**Анализ вредоносного программного обеспечения**

По материалам Чио К., Фримэн Д. Машинное обучение и безопасность / trans. Снастина О.В. М.: ДМК  Пресс, 2020. 388 с. Глава 7

1. **Титульный слайд (название модуля, автор)**

Приветствие. Занятие «Анализ вредоносного программного обеспечения».

1. **Программа модуля**

Кратко – про что будет доклад. На практическом занятии: анализ android-приложения

1. **Анализ вредоносного программного обеспечения**

Анализ вредоносного программного обеспечения (malware analysis) — это область исследования функциональных возможностей. целей, происхождения и потенциального воздействия вредоносного ПО.

1. **Вредоносное программное обеспечение**

Любое программное обеспечение, предназначенное для несанкционированного доступа к вычислительным ресурсам ЭВМ или к информации, хранящейся на ЭВМ, с целью несанкционированного использования ресурсов ЭВМ, причинение вреда владельцу ЭВМ сети ЭВМ или информации путём копирования, искажения, удаления или подмены информации.

Существует множество различных типов вредоносного ПО и также, как различные типы обычного ПО, вредоносное ПО может быть написано на любом языке программирования. с учётом различий целевых сред выполнения и требований к условиям выполнения.

В упакованном и развёрнутом состоянии большинство вредоносных программ существуют в виде бинарных файлов

1. **Обнаружение вредоносного ПО**

При наличии доступа к коду высокого уровня, относительно просто можно определить, что делает программа, однако большинство вредоносных программ распространяется в бинарном виде, перехватываются и накапливаются “в полевых условиях”:

* отлавливаются в ловушках-песочницах,
* продаются на нелегальных форумах,
* обнаруживаются на компьютерах жертв.

Чаще всего человек не может прочитать бинарные файлы, так как они предназначены для непосредственного выполнения компьютером., следовательно профилирование характеристик и поведения вредоносной программы становится процессом реверс-инжиниринга., цель которого — понять. что делает эта программа, как если бы имелся исходный код на языке высокого уровня. Без знания контекста интерпретации, стандарта кодирования и алгоритма декодирования сами по себе бинарные данные фактически бессмысленны. Однако система машинного обучения хороша настолько, насколько качественными являются её входные данные. Исходные бинарные данные требуют составления плана сбора данных, их очистки и валидации перед отправкой в модель машинного обучения. Предварительная обработка таких исходных данных важна для выбора их оптимального формата для передачи в модели машинного обучения.

1. **Пути распространения вредоносного ПО**

Вредоносное ПО может быть встроено в бинарные файлы разнообразных форматов, работа которых совершенно отличается друг от друга. Например, PE--файлы в ОС Windows, ELF­-файлы в Unix­-системах (Executable and Linkable Format) и APK-­файлы в ОС Android (формат Android Package Kit с расширением .apk и др.) имеют совершенно различную внутреннюю структуру файлов и требуют разного контекста выполнения.

Широко распространены вредоносные компоненты, внедряющиеся в файлы документов, например с расширениями .doc, .pdf и .rtf, и использующие макросы и динамически выполняемые элементы в структуре документа, чтобы выполнить вредительские действия.

Вредоносное ПО также может быть представлено в форме расширений и динамически подключаемых компонентов (plug­ins) для распространенных программных платформ, таких как веб-браузеры и комплексные рабочие веб­среды.

1. **Классификация вредоносного программного обеспечения**

Группы классификации вредоносного ПО объединяют отдельные экземпляры на основе общих свойств. Можно классифицировать вредоносное ПО разными способами в зависимости от конкретной задачи и поставленной цели.

Например, классификация вредоносного ПО по степени его опасности и функциональности будет полезна группе обеспечения безопасности для определения степени угрозы.

Для обобщённого анализа вредоносного ПО распространённой промышленной методикой является группировка экземпляров по семействам (family).

1. **Семейства вредоносного ПО (Malware families)**

Семейства позволяют прослеживать авторство, коррелировать информацию и идентифицировать новые варианты обнаруженных вредоносных программ. Экземпляры вредоносного ПО из одного семейства могут иметь одинаковый код, возможности, авторство, функциональные характеристики, цели и/или исходные предпосылки.

Различия между экземплярами вредоносного ПО из одного семейства могут определяться по разным компиляторам или секциям исходного кода, добавляемым и/или удаляемым для изменения функциональности вредоносной программы. Экземпляры вредоносных программ, которые со временем эволюционируют в ответ на изменения стратегий их выявления и нейтрализации, зачастую также демонстрируют сходство между старой и новой версиями,

1. **Пример семейства вредоносного ПО — Conficker**

Червь, предназначенный для внедрения в ОС Microsoft Widows. Существует множество вариаций червя Conficker с различным кодом, авторами и поведением, однако определённые характеристики этих червей позволяют причислить их к одному семейству, обозначая их общее происхождение: все черви Conficker

* используют уязвимости Windows,
* предпринимают перебор по словарю для подбора пароля к учётной записи администратора.
* Устанавливают скрытое ПО для использования хоста в деятельности ботнета.

1. **Место безопасных файлов в классификации вредоносного ПО**

В классификации вредоносного ПО также может быть класс бинарных файлов, не являющихся вредоносными, так как основная цель антивирусного ПО — определение вероятности того, что файлу можно доверять и выполнять его в защищаемой среде. Обычно решение такой задачи основано на методе сравнения сигнатур.

1. **Метод сигнатур**

Метод статически сигнатур позволяет, имея достаточный набор свойств и образцов поведения ранее обнаруженного и исследованного вредоносного ПО, сравнивать новые появляющиеся в системе бинарные файлы с этим набором данных, чтобы определить, нет ли совпадений с каким-­либо признаками вредоносного ПО, выявленными ранее.

1. **Метод сигнатур. Полиморфные вредоносное ПО**

Метаморфные или полиморфные вирусы и черви используют статические и динамические методики маскировки для изменения характеристик своего кода, поведения и свойств, применяемые для алгоритмов генерации сигнатур в механизмах идентификации вредоносного ПО. Такой подход в настоящее время становится все более распространенным из-за успешного создания помех и препятствий для механизмов синтаксических сигнатур вредоносного ПО.

Механизмы статических сигнатур продолжают погоню за постоянно сужающимся набором статических сигналов, которым авторы вредоносного ПО пренебрегают или не могут изменить коренным образом.

1. **Методы машинного обучения в классификации вредоносного ПО**

Методы машинного обучения могут помочь устранить проблемы, возникающие при использовании метода сигнатур, благодаря следующим свойствам:

* Нечёткое сравнение — алгоритмы машинного обучения могут определять сходство между двумя и более объектами с помощью вещественной метрики расстояния. Экземпляры данных, отображаемые в векторное пространство признаков, могут быть сгруппированы на основании относительны расстояний меду ними. Такая возможность определения приблизительных совпадений между объектами полезна при классификации вредоносного ПО. различия в свойствах которого приводят в замешательство методы статического сравнения сигнатур.
* Автоматизированный выбор свойств — автоматическое определение веса (значимости) признака и выбор самых значимых признаков являются главными аспектами машинного обучения, которые помогают осуществить классификацию вредоносного ПО. На основе статистических свойств тренировочного набора данных можно ранжировать признаки по их относительной важности при определении отличия экземпляра, принадлежащего классу А, от другого экземпляра, принадлежащего классу В. Кроме того, появляется возможность объединения в группу двух экземпляров класса А. Некоторые алгоритмы снижения размерности и выбора признаков способны обнаруживать скрытые свойства экземпляров, которые чрезвычайно трудно выявить даже эксперту в этой области. Методы машинного обучения освобождают аналитиков вредоносного ПО от некоторых трудоемких операций по определению значимости каждого признака. Позволяя самим данным выявлять и определять набор признаков для использования в схеме классификации.
* Адаптируемость: вечное противостояние между создателями вредоносного ПО и защитниками систем приводит к непрерывному изменению типов и шаблонов предпринимаемых атак. Как и при разработке обычного ПО, вредоносное ПО развивается и улучшается со временем, поскольку его авторы добавляют функциональные возможности и исправляют ошибки.

Кроме того, как было отмечено выше, авторы вредоносного ПО всегда стремятся к непрерывным усовершенствованиям, изменяя поведение своих программ, чтобы избежать обнаружения. С помощью методики нечеткого сравнения и управляемого данными процесса выбора признаков системы классификации вредоносного ПО, реализованные на основе машинного обучения, способны адаптироваться к изменениям исходных условий (входных данных) и отслеживать развитие вредоносного ПО во времени.

Машинное обучение может помочь существенно снизить объем ручной работы и уровень экспертных знаний, требуемый для классификации вредоносного ПО.

1. **Современные процессы выполнения кода**

Существуют 2 основных типа выполнения кола: выполнение компилируемого кода и выполнение интерпретируемого кода.

(схема на слайде с рис.4.1 стр 155)

1. **Типовой процесс атаки вредоносного программного обеспечения**

При исследовании и классификации вредоносного ПО важно понимать, что именно делает вредоносная программа и как возникла уязвимость. Разные типы вредоносного ПО используют различные методы распространения, преследуют разнообразные цели и создают разные уровни угрозы. Однако существует типовой процесс атаки вредоносного ПО.

(схема на слайде с рис.4.2 стр 165)

1. **Типы поведения вредоносного ПО**

Вредоносное ПО часто демонстрирует определённые типы поведения:

* Маскировка своего присутствия — вредоносное ПО использует методики сжатия и шифрования для придания своему коду более компактной и замаскированной формы.
* Стремление к выполнению своей функции. Вредоносная программа должна обеспечить достаточную степень живучести, чтобы не быть уничтоженной изменениями в системе или при обнаружении системным администратором. Возможно принудительное завершение антивирусных программ.
* Сбор данных и оповещение — после выполнение своей функции вредоносная программа отправляет собранные данные на внешний пункт сбора, либо отправляет оповещение на удалённый сервер для получения дальнейших инструкций.

1. **Генерация признаков. Сбор данных**

Если просто позволить приложению принимать бесконечный поток информации из интернета, то вряд ли удастся достичь приемлемого качества данных для машинного обучения. В итоге будет собрана масса ненужных данных вместе с теми данными, которые действительно необходимы, но и они могут быть искажены или неточны.

Специалисты по исследованию данных используют следующие положения для улучшения процесса сбора данных:

* Важность знаний предметной области — глубокое экспертное исследование предметной области может помочь быстро оценить важные признаки, которые нужно собрать.
* Масштабируемые процессы сбора данных —для получения действительно полезных результатов часто необходимо предоставить алгоритму машинного обучения огромные объёмы данных.
* Валидация данных
* Итеративные эксперименты

1. **Генерация признаков**

Для генерации признаков применяются следующие методики:

* Статические методы
  + анализ структуры
  + статический анализ
* Динамические методы
  + анализ поведения
  + отладка
  + динамические контрольные измерения

Более подробно методики генерации признаков будут рассмотрены в лабораторной работе.

1. **Выбор признаков (Feature selection)**

В большинстве случаев бездумная загрузка огромного количества признаков в алгоритм машинного обучения создают ненужный шум и пагубно влияют на точность и эффективность модели. Поэтому важно выбирать только самые важные и значимые признаки.

Один из широко распространённых способов выбора признаков — использование человеческого опыта. Люди-эксперты могут обеспечить процесс руководства моделями машинного обучения, который проявляется главным образом в форме добытых вручную признаков, считающихся наиболее важными элементами информации, используемыми в процессе обучения человека.

Статистически управляемые алгоритмы выбора признаков — методы снижения размерности наборов данных:

* Одномерный анализ (univariate analysis) — модели поочерёдно подаётся на вход по одному признаку. С помощью итеративно выполняемых одномерных статистических тестов по каждому отдельном признаку можно вывести относительную оценку того, насколько хорошо каждый признак соответствует распределению меток в тренировочном наборе.
* Рекурсивное исключение признаков (recursive feature elimination): действуя с противоположного направления, такие методы начинают с обработки полного набора признаков и рекурсивно рассматривают постоянно уменьшающиеся подмножества признаков с анализом того, как исключение признаков влияет на точность тренировки модели оценки, предложенной исследователем;
* неявное представление признаков (latent feature representations): такие методы, как сингулярное разложение (Singular Value Decomposition – SVD) и метод главных компонентов (Principal Component Analysis – PCA), выполняют преобразование данных с высокой размерностью в пространства данных с более низкой размерностью. Эти алгоритмы предназначены для минимизации потерь информации при сокращении количества признаков, необходимых для эффективной работы моделей машинного обучения.
* классификация признаков в зависимости от конкретной модели (model­ specific feature ranking) — когда важность признака пропорциональна весам обученной модели, соответствующим этому признаку,

1. **Обучение без учителя и глубокое обучение**

Существует класс алгоритмов глубоких нейронных сетей, способных обучаться без учителя, например автокодировики

1. **Полезные ссылки**

Выводы по лекции.

Спасибо за внимание.