FB.DMI\_BLOCKING\_METHODS\_ON\_URL

Детектор определяет ошибку, в ходе которой методы URL-адреса Equals и hashCode выполняют разрешение доменных имен, что может привести к значительному снижению производительности.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Выполнение методов equals() и hashCode() для объектов URL, которые в свою очередь вызывают разрешение доменных имен, может существенно замедлить работу программы, особенно при работе с большим количеством URL-адресов. |
| Комментарий | Рекомендуется избегать использования equals() и hashCode() на объектах URL без предварительной обработки или кэширования результатов разрешения доменных имен. Вместо этого можно использовать заранее разрешённые IP-адреса или кэшировать доменные имена, чтобы избежать ненужных сетевых запросов. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  try {  URL url1 = new URL("<http://example.com>");  URL url2 = new URL("<http://example.com>");  // метод может блокировать выполнение  System.out.println(url1.equals(url2));  // hashCode может также выполнять сетевые операции  System.out.println(url1.hashCode());  } catch (Exception e) {  throw new RuntimeException(e);  }  } | public static void correctTest() {  try {  URL url1 = new URL("<http://example.com>");  URL url2 = new URL("<http://example.com>");  System.out.println(customEquals(url1, url2));  System.out.println(customHashCode(url1));  } catch (Exception e) {  throw new RuntimeException(e);  }  }  // Собственная реализация equals, которая не вызывает сетевые операции  public static boolean customEquals(URL url1, URL url2) {  return url1.getProtocol().equals(url2.getProtocol()) &&  url1.getHost().equals(url2.getHost()) &&  url1.getPort() == url2.getPort();  }  // Собственная реализация hashCode, которая не вызывает сетевые операции  public static int customHashCode(URL url) {  return Objects.hash(url.getProtocol(), url.getHost(), url.getPort());  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Используйте IP-адреса:** Если возможно, используйте непосредственно IP-адреса вместо доменных имен в URL-адресах. Это исключит необходимость разрешения DNS.
2. **Кэшируйте результаты разрешения DNS:** Если вам необходимо работать с доменными именами, рассмотрите возможность предварительного разрешения доменных имен и хранения IP-адресов. Затем используйте эти IP-адреса для сравнения и вычисления хеш-кода.
3. **Реализуйте собственные методы equals() и hashCode():** Создайте класс-обертку для URL или реализуйте собственную логику сравнения и вычисления хеш-кода, которая не вызывает разрешение доменных имен. Сравнивайте протокол, хост (без разрешения), порт и путь. Пример такой реализации приведен в корректном коде выше.
4. **Избегайте использования URL в коллекциях, где часто вызываются equals() и hashCode():** Если вам нужно хранить URL-адреса в HashSet, HashMap или других коллекциях, которые интенсивно используют equals() и hashCode(), используйте класс-обертку с собственной реализацией этих методов.
5. **Будьте осторожны при сравнении URL-адресов:** Если вам необходимо сравнить URL-адреса, осознавайте потенциальные блокировки из-за разрешения DNS и принимайте соответствующие меры.