# Перечень примерных вопросов

| | |
|---|---|
| Специальность | 10.02.05 Обеспечение информационной безопасности автоматизированных систем |
| Наименование МДК | МДК.01.04. Эксплуатация автоматизированных (информационных) систем в защищённом исполнении |
| Номер группы | БИ50-1-17, БИ50-2-17, БИ50-11-18 |
| Период | 1-й семестр 2019/2020 учебного года |
| Форма контроля | Экзамен |
| Ф. И. О. преподавателя | Караваев Сергей Владимирович |

Экзамен будет проведён в период с 23 по 28 декабря 2019 года включительно.

В каждом билете — 2 теоретических вопроса и 1 практическое задание.

## Теоретические вопросы

1. Безопасность автоматизированной информационной системы: определение по Р 50.1.053—2005, конфиденциальность, доступность, целостность, подотчётность и подлинность ресурсов.
2. Нормативно-правовые акты Российской Федерации в сфере информационной безопасности.
3. Электронный документооборот: определение, классификация систем автоматизации документооборота, угрозы безопасности.
4. Модель угроз безопасности и нарушителя безопасности АИС.
5. Простая аутентификация субъекта доступа по ГОСТ Р ИСО/МЭК 9594-8—98: определение, принципы, уязвимости. Привести описание подсистемы аутентификации, применяемой в *Laravel*.
6. Авторизация в автоматизированной (информационной) системе. Рассмотреть дискреционное (избирательное) управление доступом (*DAC*) на примере подсистемы политик (англ. *policies*) в *Laravel*.
7. *Hardening* сервера *HTTP* и транслятора серверного языка программирования.
8. *HTTP over TLS (HTTP/TLS)*: назначение, задействованные уровни стека протоколов Интернета, схема `https`.
9. *Secure cookie protocol*: определение, услуги. Привести пример реализации.
10. Контроль ввода информации в АИС: средства проверки в полях ввода в *HTML5*.
11. Контроль ввода информации в АИС: подсистема *Request* в *Laravel*.
12. Шаблонизация как фактор защиты веб-приложения. Принцип работы шаблонизаторов. Назвать один из наиболее популярных шаблонизаторов, привести пример его использования для защиты от *XSS*.
13. Безопасность паролей: оценка сложности, защита от атак на криптографические хеш-функции (в т. ч. по радужным таблицам), порядок сброса.
14. Исключительная ситуация: определение по ГОСТ 28397—89, порождение, обработка, регенерация. Справка: https://laravel.ru/docs/v5/errors.
15. Регистрация аварийных ситуаций в автоматизированной системе. Справка: https://laravel.com/docs/master/logging.
16. *Classic clickjacking*: векторы атаки, методика противодействия, примеры.
17. *Man-in-the-middle (MitM)*: векторы атаки, методика противодействия, примеры.

18. Атака по обходу каталогов (англ. *path traversal attack*): векторы атаки, методика противодействия. *SEO-friendly URL*: определение, способы реализации.

19. *SQL*-инъекции I порядка: векторы атаки через веб-приложение, методика противодействия, примеры.

20. *SQL*-инъекции II порядка: векторы атаки через веб-приложение, методика противодействия, примеры.

21. *Subresource spoofing*: векторы атаки, методика противодействия (в т. ч. посредством реализации стандарта *Subresource integrity*), примеры.

22. Отражённые (англ. *reflected*) *XSS*: векторы атаки, методика противодействия, примеры.

23. Хранимые (англ. *stored*) *XSS*: векторы атаки, методика противодействия, примеры.

24. Подделка межсайтового запроса (*CSRF*): векторы атаки, методика противодействия, примеры.

25. *Denial of service (DoS)*: векторы атаки, методика противодействия, примеры.

## Примерные практические задания

1. Продемонстрируйте работу подсистемы политик в *Laravel*.

2. Реализуйте и введите в эксплуатацию модуль, например класс-наследник *Request* в *Laravel*, обеспечивающей контроль ввода (задать ограничения на длину определённых строк, на значение определённых чисел; делать определённые атрибуты обязательными).

3. Реализуйте и введите в эксплуатацию шаблон (совместимый, например, с шаблонизатором *Blade*) для вывода определённых данных из базы данных в формате *HTML*. Обеспечьте кодирование данных для защиты от хранимых *XSS*.

4. Реализуйте защиту от атак по обходу каталогов (англ. *path traversal attack*).

5. Реализуйте подсистемы регистрации и аутентификации субъектов доступа.

6. Выполняется определённый запрос. Проведите *SQL*-инъекцию, которая приведёт к определённому результату. Исправьте указанный фрагмент исходного кода приложения базы данных таким образом, чтобы уязвимость для *SQL*-инъекций была устранена.

7. Реализуйте и введите в эксплуатацию шаблон формы *HTML* (совместимый, например, с шаблонизатором *Blade*) для определённых действий над определённой таблицей базы данных. Обеспечьте защиту от *CSRF* с помощью *token*’ов и от *SQL*-инъекций — с помощью подготовленных предложений *SQL*.

8. Реализуйте защиту от *subresource spoofing* и *classic clickjacking*.

9. С применением *Wireshark* продемонстрируйте работу определённого протокола шифрования на произвольном веб-приложении.

10. Настройте обработчик исключительных ситуаций, порождаемых веб-сайтом. Справка: https://laravel.com/docs/master/errors, https://laravel.com/docs/master/logging.