Fortify Standalone Report Generator

Developer Workbook

cwe327

Table of Contents

Executive Summary
Project Description
Issue Breakdown by Fortify Categories
Results Outline

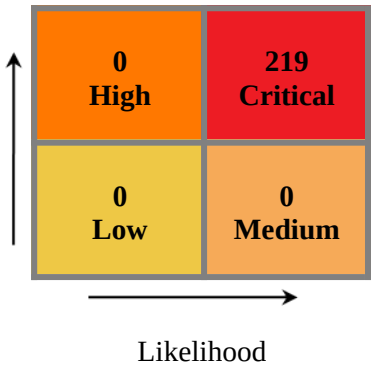
Executive Summary

This workbook is intended to provide all necessary details and information for a developer to understand and remediate the different issues discovered during the cwe327 project audit. The information contained in this workbook is targeted at project managers and developers.

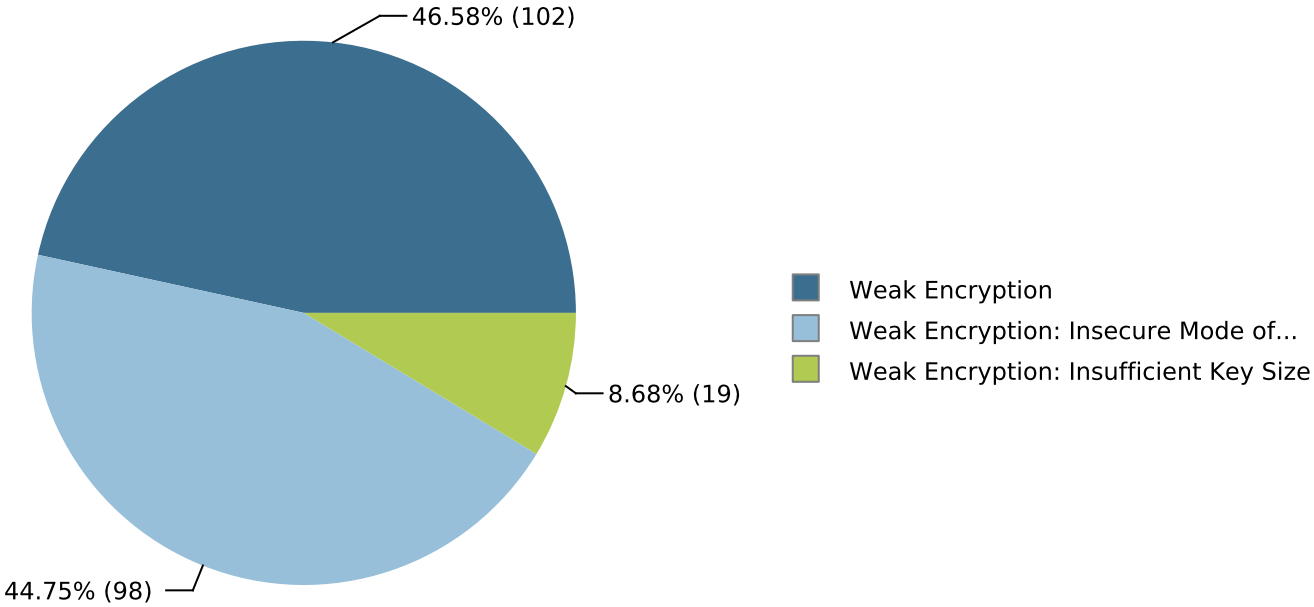
This section provides an overview of the issues uncovered during analysis.

Project Name:	cwe327
Project Version:	
SCA:	Results Present
WebInspect:	Results Not Present
WebInspect Agent:	Results Not Present
Other:	Results Not Present

Issues by Priority



Top Ten Critical Categories



Project Description

This section provides an overview of the Fortify scan engines used for this project, as well as the project meta-information.

SCA

Date of Last Analysis:	Aug 5, 2021, 8:51 AM	**Engine Version:**	21.1.1.0009
Host Name:	ip-10-138-53-201	**Certification:**	VALID
Number of Files:	46	**Lines of Code:**	2,446

Rulepack Name	Rulepack Version
Fortify Secure Coding Rules, Core, Annotations	2020.4.0.0007
Fortify Secure Coding Rules, Extended, Content	2020.4.0.0007
Fortify Secure Coding Rules, Extended, Java	2020.4.0.0007
Fortify Secure Coding Rules, Extended, JSP	2020.4.0.0007
Fortify Secure Coding Rules, Extended, Configuration	2020.4.0.0007
Fortify Secure Coding Rules, Core, Java	2020.4.0.0007
Fortify Secure Coding Rules, Core, Android	2020.4.0.0007

Issue Breakdown by Fortify Categories

The following table depicts a summary of all issues grouped vertically by Fortify Category. For each category, the total number of issues is shown by Fortify Priority Order, including information about the number of audited issues.

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Weak Encryption	0 / 102	0	0	0	0 / 102
Weak Encryption: Insecure Mode of Operation	0 / 98	0	0	0	0 / 98
Weak Encryption: Insufficient Key Size	0 / 19	0	0	0	0 / 19

Results Outline

Weak Encryption (102 issues)

Abstract

The identified call uses a weak encryption algorithm that cannot guarantee the confidentiality of sensitive data.

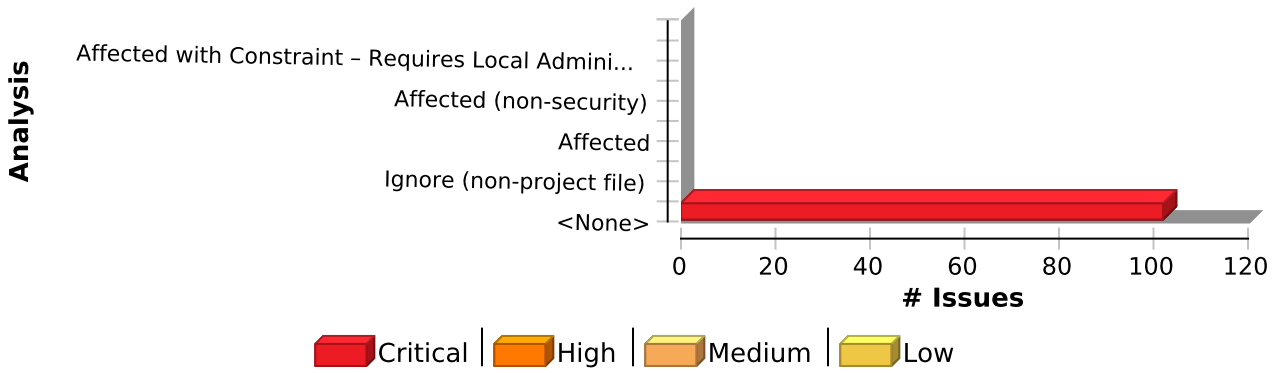
Explanation

Antiquated encryption algorithms such as DES no longer provide sufficient protection for use with sensitive data. Encryption algorithms rely on key size as one of the primary mechanisms to ensure cryptographic strength. Cryptographic strength is often measured by the time and computational power needed to generate a valid key. Advances in computing power have made it possible to obtain small encryption keys in a reasonable amount of time. For example, the 56-bit key used in DES posed a significant computational hurdle in the 1970s when the algorithm was first developed, but today DES can be cracked in less than a day using commonly available equipment.

Recommendation

Use strong encryption algorithms with large key sizes to protect sensitive data. A strong alternative to DES is AES (Advanced Encryption Standard, formerly Rijndael). Before selecting an algorithm, first determine if your organization has standardized on a specific algorithm and implementation.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Weak Encryption	102	0	0	102
Total	**102**	**0**	**0**	**102**

Weak Encryption	Critical
Package: testcases.CWE327_Use_Broken_Crypto	
testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 38 (Weak Encryption)	**Critical**

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 38 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:38
Taint Flags:

```
35 if (PRIVATE_STATIC_FINAL_FIVE == 5)
36 {
37 final String CIPHER_INPUT = "ABCDEFG123456";
38 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
39 /* Perform initialization of KeyGenerator */
40 keyGenerator.init(112);
41 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 38 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 45 (Weak Encryption) — Critical

```
43  /* FLAW: Use a weak crypto algorithm, 3DES */
44  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
45  Cipher tripleDesCipher = Cipher.getInstance("DESede");
46  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java:38
Taint Flags:

```
35  SecretKey secretKey = keyGenerator.generateKey();
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 46 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:46
Taint Flags:

```
43  byte[] byteKey = secretKey.getEncoded();
44  /* FLAW: Use a weak crypto algorithm, 3DES */
45  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
46  Cipher tripleDesCipher = Cipher.getInstance("DESede");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 46 (Weak Encryption) — Critical

```
47 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:32
Taint Flags:

```
29 if (IO.staticReturnsTrueOrFalse())
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:32
Taint Flags:

```
29 if (IO.staticReturnsTrueOrFalse())
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java:38
Taint Flags:

```
35 if (PRIVATE_STATIC_FINAL_FIVE == 5)
36 {
37 final String CIPHER_INPUT = "ABCDEFG123456";
38 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
39 /* Perform initialization of KeyGenerator */
40 keyGenerator.init(56);
41 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:32
Taint Flags:

```
29 if (true)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 32 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java:32
Taint Flags:

```
29 if (5 == 5)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 38 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
45 Cipher desCipher = Cipher.getInstance("DES");
46 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java:32
Taint Flags:

```
29 for(int j = 0; j < 1; j++)
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 32 (Weak Encryption) — Critical

```
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:32
Taint Flags:

```
29 if (IO.STATIC_FINAL_FIVE == 5)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 39 (Weak Encryption) — Critical

```
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:32
Taint Flags:

```
29 if (5 == 5)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java:32
Taint Flags:

```
29
30 final String CIPHER_INPUT = "ABCDEFG123456";
31
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33
34 /* Perform initialization of KeyGenerator */
35 keyGenerator.init(112);
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 32 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:32
Taint Flags:

```
29 if (IO.staticFive == 5)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java, line 32 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java:32
Taint Flags:

```
29 while(true)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java:32
Taint Flags:

```
29 if (IO.staticFive == 5)
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 32 (Weak Encryption) — Critical

```
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 43 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java:43
Taint Flags:

```
40 /* FLAW: Use a weak crypto algorithm, 3DES */
41 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
42
43 Cipher tripleDesCipher = Cipher.getInstance("DESede");
44 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
45
46 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:32
Taint Flags:

```
29 if (IO.staticReturnsTrue())
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 32 (Weak Encryption) — Critical

```
33  /* Perform initialization of KeyGenerator */
34  keyGenerator.init(112);
35  SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java:45
Taint Flags:

```
42  SecretKey secretKey = keyGenerator.generateKey();
43  byte[] byteKey = secretKey.getEncoded();
44  /* FLAW: Use a weak crypto algorithm, DES */
45  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
46  Cipher desCipher = Cipher.getInstance("DES");
47  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java:39
Taint Flags:

```
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 40 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:40
Taint Flags:

```
37 byte[] byteKey = secretKey.getEncoded();
38 /* FLAW: Use a weak crypto algorithm, 3DES */
39 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
40 Cipher tripleDesCipher = Cipher.getInstance("DESede");
41 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
42 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
43 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 44 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:44
Taint Flags:

```
41 SecretKey secretKey = keyGenerator.generateKey();
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, 3DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
45 Cipher tripleDesCipher = Cipher.getInstance("DESede");
46 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 39 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 52 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 52 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:52
Taint Flags:

```
49 SecretKey secretKey = keyGenerator.generateKey();
50 byte[] byteKey = secretKey.getEncoded();
51 /* FLAW: Use a weak crypto algorithm, 3DES */
52 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
53 Cipher tripleDesCipher = Cipher.getInstance("DESede");
54 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
55 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:39
Taint Flags:

```
36 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 39 (Weak Encryption) — Critical

```
37 byte[] byteKey = secretKey.getEncoded();
38 /* FLAW: Use a weak crypto algorithm, 3DES */
39 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
40 Cipher tripleDesCipher = Cipher.getInstance("DESede");
41 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
42 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 52 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java:52
Taint Flags:

```
49 SecretKey secretKey = keyGenerator.generateKey();
50 byte[] byteKey = secretKey.getEncoded();
51 /* FLAW: Use a weak crypto algorithm, DES */
52 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
53 Cipher desCipher = Cipher.getInstance("DES");
54 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
55 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 46 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, 3DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
46 Cipher tripleDesCipher = Cipher.getInstance("DESede");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 46 (Weak Encryption) — Critical

```
47 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 33 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:33
Taint Flags:

```
30 {
31 case 7:
32 final String CIPHER_INPUT = "ABCDEFG123456";
33 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
34 /* Perform initialization of KeyGenerator */
35 keyGenerator.init(112);
36 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java:32
Taint Flags:

```
29 if (IO.staticReturnsTrue())
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:45
Taint Flags:

```
42 SecretKey secretKey = keyGenerator.generateKey();
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, 3DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
46 Cipher tripleDesCipher = Cipher.getInstance("DESede");
47 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 32 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java:32
Taint Flags:

```
29 while(true)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java:39
Taint Flags:

```
36 if (PRIVATE_STATIC_FINAL_TRUE)
37 {
38 final String CIPHER_INPUT = "ABCDEFG123456";
39 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
40 /* Perform initialization of KeyGenerator */
41 keyGenerator.init(56);
42 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 44 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 44 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java:44
Taint Flags:

```
41 SecretKey secretKey = keyGenerator.generateKey();
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
45 Cipher desCipher = Cipher.getInstance("DES");
46 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java:45
Taint Flags:

```
42 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 45 (Weak Encryption) — Critical

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
46 Cipher desCipher = Cipher.getInstance("DES");
47 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:39
Taint Flags:

```
36 if (PRIVATE_STATIC_FINAL_TRUE)
37 {
38 final String CIPHER_INPUT = "ABCDEFG123456";
39 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
40 /* Perform initialization of KeyGenerator */
41 keyGenerator.init(112);
42 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 39 (Weak Encryption) — Critical

```
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 46 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:46
Taint Flags:

```
43 if (privateReturnsTrue())
44 {
45 final String CIPHER_INPUT = "ABCDEFG123456";
46 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
47 /* Perform initialization of KeyGenerator */
48 keyGenerator.init(112);
49 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 46 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java:46
Taint Flags:

```
43 if (privateReturnsTrue())
44 {
45 final String CIPHER_INPUT = "ABCDEFG123456";
46 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
47 /* Perform initialization of KeyGenerator */
48 keyGenerator.init(56);
49 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 46 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
46 Cipher desCipher = Cipher.getInstance("DES");
47 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 46 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 46 (Weak Encryption) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
46 Cipher desCipher = Cipher.getInstance("DES");
47 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 38 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 39 (Weak Encryption) — Critical

```
37  /* FLAW: Use a weak crypto algorithm, 3DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39  Cipher tripleDesCipher = Cipher.getInstance("DESede");
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java:39
Taint Flags:

```
36  SecretKey secretKey = keyGenerator.generateKey();
37  byte[] byteKey = secretKey.getEncoded();
38  /* FLAW: Use a weak crypto algorithm, DES */
39  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
40  Cipher desCipher = Cipher.getInstance("DES");
41  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
42  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:32
Taint Flags:

```
29  if (IO.staticTrue)
30  {
31  final String CIPHER_INPUT = "ABCDEFG123456";
32  KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 32 (Weak Encryption) — Critical

```
33  /* Perform initialization of KeyGenerator */
34  keyGenerator.init(112);
35  SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:39
Taint Flags:

```
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, 3DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39  Cipher tripleDesCipher = Cipher.getInstance("DESede");
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java:39
Taint Flags:

```
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java:32
Taint Flags:

```
29 if (IO.STATIC_FINAL_TRUE)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 38 (Weak Encryption) — Critical

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 38 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 40 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java:40
Taint Flags:

```
37 byte[] byteKey = secretKey.getEncoded();
38 /* FLAW: Use a weak crypto algorithm, DES */
39 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
40 Cipher desCipher = Cipher.getInstance("DES");
41 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
42 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
43 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java:32
Taint Flags:

```
29
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 32 (Weak Encryption) — Critical

```
30 final String CIPHER_INPUT = "ABCDEFG123456";
31
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33
34 /* Perform initialization of KeyGenerator */
35 keyGenerator.init(56);
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 39 (Weak Encryption) — Critical

```
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 44 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:44
Taint Flags:

```
41 SecretKey secretKey = keyGenerator.generateKey();
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, 3DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
45 Cipher tripleDesCipher = Cipher.getInstance("DESede");
46 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java:32
Taint Flags:

```
29 for(int j = 0; j < 1; j++)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(112);
35 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java:39
Taint Flags:

```
36 if (privateTrue)
37 {
38 final String CIPHER_INPUT = "ABCDEFG123456";
39 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
40 /* Perform initialization of KeyGenerator */
41 keyGenerator.init(56);
42 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 41 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 41 (Weak Encryption) — Critical

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java:41
Taint Flags:

```
38 byte[] byteKey = secretKey.getEncoded();
39
40 /* FLAW: Use a weak crypto algorithm, 3DES */
41 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
42
43 Cipher tripleDesCipher = Cipher.getInstance("DESede");
44 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:45
Taint Flags:

```
42 SecretKey secretKey = keyGenerator.generateKey();
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, 3DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
46 Cipher tripleDesCipher = Cipher.getInstance("DESede");
47 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 45 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
45 Cipher desCipher = Cipher.getInstance("DES");
46 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java:32
Taint Flags:

```
29 if (IO.staticTrue)
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 32 (Weak Encryption) — Critical

```
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java:32
Taint Flags:

```
29 if (IO.STATIC_FINAL_FIVE == 5)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 43 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java:43
Taint Flags:

```
40 /* FLAW: Use a weak crypto algorithm, DES */
41 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
42
43 Cipher desCipher = Cipher.getInstance("DES");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 43 (Weak Encryption) — Critical

```
44 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
45
46 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:39
Taint Flags:

```
36 if (privateTrue)
37 {
38 final String CIPHER_INPUT = "ABCDEFG123456";
39 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
40 /* Perform initialization of KeyGenerator */
41 keyGenerator.init(112);
42 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:38
Taint Flags:

```
35 if (privateFive == 5)
36 {
37 final String CIPHER_INPUT = "ABCDEFG123456";
38 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
39 /* Perform initialization of KeyGenerator */
40 keyGenerator.init(112);
41 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 41 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java:41
Taint Flags:

```
38 byte[] byteKey = secretKey.getEncoded();
39
40 /* FLAW: Use a weak crypto algorithm, DES */
41 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
42
43 Cipher desCipher = Cipher.getInstance("DES");
44 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 38 (Weak Encryption) — Critical

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java:38
Taint Flags:

```
35 SecretKey secretKey = keyGenerator.generateKey();
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 33 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 33 (Weak Encryption) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java:33
Taint Flags:

```
30 {
31 case 7:
32 final String CIPHER_INPUT = "ABCDEFG123456";
33 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
34 /* Perform initialization of KeyGenerator */
35 keyGenerator.init(56);
36 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 53 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:53
Taint Flags:

```
50 byte[] byteKey = secretKey.getEncoded();
51 /* FLAW: Use a weak crypto algorithm, 3DES */
52 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
53 Cipher tripleDesCipher = Cipher.getInstance("DESede");
54 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
55 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
56 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java:38
Taint Flags:

```
35 if (privateFive == 5)
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 38 (Weak Encryption) — Critical

```
36 {
37 final String CIPHER_INPUT = "ABCDEFG123456";
38 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
39 /* Perform initialization of KeyGenerator */
40 keyGenerator.init(56);
41 SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:32
Taint Flags:

```
29 if (IO.STATIC_FINAL_TRUE)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DESede");
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 32 (Weak Encryption) — Critical

```
33  /* Perform initialization of KeyGenerator */
34  keyGenerator.init(112);
35  SecretKey secretKey = keyGenerator.generateKey();
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java:38
Taint Flags:

```
35  SecretKey secretKey = keyGenerator.generateKey();
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 38 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:38
Taint Flags:

```
35  SecretKey secretKey = keyGenerator.generateKey();
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, 3DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39  Cipher tripleDesCipher = Cipher.getInstance("DESede");
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 39 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 53 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java:53
Taint Flags:

```
50 byte[] byteKey = secretKey.getEncoded();
51 /* FLAW: Use a weak crypto algorithm, DES */
52 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
53 Cipher desCipher = Cipher.getInstance("DES");
54 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
55 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
56 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 44 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 44 (Weak Encryption) — Critical

Sink Details

Sink: SecretKeySpec()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java:44
Taint Flags:

```
41 SecretKey secretKey = keyGenerator.generateKey();
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
45 Cipher desCipher = Cipher.getInstance("DES");
46 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 45 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, 3DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
45 Cipher tripleDesCipher = Cipher.getInstance("DESede");
46 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 32 (Weak Encryption) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption	Critical
Package: testcases.CWE327_Use_Broken_Crypto	
testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 32 (Weak Encryption)	**Critical**

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java:32
Taint Flags:

```
29 if (true)
30 {
31 final String CIPHER_INPUT = "ABCDEFG123456";
32 KeyGenerator keyGenerator = KeyGenerator.getInstance("DES");
33 /* Perform initialization of KeyGenerator */
34 keyGenerator.init(56);
35 SecretKey secretKey = keyGenerator.generateKey();
```

Weak Encryption: Insecure Mode of Operation (98 issues)

Abstract

Do not use cryptographic encryption algorithms with an insecure mode of operation.

Explanation

The mode of operation of a block cipher is an algorithm that describes how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block. Some modes of operation include Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), and Counter (CTR). ECB mode is inherently weak, as it produces the same ciphertext for identical blocks of plain text. CBC mode is vulnerable to padding oracle attacks. CTR mode is the superior choice because it does not have these weaknesses. **Example 1:** The following code uses the AES cipher with ECB mode:

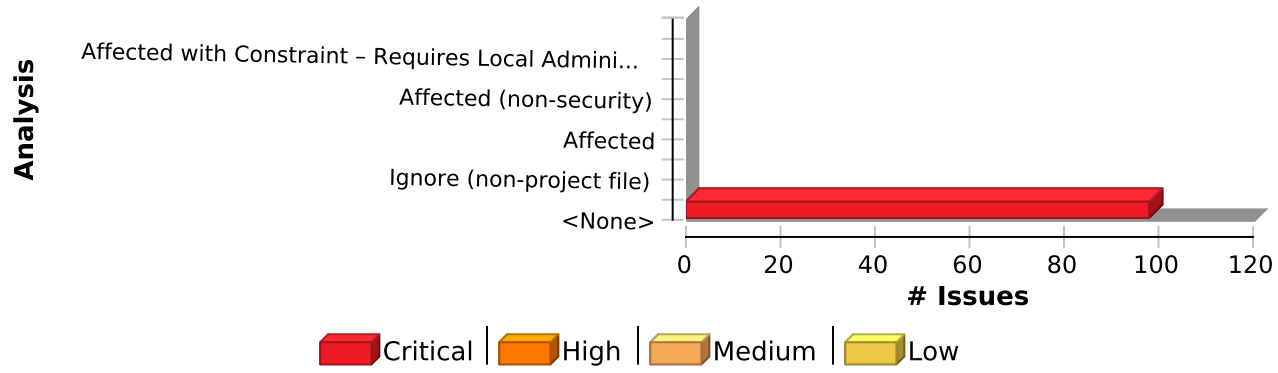
```
...
SecretKeySpec key = new SecretKeySpec(keyBytes, "AES");
Cipher cipher = Cipher.getInstance("AES/ECB/PKCS7Padding", "BC");
cipher.init(Cipher.ENCRYPT_MODE, key);
...
```

Recommendation

Avoid using ECB and CBC modes of operation when encrypting data larger than a block. CBC mode is somewhat inefficient and poses a serious risk if used with SSL [1]. Instead, use CCM (Counter with CBC-MAC) mode or, if performance is a concern, GCM (Galois/Counter Mode) mode where they are available. **Example 2:** The following code uses the AES cipher with GCM mode:

```
...
SecretKeySpec key = new SecretKeySpec(keyBytes, "AES");
Cipher cipher = Cipher.getInstance("AES/GCM/PKCS5Padding", "BC");
cipher.init(Cipher.ENCRYPT_MODE, key);
...
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Weak Encryption: Insecure Mode of Operation	98	0	0	98
Total	**98**	**0**	**0**	**98**

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 53 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java:53
Taint Flags:

```
50 byte[] byteKey = secretKey.getEncoded();
51 /* FLAW: Use a weak crypto algorithm, DES */
52 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
53 Cipher desCipher = Cipher.getInstance("DES");
54 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
55 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
56 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 60 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 60 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:60
Taint Flags:

```
57 /* FIX: Use a stronger crypto algorithm, AES */
58 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
59
60 Cipher aesCipher = Cipher.getInstance("AES");
61 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
62
63 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 99 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java:99
Taint Flags:

```
96 byte[] byteKey = secretKey.getEncoded();
97 /* FIX: Use a stronger crypto algorithm, AES */
98 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
99 Cipher aesCipher = Cipher.getInstance("AES");
100 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
101 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
102 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 100 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 100 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java:100
Taint Flags:

```
97  byte[] byteKey = secretKey.getEncoded();
98  /* FIX: Use a stronger crypto algorithm, AES */
99  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
100 Cipher aesCipher = Cipher.getInstance("AES");
101 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
102 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
103 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 76 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java:76
Taint Flags:

```
73  /* FIX: Use a stronger crypto algorithm, AES */
74  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
75
76  Cipher aesCipher = Cipher.getInstance("AES");
77  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
78
79  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:39
Taint Flags:

```
36  byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

```
37  /* FLAW: Use a weak crypto algorithm, 3DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39  Cipher tripleDesCipher = Cipher.getInstance("DESede");
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
94  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

```
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 45 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, 3DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
45 Cipher tripleDesCipher = Cipher.getInstance("DESede");
46 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 104 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 104 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:104
Taint Flags:

```
101 /* FIX: Use a stronger crypto algorithm, AES */
102 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
103
104 Cipher aesCipher = Cipher.getInstance("AES");
105 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
106
107 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 83 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:83
Taint Flags:

```
80 byte[] byteKey = secretKey.getEncoded();
81 /* FIX: Use a stronger crypto algorithm, AES */
82 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
83 Cipher aesCipher = Cipher.getInstance("AES");
84 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
85 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
86 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 73 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 73 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java:73
Taint Flags:

```
70 /* FIX: Use a stronger crypto algorithm, AES */
71 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
72
73 Cipher aesCipher = Cipher.getInstance("AES");
74 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
75
76 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 59 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java:59
Taint Flags:

```
56 byte[] byteKey = secretKey.getEncoded();
57 /* FIX: Use a stronger crypto algorithm, AES */
58 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
59 Cipher aesCipher = Cipher.getInstance("AES");
60 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
61 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
62 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 77 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:77
Taint Flags:

```
74 /* FIX: Use a stronger crypto algorithm, AES */
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 77 (Weak Encryption: Insecure Mode of Operation) — Critical

```
75  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
76
77  Cipher aesCipher = Cipher.getInstance("AES");
78  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
79
80  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:70
Taint Flags:

```
67  byte[] byteKey = secretKey.getEncoded();
68  /* FIX: Use a stronger crypto algorithm, AES */
69  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
70  Cipher aesCipher = Cipher.getInstance("AES");
71  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
73  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 107 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java:107
Taint Flags:

```
104  byte[] byteKey = secretKey.getEncoded();
105  /* FIX: Use a stronger crypto algorithm, AES */
106  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
107  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 107 (Weak Encryption: Insecure Mode of Operation) — Critical

```
108  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
109  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
110  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 77 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java:77
Taint Flags:

```
74  /* FIX: Use a stronger crypto algorithm, AES */
75  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
76
77  Cipher aesCipher = Cipher.getInstance("AES");
78  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
79
80  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java:39
Taint Flags:

```
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
94  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java:70
Taint Flags:

```
67  /* FIX: Use a stronger crypto algorithm, AES */
68  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70  Cipher aesCipher = Cipher.getInstance("AES");
71  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 100 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 100 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:100
Taint Flags:

```
97  byte[] byteKey = secretKey.getEncoded();
98  /* FIX: Use a stronger crypto algorithm, AES */
99  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
100 Cipher aesCipher = Cipher.getInstance("AES");
101 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
102 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
103 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 73 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java:73
Taint Flags:

```
70  /* FIX: Use a stronger crypto algorithm, AES */
71  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
72
73  Cipher aesCipher = Cipher.getInstance("AES");
74  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
75
76  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 107 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 107 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:107
Taint Flags:

```
104 byte[] byteKey = secretKey.getEncoded();
105 /* FIX: Use a stronger crypto algorithm, AES */
106 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
107 Cipher aesCipher = Cipher.getInstance("AES");
108 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
109 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
110 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

```
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
94  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

```
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 76 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:76
Taint Flags:

```
73 /* FIX: Use a stronger crypto algorithm, AES */
74 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
75
76 Cipher aesCipher = Cipher.getInstance("AES");
77 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
78
79 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_03.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 45 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 45 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
45 Cipher desCipher = Cipher.getInstance("DES");
46 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_17.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:93
Taint Flags:

```
90 byte[] byteKey = secretKey.getEncoded();
91 /* FIX: Use a stronger crypto algorithm, AES */
92 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93 Cipher aesCipher = Cipher.getInstance("AES");
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:93
Taint Flags:

```
90 byte[] byteKey = secretKey.getEncoded();
91 /* FIX: Use a stronger crypto algorithm, AES */
92 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93 Cipher aesCipher = Cipher.getInstance("AES");
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

```
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 40 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java:40
Taint Flags:

```
37 byte[] byteKey = secretKey.getEncoded();
38 /* FLAW: Use a weak crypto algorithm, DES */
39 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
40 Cipher desCipher = Cipher.getInstance("DES");
41 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
42 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
43 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

```
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java:93
Taint Flags:

```
90 byte[] byteKey = secretKey.getEncoded();
91 /* FIX: Use a stronger crypto algorithm, AES */
92 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93 Cipher aesCipher = Cipher.getInstance("AES");
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 77 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:77
Taint Flags:

```
74 /* FIX: Use a stronger crypto algorithm, AES */
75 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
76
77 Cipher aesCipher = Cipher.getInstance("AES");
78 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
79
80 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java:70
Taint Flags:

```
67 byte[] byteKey = secretKey.getEncoded();
68 /* FIX: Use a stronger crypto algorithm, AES */
69 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
73 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:70
Taint Flags:

```
67  /* FIX: Use a stronger crypto algorithm, AES */
68  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70  Cipher aesCipher = Cipher.getInstance("AES");
71  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java, line 60 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_16.java:60
Taint Flags:

```
57  byte[] byteKey = secretKey.getEncoded();
58  /* FIX: Use a stronger crypto algorithm, AES */
59  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
60  Cipher aesCipher = Cipher.getInstance("AES");
61  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
62  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
63  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 84 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:84
Taint Flags:

```
81 /* FIX: Use a stronger crypto algorithm, AES */
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 84 (Weak Encryption: Insecure Mode of Operation) — Critical

```
82  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
83
84  Cipher aesCipher = Cipher.getInstance("AES");
85  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
86
87  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 99 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:99
Taint Flags:

```
96  byte[] byteKey = secretKey.getEncoded();
97  /* FIX: Use a stronger crypto algorithm, AES */
98  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
99  Cipher aesCipher = Cipher.getInstance("AES");
100 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
101 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
102 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

```
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 59 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java:59
Taint Flags:

```
56 byte[] byteKey = secretKey.getEncoded();
57 /* FIX: Use a stronger crypto algorithm, AES */
58 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
59 Cipher aesCipher = Cipher.getInstance("AES");
60 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
61 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
62 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 46 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java, line 46 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_04.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
46 Cipher desCipher = Cipher.getInstance("DES");
47 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java:93
Taint Flags:

```
90 byte[] byteKey = secretKey.getEncoded();
91 /* FIX: Use a stronger crypto algorithm, AES */
92 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93 Cipher aesCipher = Cipher.getInstance("AES");
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 60 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java:60
Taint Flags:

```
57 byte[] byteKey = secretKey.getEncoded();
58 /* FIX: Use a stronger crypto algorithm, AES */
59 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
60 Cipher aesCipher = Cipher.getInstance("AES");
61 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
62 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
63 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java:93
Taint Flags:

```
90 byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_11.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

```
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
94  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 104 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:104
Taint Flags:

```
101  /* FIX: Use a stronger crypto algorithm, AES */
102  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
103
104  Cipher aesCipher = Cipher.getInstance("AES");
105  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
106
107  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 83 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:83
Taint Flags:

```
80  byte[] byteKey = secretKey.getEncoded();
81  /* FIX: Use a stronger crypto algorithm, AES */
82  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
83  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 83 (Weak Encryption: Insecure Mode of Operation) — Critical

```
84 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
85 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
86 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 46 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, 3DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
46 Cipher tripleDesCipher = Cipher.getInstance("DESede");
47 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
40 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 46 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, 3DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
46 Cipher tripleDesCipher = Cipher.getInstance("DESede");
47 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 76 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:76
Taint Flags:

```
73 /* FIX: Use a stronger crypto algorithm, AES */
74 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
75
76 Cipher aesCipher = Cipher.getInstance("AES");
77 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
78
79 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 53 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 53 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:53
Taint Flags:

```
50 byte[] byteKey = secretKey.getEncoded();
51 /* FLAW: Use a weak crypto algorithm, 3DES */
52 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
53 Cipher tripleDesCipher = Cipher.getInstance("DESede");
54 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
55 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
56 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java, line 43 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_01.java:43
Taint Flags:

```
40 /* FLAW: Use a weak crypto algorithm, DES */
41 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
42
43 Cipher desCipher = Cipher.getInstance("DES");
44 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
45
46 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 40 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 40 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:40
Taint Flags:

```
37 byte[] byteKey = secretKey.getEncoded();
38 /* FLAW: Use a weak crypto algorithm, 3DES */
39 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
40 Cipher tripleDesCipher = Cipher.getInstance("DESede");
41 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
42 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
43 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 92 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:92
Taint Flags:

```
89 byte[] byteKey = secretKey.getEncoded();
90 /* FIX: Use a stronger crypto algorithm, AES */
91 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
92 Cipher aesCipher = Cipher.getInstance("AES");
93 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
94 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
95 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

```
37  /* FLAW: Use a weak crypto algorithm, 3DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39  Cipher tripleDesCipher = Cipher.getInstance("DESede");
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
94  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:93
Taint Flags:

```
90  byte[] byteKey = secretKey.getEncoded();
91  /* FIX: Use a stronger crypto algorithm, AES */
92  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

```
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_09.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 60 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:60
Taint Flags:

```
57 /* FIX: Use a stronger crypto algorithm, AES */
58 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
59
60 Cipher aesCipher = Cipher.getInstance("AES");
61 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
62
63 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:70
Taint Flags:

```
67  /* FIX: Use a stronger crypto algorithm, AES */
68  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70  Cipher aesCipher = Cipher.getInstance("AES");
71  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 99 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java:99
Taint Flags:

```
96  byte[] byteKey = secretKey.getEncoded();
97  /* FIX: Use a stronger crypto algorithm, AES */
98  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
99  Cipher aesCipher = Cipher.getInstance("AES");
100 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
101 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
102 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 45 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 45 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, 3DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
45 Cipher tripleDesCipher = Cipher.getInstance("DESede");
46 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java, line 45 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_07.java:45
Taint Flags:

```
42 byte[] byteKey = secretKey.getEncoded();
43 /* FLAW: Use a weak crypto algorithm, DES */
44 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
45 Cipher desCipher = Cipher.getInstance("DES");
46 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
47 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
48 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_10.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java, line 84 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_08.java:84
Taint Flags:

```
81 /* FIX: Use a stronger crypto algorithm, AES */
82 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
83
84 Cipher aesCipher = Cipher.getInstance("AES");
85 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
86
87 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_13.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

```
37  /* FLAW: Use a weak crypto algorithm, DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39  Cipher desCipher = Cipher.getInstance("DES");
40  desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:39
Taint Flags:

```
36  byte[] byteKey = secretKey.getEncoded();
37  /* FLAW: Use a weak crypto algorithm, 3DES */
38  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39  Cipher tripleDesCipher = Cipher.getInstance("DESede");
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 92 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java:92
Taint Flags:

```
89  byte[] byteKey = secretKey.getEncoded();
90  /* FIX: Use a stronger crypto algorithm, AES */
91  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
92  Cipher aesCipher = Cipher.getInstance("AES");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_15.java, line 92 (Weak Encryption: Insecure Mode of Operation) — Critical

```
93  aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
94  byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
95  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 100 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java:100
Taint Flags:

```
97  byte[] byteKey = secretKey.getEncoded();
98  /* FIX: Use a stronger crypto algorithm, AES */
99  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
100 Cipher aesCipher = Cipher.getInstance("AES");
101 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
102 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
103 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 99 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:99
Taint Flags:

```
96  byte[] byteKey = secretKey.getEncoded();
97  /* FIX: Use a stronger crypto algorithm, AES */
98  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
99  Cipher aesCipher = Cipher.getInstance("AES");
100 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
101 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
102 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 77 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java:77
Taint Flags:

```
74 /* FIX: Use a stronger crypto algorithm, AES */
75 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
76
77 Cipher aesCipher = Cipher.getInstance("AES");
78 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
79
80 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_02.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 93 (Weak Encryption: Insecure Mode of Operation) — Critical

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:93
Taint Flags:

```
90 byte[] byteKey = secretKey.getEncoded();
91 /* FIX: Use a stronger crypto algorithm, AES */
92 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
93 Cipher aesCipher = Cipher.getInstance("AES");
94 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
95 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
96 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java, line 46 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_05.java:46
Taint Flags:

```
43 byte[] byteKey = secretKey.getEncoded();
44 /* FLAW: Use a weak crypto algorithm, DES */
45 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
46 Cipher desCipher = Cipher.getInstance("DES");
47 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
48 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
49 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_14.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DES");
39 Cipher desCipher = Cipher.getInstance("DES");
40 desCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41 byte[] encrypted = desCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42 IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java, line 76 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_06.java:76
Taint Flags:

```
73 /* FIX: Use a stronger crypto algorithm, AES */
74 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
75
76 Cipher aesCipher = Cipher.getInstance("AES");
77 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
78
79 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 43 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java:43
Taint Flags:

```
40 /* FLAW: Use a weak crypto algorithm, 3DES */
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 43 (Weak Encryption: Insecure Mode of Operation) — Critical

```
41 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
42
43 Cipher tripleDesCipher = Cipher.getInstance("DESede");
44 tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
45
46 byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 70 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good1()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:70
Taint Flags:

```
67 /* FIX: Use a stronger crypto algorithm, AES */
68 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
69
70 Cipher aesCipher = Cipher.getInstance("AES");
71 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
72
73 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:39
Taint Flags:

```
36 byte[] byteKey = secretKey.getEncoded();
37 /* FLAW: Use a weak crypto algorithm, 3DES */
38 SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "DESede");
39 Cipher tripleDesCipher = Cipher.getInstance("DESede");
```

Weak Encryption: Insecure Mode of Operation — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 39 (Weak Encryption: Insecure Mode of Operation) — Critical

```
40  tripleDesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
41  byte[] encrypted = tripleDesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
42  IO.writeLine(IO.toHex(encrypted));
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 100 (Weak Encryption: Insecure Mode of Operation) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: getInstance()
Enclosing Method: good2()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:100
Taint Flags:

```
97  byte[] byteKey = secretKey.getEncoded();
98  /* FIX: Use a stronger crypto algorithm, AES */
99  SecretKeySpec secretKeySpec = new SecretKeySpec(byteKey, "AES");
100 Cipher aesCipher = Cipher.getInstance("AES");
101 aesCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec);
102 byte[] encrypted = aesCipher.doFinal(CIPHER_INPUT.getBytes("UTF-8"));
103 IO.writeLine(IO.toHex(encrypted));
```

Weak Encryption: Insufficient Key Size (19 issues)

Abstract

An otherwise strong encryption algorithm is vulnerable to brute-force attack when an insufficient key size is used.

Explanation

Current cryptography guidelines suggest that a key length of at least 2048 bits should be used with the RSA algorithm. However, continued advancements in computing power and factoring techniques [1] mean that future increases in the recommended key size are inevitable. **Example 1:** The following code generates a 512-bit RSA encryption key.

```
  public static KeyPair getRSAKey() throws NoSuchAlgorithmException {
    KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");
    keyGen.initialize(512);

    KeyPair key = keyGen.generateKeyPair();
    return key;
  }
```

When it comes to symmetric encryption, the key lengths should be at least 128 bits. This category was derived from the Cigital Java Rulepack.

Recommendation

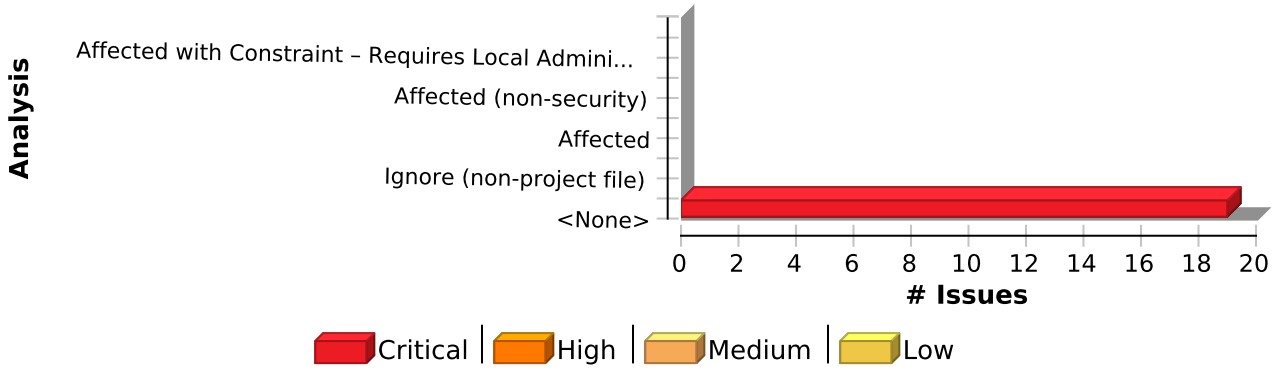
At a bare minimum, ensure RSA keys are no less than 2048 bits long. Applications that require strong encryption for the next several years should use keys at least 4096 bits long. When the RSA algorithm is used, ensure the specified key length is at least 2048 bits. **Example 2:** The following code generates a 2048-bit RSA encryption key.

```
  public static KeyPair getGoodRSAKey() throws NoSuchAlgorithmException {
    KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA");
    keyGen.initialize(2048);

    KeyPair key = keyGen.generateKeyPair();
    return key;
  }
```

Similarly, when symmetric encryption is used, ensure the specified key length is at least 128 bits for AES and 168 bits for Triple DES. **Example 3:** The following code generates a 128-bit AES encryption key.

```
  public static SecretKey getGoodAESKey() throws NoSuchAlgorithmException {
    KeyGenerator keyGen = KeyGenerator.getInstance("AES");
    keyGen.init(128);

    SecretKey key = keyGen.generateKey();
    return key;
  }
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Weak Encryption: Insufficient Key Size	19	0	0	19
Total	**19**	**0**	**0**	**19**

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_01.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_01 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29
30  final String CIPHER_INPUT = "ABCDEFG123456";
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 33 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java, line 33 (Weak Encryption: Insufficient Key Size) — Critical

Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_07.java:33
Taint Flags:

```
30 */
31 private int privateFive = 5;
32
33 public void bad() throws Throwable
34 {
35 if (privateFive == 5)
36 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java, line 34 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_04.java:34
Taint Flags:

```
31 private static final boolean PRIVATE_STATIC_FINAL_TRUE = true;
32 private static final boolean PRIVATE_STATIC_FINAL_FALSE = false;
33
34 public void bad() throws Throwable
35 {
36 if (PRIVATE_STATIC_FINAL_TRUE)
37 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:27
Taint Flags:

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_12 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  if (IO.staticReturnsTrueOrFalse())
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__DES_12.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__DES_12 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  if (IO.staticReturnsTrueOrFalse())
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_11 extends AbstractTestCase
26 {
```

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_11.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

```
27 public void bad() throws Throwable
28 {
29 if (IO.staticReturnsTrue())
30 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_09.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_09 extends AbstractTestCase
26 {
27 public void bad() throws Throwable
28 {
29 if (IO.STATIC_FINAL_TRUE)
30 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_14 extends AbstractTestCase
26 {
27 public void bad() throws Throwable
28 {
29 if (IO.staticFive == 5)
```

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_14.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

```
30 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_16.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_16 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  while(true)
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java, line 33 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_06.java:33
Taint Flags:

```
30  */
31  private static final int PRIVATE_STATIC_FINAL_FIVE = 5;
32
33  public void bad() throws Throwable
34  {
35  if (PRIVATE_STATIC_FINAL_FIVE == 5)
36  {
```

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_13.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_13 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  if (IO.STATIC_FINAL_FIVE == 5)
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java, line 34 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_05.java:34
Taint Flags:

```
31  private boolean privateTrue = true;
32  private boolean privateFalse = false;
33
34  public void bad() throws Throwable
35  {
36  if (privateTrue)
37  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_17.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_17 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  for(int j = 0; j < 1; j++)
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_02.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_02 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  if (true)
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 41 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java, line 41 (Weak Encryption: Insufficient Key Size) — Critical

File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_08.java:41
Taint Flags:

```
38 return false;
39 }
40
41 public void bad() throws Throwable
42 {
43 if (privateReturnsTrue())
44 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_15.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_15 extends AbstractTestCase
26 {
27 public void bad() throws Throwable
28 {
29 switch (7)
30 {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java:27
Taint Flags:

```
24
```

Weak Encryption: Insufficient Key Size — Critical

Package: testcases.CWE327_Use_Broken_Crypto

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_12.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

```
25 public class CWE327_Use_Broken_Crypto__3DES_12 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  if (IO.staticReturnsTrueOrFalse())
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_10.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_10 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
28  {
29  if (IO.staticTrue)
30  {
```

testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 27 (Weak Encryption: Insufficient Key Size) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: bad
Enclosing Method: bad()
File: testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java:27
Taint Flags:

```
24
25 public class CWE327_Use_Broken_Crypto__3DES_03 extends AbstractTestCase
26 {
27  public void bad() throws Throwable
```

Weak Encryption: Insufficient Key Size	Critical
Package: testcases.CWE327_Use_Broken_Crypto	
testcases/CWE327_Use_Broken_Crypto/CWE327_Use_Broken_Crypto__3DES_03.java, line 27 (Weak Encryption: Insufficient Key Size)	**Critical**

```
28  {
29  if (5 == 5)
30  {
```