Буйтвидас А.В. Валидация структурированного контента и её роль в защите ВЕБ-приложений // Международный журнал информационных технологий и энергоэффективности. – 2025. – Т. 10 № 1(51) с. 112–116

Международный журнал информационных технологий и энергоэффективности

Сайт журнала: http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.736

ВАЛИДАЦИЯ СТРУКТУРИРОВАННОГО КОНТЕНТА И ЕЁ РОЛЬ В ЗАЩИТЕ ВЕБ-ПРИЛОЖЕНИЙ

Буйтвидас А.В.
ФГАОУ ВО «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА», Москва, Россия, (127055, город Москва, ул. Образцова, д.9 стр.9), e-mail: buytik13@gmail.com

В статье определены наиболее распространённые типы структурированных данных, использующиеся в веб-приложениях. Описаны примеры уязвимостей, которые можно эксплуатировать, опираясь на некорректную структуру передаваемых данных. Сформулированы правила составления схем валидации наиболее популярных структур данных XML и JSON, которые позволят защитить приложение от атак, использующих подобные уязвимости.

Ключевые слова**:** JSON. XML, валидация данных, кибербезопасность, защита информации.

VALIDATION OF STRUCTURED CONTENT AND IT'S ROLE IN WEB APPLICATION PROTECTION

Buitvydas A.V.
RUSSIAN UNIVERSITY OF TRANSPORT, Moscow, Russia, (127055, Moscow, Obraztsova str., 9, bldg. 9), e-mail: buytik13@gmail.com

The article identifies the most common types of structured data used in web applications. Examples of vulnerabilities that can be exploited based on the incorrect structure of the transmitted. The rules compiling validation schemes for the most popular XML and JSON data structures are formulated, which will protect the application from attacks using such vulnerabilities.

Keywords: JSON, XML, data validation, cybersecurity, data protection.

Согласно статистике, собранной посредством сканирования общедоступных веб-приложений работающих на REST архитектуре, на октябрь 2024 года наибольшее распространение среди типов данных, используемых в приложениях, получили такие форматы, как binary, pdf, xml, json.

Таблица 1. - Статистика используемых медиа-типов в приложениях [1]

Разведанный mimetype	**Август 2024, %**	**Сентябрь 2024, %**	**Октябрь 2024, %**
application/pdf	0.6007	1.0165	0.7784
application/atom+xml	0.1005	0.1106	0.0989
application/xml	0.0238	0.0265	0.0542
application/rss+xml	0.0497	0.0577	0.0525
application/octet-stream	0.0443	0.0544	0.0496
application/json	0.0259	0.0306	0.0321

Так как pdf [2] и тем более бинарные файлы(application/octet-stream) не являются структурированным контентом, рассмотрим более подробно валидацию популярных типов данных JSON[3] и XML[4].

Хотя спецификации XML и схем XML(XSD)[5] уже предоставляют инструменты, необходимые для защиты приложений использующих XML, они также содержат множество недостатков безопасности. Их можно использовать для выполнения нескольких типов атак, включая извлечение файлов, подделку запросов к серверу, сканирование портов и прочее. В этой статье рассмотрим, как злоумышленники могут эксплуатировать уязвимости приложений, основанных на XML в библиотеках и программном обеспечении, которые можно разделить на два направления атаки:

1. Некорректно сформированные XML-документы, использование уязвимостей, когда приложения обрабатывают XML-документы, которые сформированы не по спецификации.
2. Невалидные XML документы, использование уязвимостей, связанных с отправкой документов соответствующих спецификации, но с неверной структурой данных.

Обработка содержимого, не соответствующего спецификации, может привести к чрезмерной утилизации процессора. Если документ XML имеет неверный формат, анализатор XML обнаружит фатальную ошибку, он должен прекратить выполнение, документ не должен подвергаться какой-либо дополнительной обработке, а приложение должно отобразить сообщение об ошибке.

Одна из таких популярных атак - это отправка документа с большой вложенностью без соответствующих закрывающих тэгов:

Пример:

```
<S1>
 <S2>
  <S3>
   ...
    <S10000>
```

Таким образом, даже если проверяется размер передаваемого сообщения, такая атака позволит передать как минимум вдвое больше тегов чем предполагается. Обработка одного такого документа займёт много процессорного времени.

Такая атака при отсутствии правильно настроенной схемы валидации приведёт к отказу сервиса (DOS), без необходимости со стороны злоумышленника задействовать большие вычислительные ресурсы для атаки на приложение.

Если отсутствует строгий контроль состава структуры данных, то существует много вариантов атак, которыми может воспользоваться злоумышленник.

Рассмотрим на примере интернет магазина.

Пусть некоторый магазин продаёт электронные книги и валидная структура на покупку выглядит так:

```
<purchase>
 <id>1</id>
 <price>100</price>
<quantity>1</quantity>
</purchase>
```

Где, id – это номер/артикул товара, price – цена товара, quantity – количество товара

Злоумышленник может отправить, например, отрицательное количество товара или дублирующиеся теги в результате чего получится книгу бесплатно в случае если приложение обрабатывает только первый встретившийся id:

```
<purchase>
 <id>1</id><price>0</price><quantity>1</quantity><id></id>
 <price>100</price>
 <quantity>1</quantity>
</purchase>
```

Для того чтобы предотвратить такие проблемы, опишем схему валидации данных чтобы исключить возможность обработки негативных значений и дублирования тегов:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="purchase">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="id" type="xs: positiveInteger"/>
    <xs:element name="price" type="xs:decimal"/>
    <xs:element name="quantity" type="xs:positiveInteger"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

Чтобы корректно обрабатывать такие исключения XSD схема и обработчик должны в точности следовать спецификации, принимать только корректно сформированные XML, следовать чёткой валидации:

1. Запрет передачи не декларированных в схеме тегов(отсутствуют параметры xs:any, xs:anyType, xs:anySimpleType)
2. Должны быть ограничения по длине для всех строковых тегов
3. Для числовых значений должны быть указаны ограничения по минимальным и максимальным значениям.

4. Отсутствие циклических ссылок, т.е. если у тега “А” определено свойство ref со ссылкой на некоторый тег “Б”, то необходимо убедиться что у этого тега “Б” нет свойства ref со ссылкой на тег “А”, должно выполняться для любого уровня вложенности.

Эксплуатация уязвимостей приложений, использующих структуры JSON во многом схожи с XML и делятся на некорректно сформированные JSON документы и невалидные JSON документы, поэтому далее описываются только правила защиты от подобных уязвимостей.

Изначально в спецификации JSON отсутствовало такое понятие как JSON Schema[6], однако в связи с необходимостью валидации данных по аналогии с XML был разработан этот стандарт. Для JSON схем требования к схемам частично схожи с XML, однако есть и отличия:

1. Должны быть ограничения по длине для всех строковых тегов
2. Для строковых тегов, по возможности, должны быть определены patternProperties
3. Для числовых значений должны быть указаны ограничения по минимальным и максимальным значениям.
4. Отсутствие циклических ссылок, т.е. если у тега “А” определено свойство ref со ссылкой на некоторый тег “Б”, то необходимо убедиться что у этого тега “Б” нет свойства ref со ссылкой на тег “А”, должно выполняться для любого уровня вложенности.
5. У всех тегов с типом object должно быть определено свойство запрета не описанных в схеме полей additionalProperties: false
6. У тегов с типом массив, должно быть должно быть определено свойство maxItems – максимальное число элементов массива.

Заключение

В статье определены наиболее распространённые типы структурированных данных, использующиеся в веб-приложениях. Описаны примеры уязвимостей, которые можно эксплуатировать, опираясь на некорректную структуру передаваемых данных. Сформулированы правила составления схем валидации наиболее популярных структур данных XML и JSON, которые позволят защитить приложение от атак, использующих подобные уязвимости. Валидация данных позволяет повысить надёжность и безопасность приложений.

Список литературы

1. Статистика ежемечного сканирования по медиатипам [Электронный ресурс] https://commoncrawl.github.io/cc-crawl-statistics/plots/mimetypes (дата обращения 24.11.2024)
2. RFC для PDF формата [Электронный ресурс] https://datatracker.ietf.org/doc/html/rfc7995 (дата обращения 24.11.2024)
3. The application/json медиатип JavaScript Object Notation (JSON) [Электронный ресурс] https://datatracker.ietf.org/doc/html/rfc4627.html (дата обращения 24.11.2024)
4. Медиатип XML [Электронный ресурс] https://www.rfc-editor.org/rfc/rfc7303 (дата обращения 24.11.2024)
5. [Электронный ресурс] Спецификация XSD https://www.w3.org/TR/xmlschema11-1/ (дата обращения 24.11.2024)

6. Спецификация JSON Schema [Электронный ресурс] https://json-schema.org/ (дата обращения 24.11.2024)

References

1. Statistics of Monthly Scanning by Media Types [Electronic resource] https://commoncrawl.github.io/cc-crawl-statistics/plots/mimetypes (accessed 24.11.2024)
2. RFC for PDF format [Electronic resource] https://datatracker.ietf.org/doc/html/rfc7995 (accessed 24.11.2024)
3. The application/json media type JavaScript Object Notation (JSON) [Electronic resource] https://datatracker.ietf.org/doc/html/rfc4627.html (accessed 24.11.2024)
4. Media type XML [Electronic resource] https://www.rfc-editor.org/rfc/rfc7303 (accessed 24.11.2024)
5. Specification of XSD https://www.w3.org/TR/xmlschema11-1/ (accessed 24.11.2024)