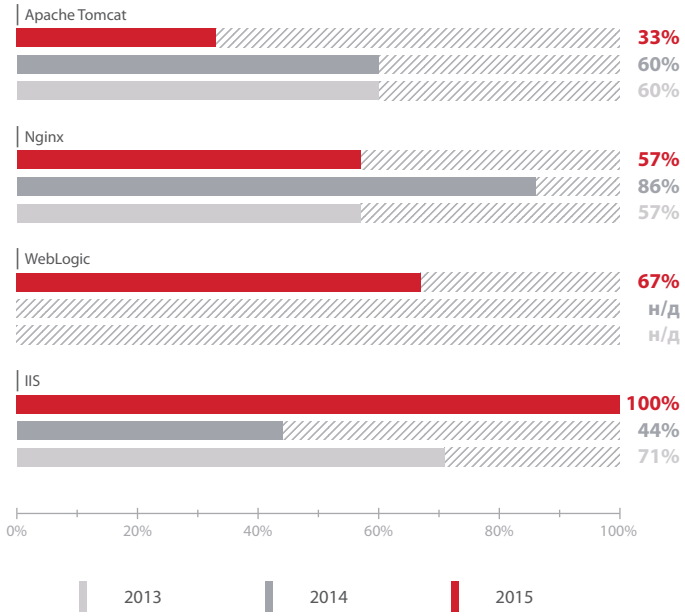

Рис. 15. Веб-приложения с уязвимостями высокой степени риска

Ежегодные исследования показывают, что уровень защищенности ресурса не зависит напрямую от типа используемого веб-сервера, если его владелец своевременно устанавливает все необходимые обновления ПО. Анализ защищенности веб-серверов не входил в границы работ, на основе которых проводилось данное исследование. Критически опасные уязвимости, показанные на рис. 15, выявлены в приложении, а не в самом веб-сервере. Некоторые уязвимости веб-приложений, выделенные согласно классификации WASC TC v. 2, являются следствием некорректного администрирования веб-приложений. Настоящий раздел направлен на выявление закономерностей в ошибках администрирования тех или иных серверов.

Для большинства исследованных веб-серверов самой распространенной ошибкой администрирования является «Утечка информации» (Information Leakage). Данный недостаток был обнаружен во всех исследованных приложениях под управлением серверов Microsoft IIS. Второе место рейтинга в 2015 году для большинства серверов занял недостаток «Отсутствие защиты от подбора учетных данных» (Brute Force), который стал наиболее распространенной уязвимостью в приложениях под управлением веб-сервера Nginx. Разглашение информации о версиях используемого ПО (Fingerprinting) для большинства веб-серверов находится лишь на третьем месте по распространенности, а в приложениях под управлением сервера WebLogic не было выявлено вовсе.

Таблица 2. Уязвимости, связанные с ошибками администрирования

Nginx	Доля сайтов	IIS	Доля сайтов	Apache Tomcat	Доля сайтов	WebLogic	Доля сайтов
Brute Force	57%	Information Leakage	100%	Information Leakage	67%	Information Leakage	67%
Information Leakage	43%	Brute Force	75%	Brute Force	33%	Brute Force	33%
Fingerprinting	43%	Fingerprinting	75%	Fingerprinting	33%	Insufficient Transport Layer Protection	33%
Insufficient Transport Layer Protection	14%	Insufficient Transport Layer Protection	50%			Server Misconfiguration	33%
Server Misconfiguration	14%	Server Misconfiguration	25%			Application Misconfiguration	33%
		Application Misconfiguration	25%				

4.4. Статистика по отраслям экономики

Наиболее уязвимыми в 2015 году оказались веб-приложения финансовых организаций и IT-компаний: в каждом из них были выявлены те или иные критически опасные уязвимости. В предыдущем году также именно приложения банковского сектора были наиболее уязвимы. Мы предполагаем, это вызвано тем, что финансовые организации в первую очередь уделяют внимание безопасности систем, непосредственно связанных с денежными расчетами (например, платежных систем и систем дистанционного банковского обслуживания), и лишь затем — защите других приложений. Анализ наиболее распространенных уязвимостей банковского ПО представлен в статье «Уязвимости приложений финансовой отрасли (2015 г.)».

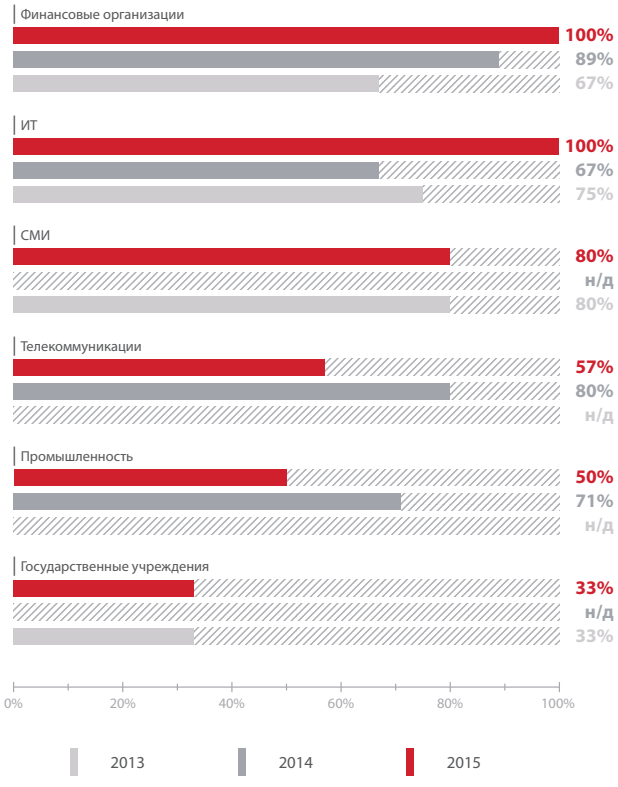

Рис. 16. Доли приложений с уязвимостями высокого уровня риска в различных отраслях экономики

В 80% исследованных веб-приложений СМИ были обнаружены уязвимости высокой степени риска, что соответствует уровню 2013 года. Высокий процент веб-приложений, подверженных критически опасным уязвимостям, отмечается также в телекоммуникационной отрасли и в промышленности, где каждая вторая система в 2015 году содержала критически опасную уязвимость. Важно отметить, что этот показатель существенно лучше, чем в 2014 году.

Наименее уязвимыми оказались веб-приложения государственных организаций. Тем не менее доля систем с критически опасными уязвимостями и здесь довольно высока — 33%, что соответствует уровню 2013 года. Необходимо отметить, что, как и в 2013 году, выборка госорганизаций не так велика, чтобы говорить об уровне защищенности веб-приложений по отрасли в целом. Как правило, в государственных учреждениях анализ защищенности проводится только для единичных систем, безопасности которых уделяется повышенное внимание. В целом же реальная безопасность государственных веб-ресурсов ниже, чем можно заключить по результатам таких исследований, поскольку в большинстве госорганизаций проводить регулярные работы по анализу защищенности веб-приложений не принято. Для защиты приложений в государственных структурах — при отсутствии регулярных проверок уровня защищенности — рекомендуется применять межсетевой экран уровня приложений (WAF).

В 2013 и 2014 годах объем выборки по ряду сфер экономики был недостаточным, поэтому сравнения результатов прошлых лет с показателями 2015 года для этих сфер мы не приводим.

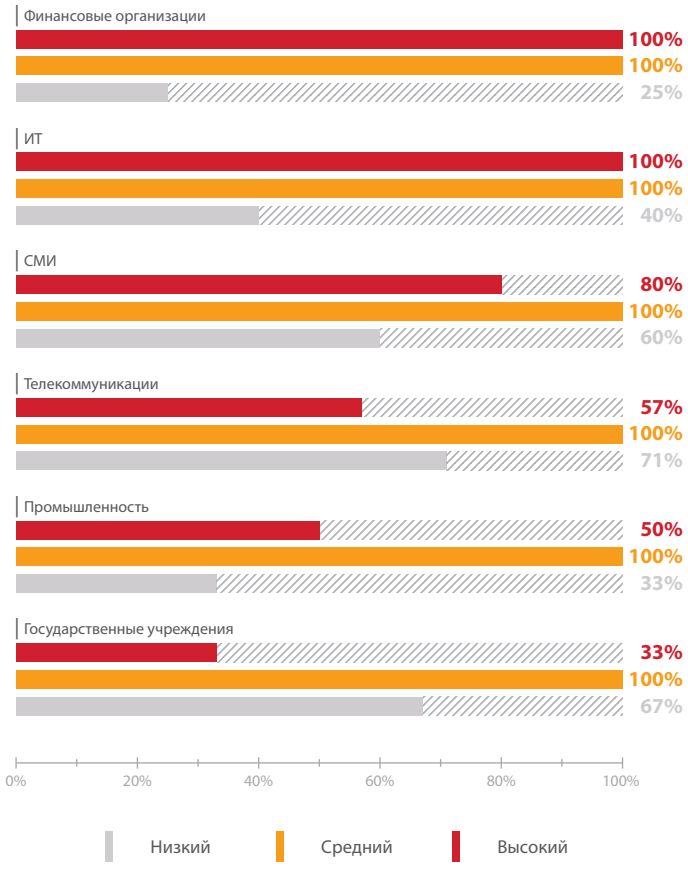

Рис. 17. Доли сайтов с уязвимостями различного уровня риска

По соотношению среднего количества уязвимостей на одну систему наименее защищенными также оказались ресурсы финансовых организаций и IT-компаний, где на одно приложение приходится более 3 критически опасных уязвимостей. (В финансовых организациях данный показатель и вовсе составил 6,3.) Выявлены критически опасные уязвимости и в приложениях телекоммуникационных и промышленных компаний (2 и 1,5 соответственно). Среднее количество критически опасных ошибок в веб-ресурсах СМИ и госучреждений ниже, но такой результат стал, вероятно, следствием применения метода белого ящика (в том числе — использования автоматизированного анализатора защищенности исходного кода приложений) при анализе систем из всех прочих отраслей. Можно предположить, что при исследовании государственных систем и приложений СМИ методом белого ящика — вскроются дополнительные уязвимости и результат будет иным.

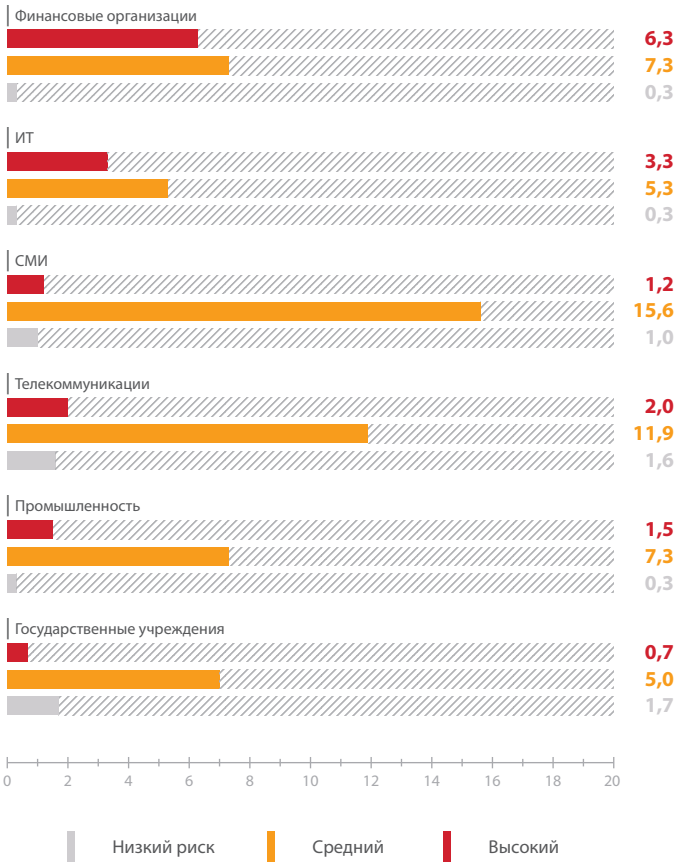

Рис. 18. Среднее количество уязвимостей на одну систему

В 2015 году критически опасные уязвимости «Внедрение внешних сущностей XML» (XML External Entities), «Внедрение операторов SQL» (SQL Injection), «Выполнение команд ОС» (OS Commanding) и «Выход за пределы назначенного каталога» (Path Traversal) встречались чаще, чем другие недостатки этого уровня риска. Однако не было выявлено ни одного приложения, в котором бы все такие перечисленные уязвимости присутствовали одновременно.

Результаты исследования показывают, что веб-приложения всех отраслей экономики в той или иной степени подвержены опасным уязвимостям, при этом в области информационных технологий и финансов таким недостаткам подвержены все исследованные системы.

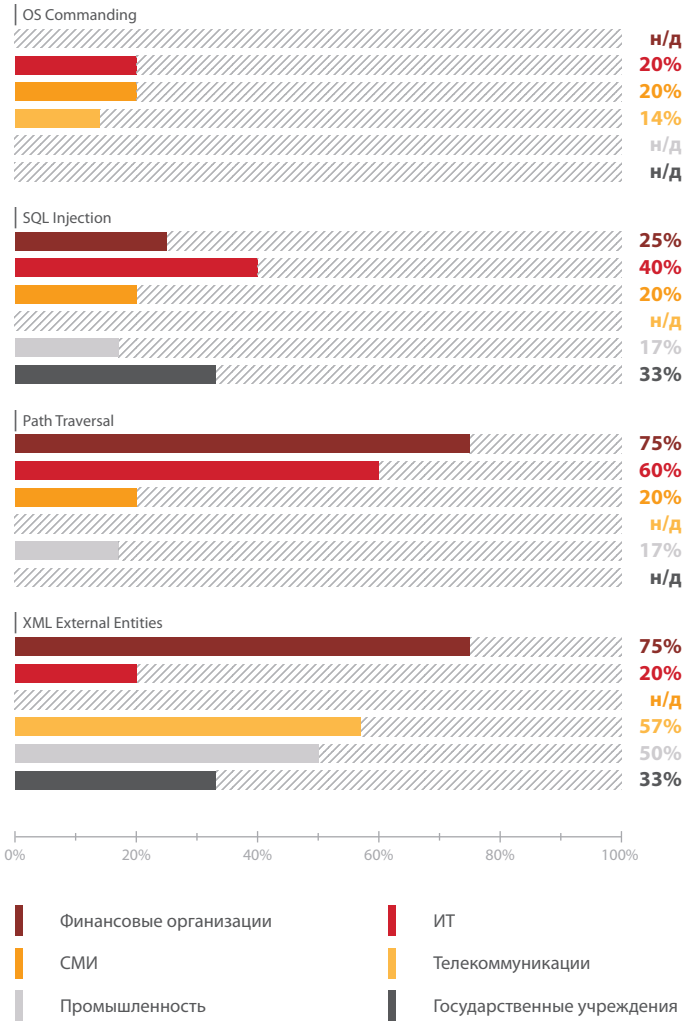

Рис. 19. Доли уязвимых сайтов из различных отраслей экономики

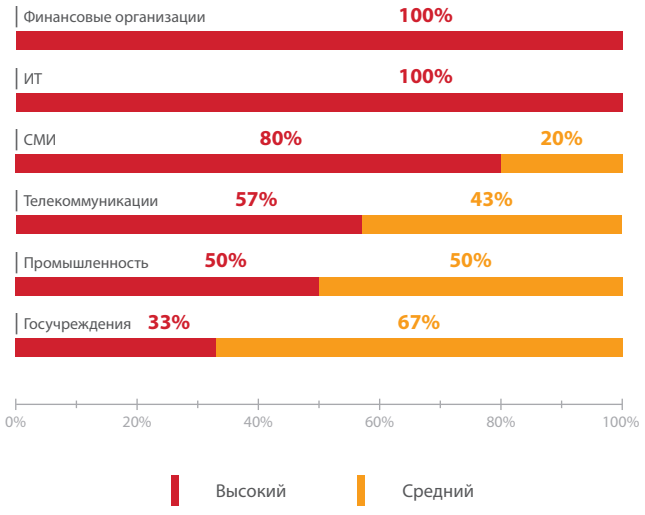

Рис. 20. Доля веб-приложений по максимальному уровню риска уязвимостей

4.5. Сравнительный анализ методов тестирования

В 2015 году большинство проектов осуществлялись методами черного и серого ящика. Исходный код приложений был проанализирован в 40% реализованных проектов, для 30% применялся автоматизированный анализ.

Специалисты Positive Technologies провели сравнительный анализ методов тестирования «черный ящик» и «серый ящик» — с методом «белый ящик». Важно отметить, что в рамках двух проектов при анализе исходного кода было выявлено несколько десятков критически опасных уязвимостей — внедрение операторов SQL, выполнение команд ОС и выход за пределы назначенного каталога, — а также более сотни недостатков конфигурации приложения. В рамках другого проекта, который проводился методом серого ящика, было выявлено несколько десятков уязвимостей «Межсайтовое выполнение сценариев». Результаты данных работ не были учтены при подсчетах, так как не отражают общую картину и существенно искажают усредненные статистические данные для всех исследованных приложений.

Как было показано ранее, уязвимости среднего уровня риска были выявлены абсолютно во всех проектах. Доля систем, подверженных критически опасным уязвимостям, которые были обнаружены методом черного ящика, существенно ниже, чем аналогичный показатель для проектов, в рамках которых осуществлялся анализ исходных кодов приложения. При этом доля таких систем довольно велика — 59% (в случае методов черного и серого ящика). Таким образом, отсутствие у атакующего доступа к исходным кодам не делает веб-приложения защищенными: для множества систем уязвимости высокого и среднего уровней риска могут быть выявлены и проэксплуатированы внешним атакующим или злоумышленником с правами авторизованного пользователя без дополнительных знаний о системе. То же было и в 2014 году.

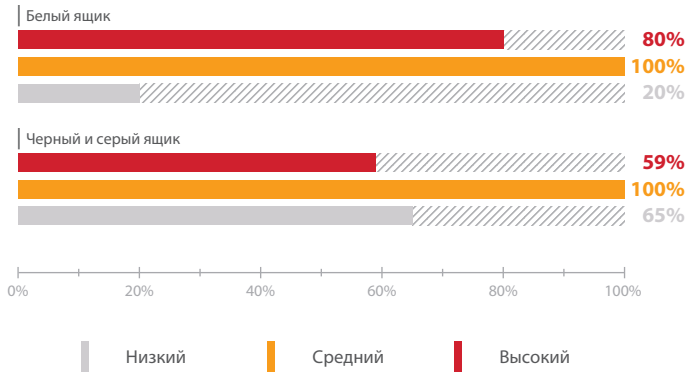

Рис. 21. Доли систем с уязвимостями разного уровня риска — по методу тестирования

При этом среднее количество выявленных уязвимостей различного уровня риска на одну систему для работ методом белого ящика существенно выше соответствующих значений при тестировании методами черного и серого ящиков. В частности, анализ исходных кодов позволил выявить в несколько раз больше уязвимостей высокой и средней степени риска на одну систему. А если принять во внимание те проекты, которые были исключены из расчетов, количество критически опасных уязвимостей и недостатков среднего уровня риска, приходящееся в среднем на одну систему, исследованную методом белого ящика, существенно возрастет. Данное предположение подтверждается при оценке результатов работ по анализу защищенности приложений ручными методами и с помощью анализатора исходного кода (см. раздел 4.5.1), где все эти данные были учтены.

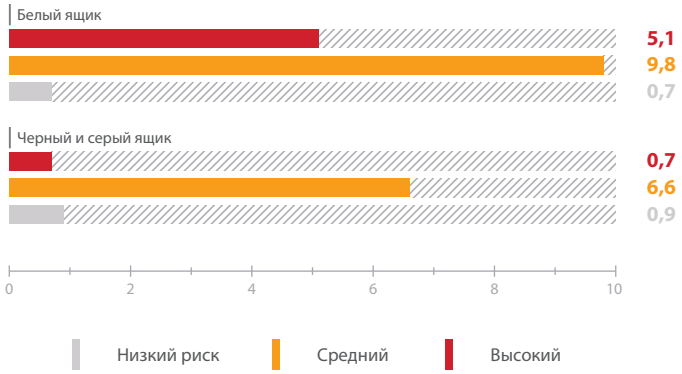

Рис. 22. Среднее количество обнаруженных уязвимостей на одну систему для различных методов тестирования

Для уязвимостей типа «Межсайтовое выполнение сценариев» среднее количество обнаруженных недостатков для двух методов тестирования различается всего на единицу. А в случае критически опасных уязвимостей «Выполнение команд ОС» и «Внедрение внешних сущностей XML» анализ исходных кодов оказался существенно эффективнее.

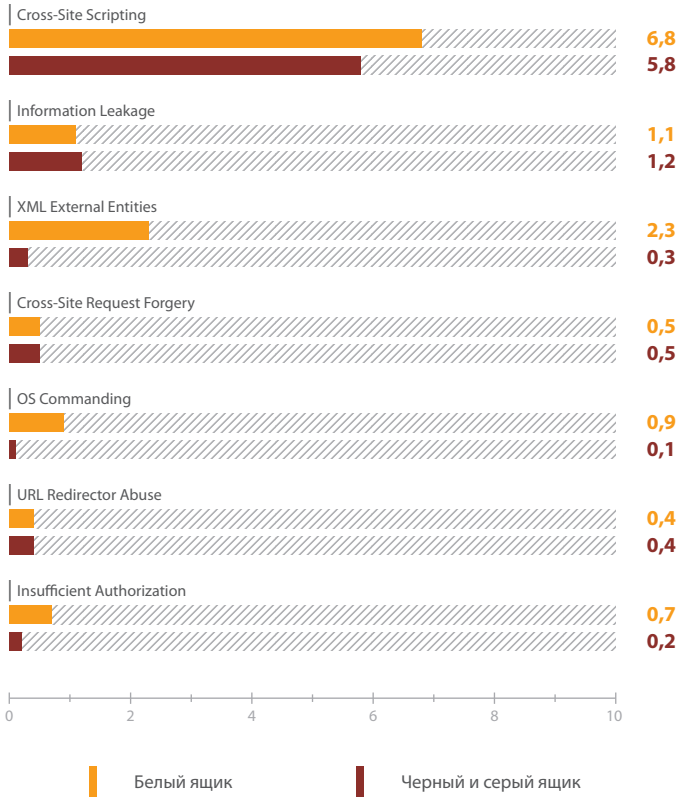

Рис. 23. Среднее количество различных уязвимостей на одну систему — по методу тестирования

Важно понимать, что недостатки, не обнаруженные при анализе защищенности приложения методами черного и серого ящиков, могут быть выявлены потенциальным нарушителем, имеющим аналогичный уровень привилегий. Например, при тестировании продуктивной системы методом черного ящика не проводятся проверки, связанные с

высоким риском отказа в обслуживании, — однако для злоумышленника подобные атаки зачастую являются основной целью. Таким образом, можно сделать вывод о высокой эффективности метода белого ящика и необходимости проведения анализа исходных кодов при оценке общего уровня защищенности приложения.

4.5.1. Оценка автоматизированного метода тестирования

В рамках исследования была проведена оценка результатов работ, осуществленных методом белого ящика вручную и с использованием автоматизированного сканера защищенности исходного кода приложений. Так как эти типы работ проводились в отношении различных систем, в данном разделе приведены лишь усредненные данные по всем исследованным приложениям. Эти данные могут быть использованы лишь для общей оценки эффективности каждого метода, без их сравнения между собой.

Порядка 40% всех выявленных с помощью анализатора уязвимостей критически опасные, почти все остальные недостатки характеризуются средним уровнем риска. Это объясняется тем, что анализатор классифицирует уязвимости в соответствии с заданными разработчиками принципами, не опираясь на классификацию WASC TC v. 2, которая принята в рамках настоящего исследования. Часть уязвимостей низкой степени риска могут быть классифицированы как недостатки конфигурации (Application Misconfiguration), которые в рамках данного исследования характеризуются средним уровнем риска.

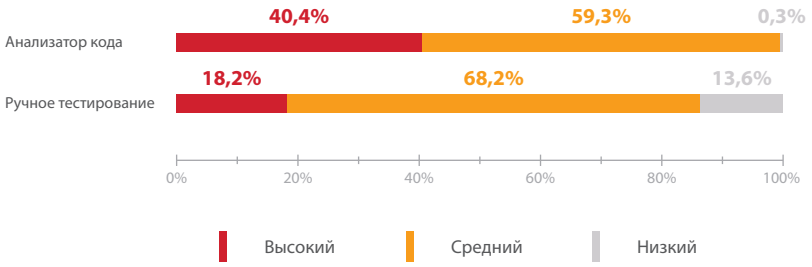

Рис. 24. Доли уязвимостей различного уровня риска

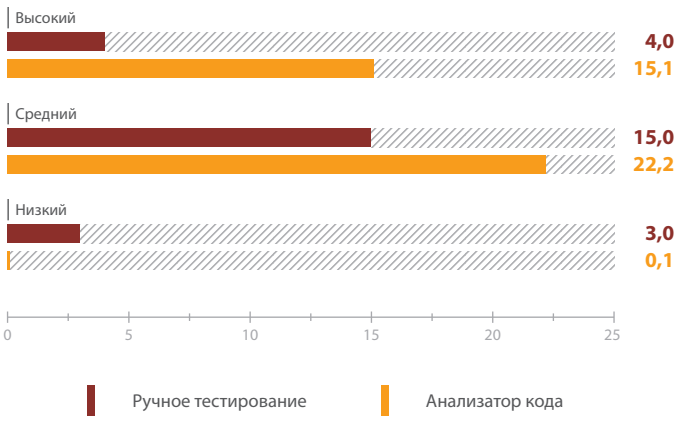

Рис. 25. Среднее количество уязвимостей определенного уровня риска на одну систему

В среднем в каждой системе при использовании анализатора кода было выявлено около 15 критически опасных уязвимостей и более 20 недостатков среднего уровня риска. Недостатков низкого уровня опасности выявлено существенно меньше, это объясняется упомянутыми различиями в классификации выявленных недостатков. Согласно полученным данным, анализатор исходных кодов позволил выявить большее число различных типов уязвимостей. Однако, как мы отмечали выше, работы проводились в отношении разных систем, и результаты автоматизированного и ручного методов не могут сравниваться между собой.

На диаграммах ниже представлены соответствующие рейтинги наиболее распространенных уязвимостей, выявленных ручным и автоматизированным методами.

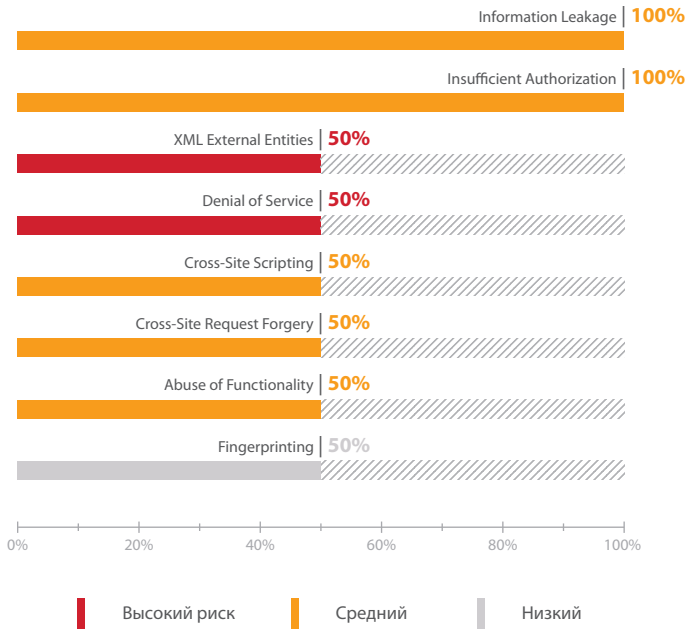

Рис. 26. Уязвимости, выявленные ручными методами (доля систем)

Как показало проведенное исследование, в целом анализ защищенности методом белого ящика существенно эффективнее других методов, при которых не анализируется исходный код приложения. При этом полученные результаты свидетельствуют, что эффективны как ручные методы анализа кода, так и автоматизированные.

Объемы кода современных приложений достаточно велики, и зачастую в них используется множество библиотек, которые также нужно анализировать. На такую работу ручными методами может уйти от нескольких дней до нескольких недель. При этом к таким работам часто привлекаются несколько экспертов — для минимизации рисков, связанных с человеческим фактором (специалист может упустить из виду какую-либо уязвимость). Автоматизированное же тестирование позволяет выполнить аналогичную задачу за несколько дней даже в случае сложных систем. Таким образом, можно выделить важные преимущества анализа защищенности приложения с помощью анализатора кодов по сравнению с ручным тестированием:

+ скорость проведения работ;
+ исключение человеческого фактора при поиске уязвимостей;
+ возможность анализа исходных кодов на любом этапе разработки приложения — без приобретения услуги по анализу защищенности у специализированных компаний.

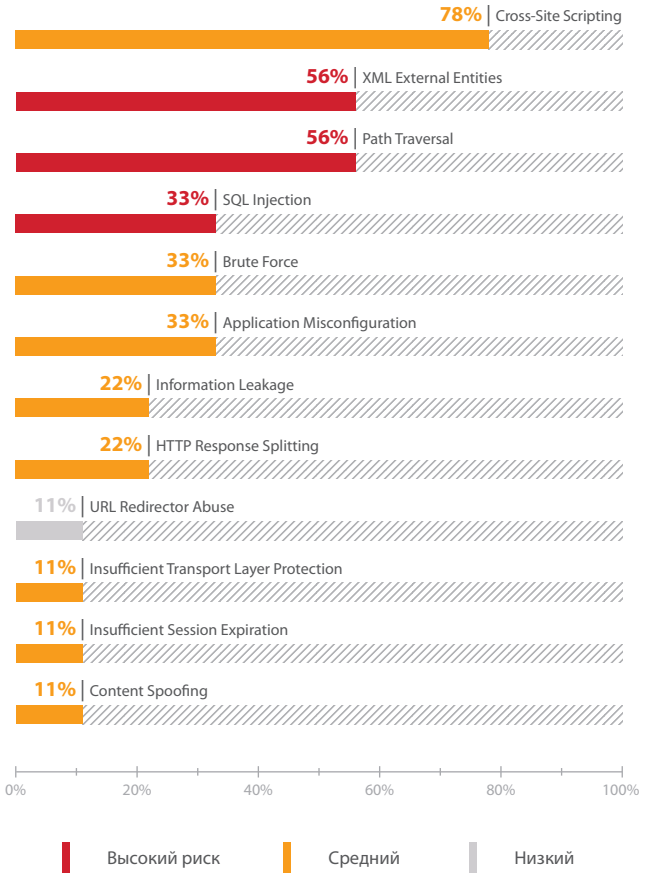

Рис. 27. Уязвимости, выявленные анализатором кода (доля систем)

4.6. Уязвимости тестовых и продуктивных веб-приложений

В данном разделе приведены результаты исследования уровня защищенности тестовых и продуктивных (находящихся в эксплуатации) систем. Распределение веб-приложений по максимальному уровню опасности уязвимостей, выявленных в тестовых и продуктивных системах представлено на рис. 28. Полученные результаты свидетельствуют о том, что системы, находящиеся на этапе разработки, более подвержены критически опасным уязвимостям.

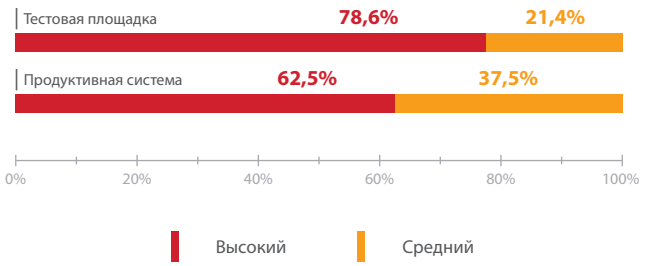

Рис. 28. Максимальный уровень риска обнаруженных уязвимостей (доли уязвимых систем)

В 62,5% продуктивных веб-ресурсов были обнаружены критически опасные уязвимости. Это подтверждает тот факт, что продуктивные веб-приложения содержат множество недостатков, которые могут быть использованы нарушителем с целью проведения атак. (Аналогичный вывод мы делали и в 2014 году.)

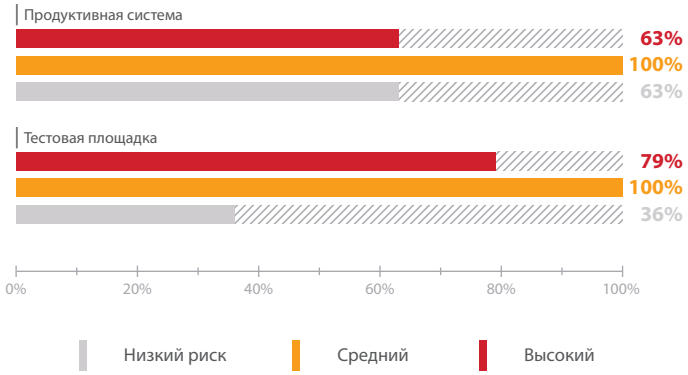

Рис. 29. Доли уязвимых сайтов

Для тестовых систем среднее количество уязвимостей высокой степени риска, приходящееся на одно приложение, почти в два раза выше по сравнению с продуктивными (10,6 против 5,4). Подобные результаты обусловлены множеством критически опасных недостатков, выявленных методом белого ящика. Важно отметить, что анализ исходных кодов выполнялся в основном в отношении тестовых систем.

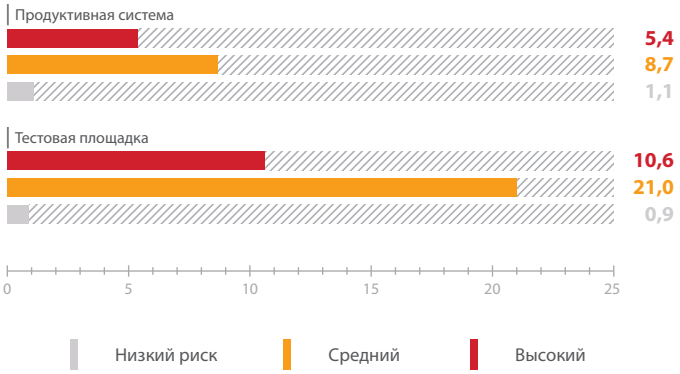

Рис. 30. Среднее количество уязвимостей на одно приложение

Полученные результаты подтверждают, что в продуктивных системах может присутствовать множество опасных уязвимостей, которыми стремится воспользоваться нарушитель. Уровень защищенности таких систем остается довольно низким.

Регулярный анализ защищенности необходимо проводить не только в отношении приложений, находящихся на этапе разработки, но и в отношении продуктивных систем. Кроме того, необходимо внедрять процессы обеспечения безопасности на всех стадиях жизненного цикла приложений (SSDL). Обеспечение безопасной разработки приложений позволило бы минимизировать количество уязвимостей, доступных атакующим на этапе эксплуатации системы. Для защиты продуктивных систем рекомендуется принимать превентивные меры, например использовать межсетевые экраны уровня приложений (web appliction firewalls).

Заключение

Проведенное исследование показало, что общий уровень защищенности веб-приложений продолжает снижаться. Разработчики стремятся обеспечить максимальную функциональность систем и не всегда уделяют должное внимание безопасности кода. При этом важно отметить широкую распространенность недостатков, связанных с ошибками администрирования. Их опасность, как правило, невысока, однако в случае эксплуатации некоторых таких уязвимостей нарушитель может не только получить чувствительную информацию (например, в результате ее разглашения на страницах приложений), но и получить несанкционированный доступ к системе (в случае подбора или перехвата учетных данных или успешной атаки на сессию).

Уязвимы все веб-приложения независимо от языка разработки и области применения. В каждой проанализированной системе были обнаружены уязвимости как минимум среднего уровня риска, а критически опасные недостатки — в 70%. Важно отметить, что ошибки в коде приложений позволяют не только получить полный контроль над сервером, но и проводить атаки на пользователей приложений, что может привести к существенным репутационным потерям.

Полученные данные свидетельствуют о необходимости регулярно проводить работы по анализу защищенности веб-приложений. Важно осуществлять такой анализ на всех стадиях разработки, а также регулярно (например, дважды в год) в процессе эксплуатации. Метод белого ящика с возможностью анализа исходных кодов существенно эффективнее при этом, чем другие методы тестирования. К тому же для проведения анализа можно применять автоматизированные средства (например, анализатор исходных кодов веб-приложений), которые обладают рядом преимуществ перед ручным тестированием (в частности — работают быстрее и исключают человеческий фактор).

Более того, приложения, уже находящихся в эксплуатации, в не меньшей степени нуждаются в эффективной защите от атак. Более половины таких ресурсов оказались подвержены критически опасным уязвимостям (62,5%). Эти недостатки могут привести не только к разглашению важных данных, но и к полной компрометации системы либо выводу ее из строя. Поэтому для защиты от атак рекомендуется применять межсетевые экраны уровня приложения.

Обеспечение всех этих мер в комплексе способно существенно повысить уровень защищенности веб-ресурсов и снизить риск компрометации систем до приемлемого уровня.

О компании

Positive Technologies — один из лидеров европейского рынка систем анализа защищенности и соответствия стандартам, а также защиты веб-приложений. Деятельность компании лицензирована Минобороны РФ, ФСБ и ФСТЭК, продукция сертифицирована ФСТЭК и «Газпромом». Организации во многих странах мира используют решения Positive Technologies для оценки уровня безопасности своих сетей и приложений, для выполнения требований регулирующих организаций и блокирования атак в режиме реального времени. Благодаря многолетним исследованиям специалисты Positive Technologies заслужили репутацию экспертов международного уровня в вопросах защиты SCADA- и ERP-систем, крупнейших банков и телеком-операторов.

Подробнее: facebook.com/PositiveTechnologies, facebook.com/PHDays, twitter.com/ptsecurity

POSITIVE TECHNOLOGIES

pt@ptsecurity.com ptsecurity.ru