



Scaling the Security Researcher to Eliminate OSS Vulnerabilities Once and for All

- Jonathan Leitschuh -
- Patrick Way -

Hello!

- Jonathan Leitschuh -

Software Engineer & Security Researcher

Dan Kaminsky Fellowship @ HUMAN Security

GitHub Star & GitHub Security Ambassador

Twitter: @JLLeitschuh

GitHub: JLLeitschuh



Hello!

- Patrick Way -

Senior Software Engineer

OpenRewrite Team @ Moderne

Twitter: @WayPatrick

GitHub: pway99



Disclaimer

Supported by
The
Dan Kaminsky Fellowship
at
HUMAN Security



Chester Higgins/The New York Times

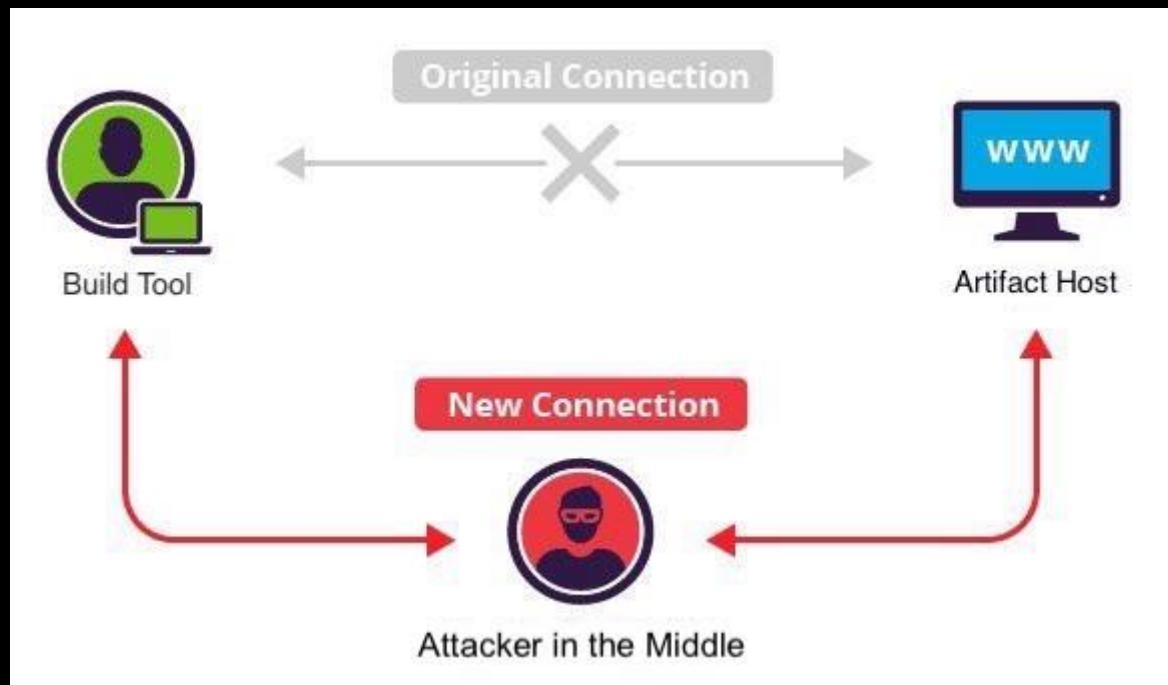
It Started
With a Simple Vulnerability

```
// build.gradle

maven {
    setUrl("http://dl.bintray.com/kotlin/ktor")
}
```

HTTP Download of Dependencies in the Java Ecosystem

Why is HTTPS important?



```
<!-- Compiler & Test Dependencies -->
<repositories>
  <repository>
    <id>example-id</id>
    <name>Example insecure repository</name>
    <url>http://[SOME URL HERE]</url>
  </repository>
</repositories>
```

HTTP Download of Dependencies in the Java Ecosystem

```
<!-- Artifact upload - Credentials!! -->
<distributionManagement>
  <repository>
    <id>example-id</id>
    <name>Example insecure repository</name>
    <url>http://[SOME URL HERE]</url>
  </repository>
</distributionManagement>
```

HTTP Download of Dependencies in the Java Ecosystem

This Vulnerability was Everywhere!



Jenkins



Who else was vulnerable?

ORACLE®



LinkedIn stripe

“25% of Sonatype Maven
Central downloads are still
using HTTP”

- Sonatype June 2019 -

How do we fix this?

Decommissioning HTTP Support

On or around January 15th, 2020

- Maven Central (Sonatype)
- JCenter (JFrog)
- Spring (Pivotal)
- Gradle Plugin Portal (Gradle)

“20% of Sonatype Maven
Central Traffic is STILL using
HTTP”

- Sonatype January 2020 -

You can imagine what happened...
January 15th, 2020

BROKEN SOFTWARE



BROKEN SOFTWARE EVERYWHERE

We stopped the bleeding

What about the other repositories?

Only the *most commonly* used repositories

- Maven Central (Sonatype)
- JCenter (JFrog)
- Spring (Pivotal)
- Gradle Plugin Portal (Gradle)

How do we fix the rest?

Bulk Pull Request Generation!

How?

CodeQL

```
import java
import semmle.code.xml.MavenPom

private class DeclaredRepository extends PomElement {
    DeclaredRepository() {
        this.getName() = "repository" or
        this.getName() = "snapshotRepository" or
        this.getName() = "pluginRepository"
    }

    string getUrl() { result = getChild("url").(PomElement).getValue() }

    predicate isInsecureRepositoryUsage() {
        getUrl().matches("http://%") or
        getUrl().matches("ftp://%")
    }
}

from DeclaredRepository repository
where repository.isInsecureRepositoryUsage()
select repository,
    "Downloading or uploading artifacts over insecure protocol (eg. http or ftp) to/from repository " +
    repository.getUrl()
```

CodeQL scans 100Ks of OSS Projects

CodeQL

```
import java
import semmle.code.xml.MavenPom

private class DeclaredRepository extends PomElement {
    DeclaredRepository() {
        this.getName() = "repository" or
        this.getName() = "snapshotRepository" or
        this.getName() = "pluginRepository"
    }

    string getUrl() { result = getChild("url").(PomElement).getValue() }

    predicate isInsecureRepositoryUsage() {
        getUrl().matches("http://%") or
        getUrl().matches("ftp://%")
    }
}

from DeclaredRepository repository
where repository.isInsecureRepositoryUsage()
select repository,
    "Downloading or uploading artifacts over insecure protocol (eg. http or ftp) to/from repository " +
    repository.getUrl()
```

\$2,300 Bounty

Pull Request Generator Version 1

- Python Based
- Wrapper over ‘hub’ CLI
- One Nasty Regular Expression
- Bouncing off GitHub’s rate limiter

```
from vulnerability_fix_engine import VulnerabilityFixModule


@dataclass()
class PomVulnerabilityFixModule(VulnerabilityFixModule):
    branch_name: str = 'fix/JLL/use_https_to_resolve_dependencies'
    clone_repos_location: str = 'cloned_repos'
    data_base_dir: str = 'insecure_pom_data'
    save_point_location: str = 'save_points'
    pr_message_file_absolute_path: str = f'{str(pathlib.Path().absolute())}/PR_MESSAGE.md'
    commit_message: str = textwrap.dedent('''\
        Use HTTPS instead of HTTP to resolve dependencies

        This fixes a security vulnerability in this project where the 'pom.xml'
        files were configuring Maven to resolve dependencies over HTTP instead of
        HTTPS.

        Signed-off-by: Jonathan Leitschuh <Jonathan.Leitschuh@gmail.com>
    ''')

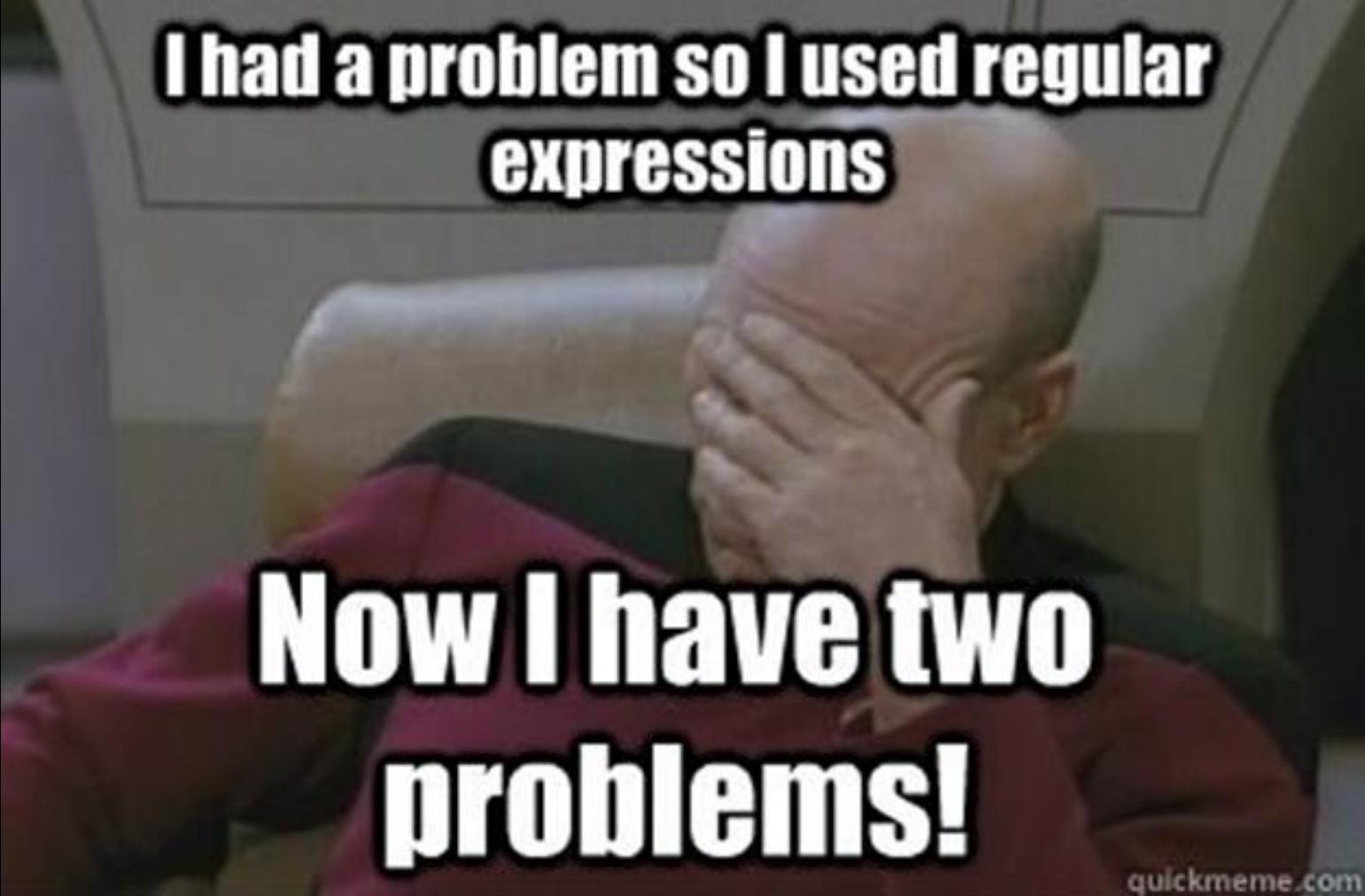
    p_fix_regex = \
        re.compile(
            r'(?:(?<=repository>)(?<=pluginRepository>)|(?<=snapshotRepository>))((?:(!repository>).*)|(<url>\s*http://((\s*)|\s*</url>)', re.IGNORECASE + re.MULTILINE + re.DOTALL
        )
    replacement = r'\1\2https://\3\4'

    @async def do_fix_vulnerable_file(self, project_name: str, file: str, expected_fix_count: int) -> int:
        async with aiofiles.open(file, newline='') as vulnerableFile:
            contents: str = await vulnerableFile.read()

            new_contents, count = self.p_fix_regex.subn(self.replacement, contents)
            if count != expected_fix_count:
                logging.warning(
                    'Fix for %s did not match expected fix count: (expected: %d, actual: %d)',
                    project_name,
                    expected_fix_count,
                    count
                )

            async with aiofiles.open(file, 'w', newline='') as vulnerableFile:
                await vulnerableFile.write(new_contents)
        return count
```

```
... p_fix_regex = \
...     re.compile(
...         r'(?:(?=<repository>) | (?=<pluginRepository>) | (?=<snapshotRepository>))((?:(?!\repository>).)*)(<url>\s*)http://(\s*)(\s*</url>)',
...         re.IGNORECASE + re.MULTILINE + re.DOTALL
...     )
... replacement = r'\1\2https://\3\4'
```



**I had a problem so I used regular
expressions**

**Now I have two
problems!**

It worked!



is:pr author:JLLeitschuh archived:false

Pull requests Issues Marketplace Explore



Created

Assigned

Mentioned

Review requests

is:pr author:JLLeitschuh archived:false [SECURITY] Use HTTPS to resolve de

1,055 Open ✓ 504 Closed

Visibility ▾

Organization ▾

Sort ▾

01Sharpshooter/Social [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#1 opened on Feb 11 by JLLeitschuh

4thline/cling [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#250 opened on Feb 11 by JLLeitschuh

1000Memories/photon-core [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#4 opened on Feb 11 by JLLeitschuh

18838928050/ssmtest [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#1 opened on Feb 11 by JLLeitschuh

2xel/spring-boot-strap-tiles [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#1 opened on Feb 11 by JLLeitschuh

weamylady2/iOS_remote [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#23 opened on Feb 11 by JLLeitschuh

yjshen/zzzobospk [SECURITY] Use HTTPS to resolve dependencies in Maven Build ✓

#1 opened on Feb 11 by JLLeitschuh

1

wlu-mstr/hbase-ormlite [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#1 opened on Feb 11 by JLLeitschuh

zhangdaiscott/jeehg [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#53 opened on Feb 11 by JLLeitschuh

wso2/carbon-device-mgt-plugins [SECURITY] Use HTTPS to resolve dependencies in Maven Build ✕

#927 opened on Feb 11 by JLLeitschuh • Review required

21

wso2/product-iots [SECURITY] Use HTTPS to resolve dependencies in Maven Build ✓

Resolution/Stale

#1940 opened on Feb 11 by JLLeitschuh • Review required

17

xautlx/s2jh4net [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#30 opened on Feb 11 by JLLeitschuh

xzer/run-jetty-run [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#214 opened on Feb 11 by JLLeitschuh

yanghua/banyan [SECURITY] Use HTTPS to resolve dependencies in Maven Build

#3 opened on Feb 11 by JLLeitschuh

Viewed ...

```
@@ -19,15 +19,15 @@
```

```
19 </repository> 19 </repository>
20 <repository> 20 <repository>
21 <id>onarandombox</id> 21 <id>onarandombox</id>
22 <url>http://repo.onarandombox.com/content/groups/public</url> 22 <url>https://repo.onarandombox.com/content/groups/public</url>
23 </repository> 23 </repository>
24 <repository> 24 <repository>
25 <id>spigot</id> 25 <id>spigot</id>
26 <url>https://hub.spigotmc.org/nexus/content/groups/public</url> 26 <url>https://hub.spigotmc.org/nexus/content/groups/public</url>
27 </repository> 27 </repository>
28 <repository> 28 <repository>
29 <id>vault-repo</id> 29 <id>vault-repo</id>
30 <url>http://nexus.hc.to/content/repositories/pub\_releases</url> 30 <url>https://nexus.hc.to/content/repositories/pub\_releases</url>
31 </repository> 31 </repository>
32 <repository> 32 <repository>
33 <id>minebench-repo</id> 33 <id>minebench-repo</id>
```

```
@@ -36,15 +36,15 @@
```

```
36 <!-- Has a copy of metrics R8-SNAPSHOT !--> 36 <!-- Has a copy of metrics R8-SNAPSHOT !-->
37 <repository> 37 <repository>
38 <id>elmakers-repo</id> 38 <id>elmakers-repo</id>
39 <url>http://maven.elmakers.com/repository</url> 39 <url>https://maven.elmakers.com/repository</url>
40 </repository> 40 </repository>
41 </repositories> 41 </repositories>
42
43 <pluginRepositories> 43 <pluginRepositories>
44 <pluginRepository> 44 <pluginRepository>
45 <id>doodleproject-repo</id> 45 <id>doodleproject-repo</id>
46 <name>DoodleProject Maven 2 Repository</name> 46 <name>DoodleProject Maven 2 Repository</name>
47 <url>http://doodleproject.sourceforge.net/maven2/release</url> 47 <url>https://doodleproject.sourceforge.net/maven2/release</url>
48 <releases> 48 <releases>
49 <enabled>true</enabled> 49 <enabled>true</enabled>
50 </releases> 50 </releases>
```

```
@@ -353,11 +353,11 @@
```

```
353 <distributionManagement> 353 <distributionManagement>
354 <repository>
```

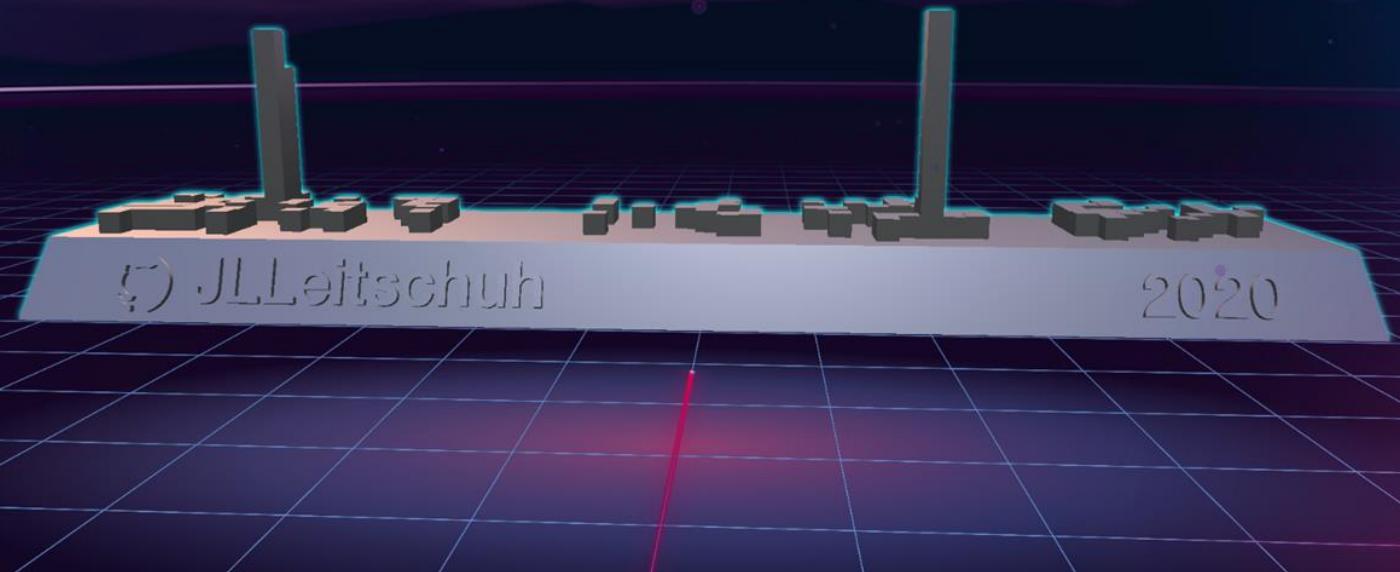
HTTP Download of Dependencies
1,596
Pull Requests

~40%
Merged or Accepted

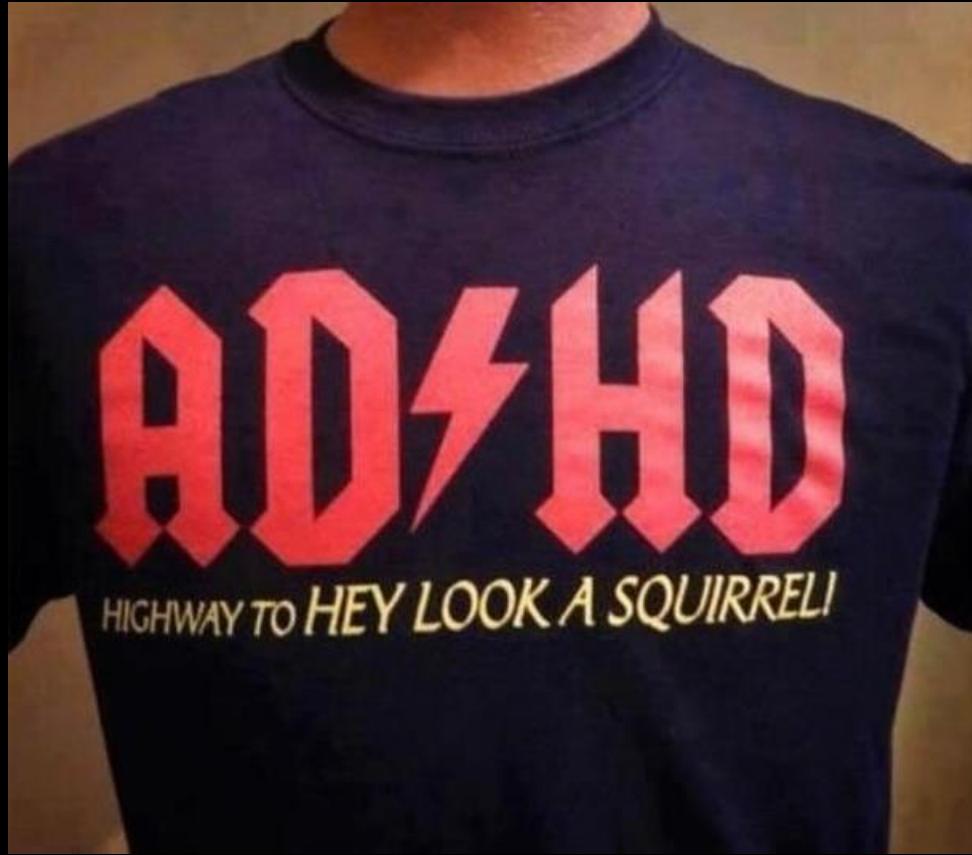
\$4,000

Thanks to the GitHub Security Lab!

I got hooked on
Bulk Pull Request Generation



I have a Problem



I was finding too many security vulnerabilities!

```
<wsos2/carbon-mediation>/components./utils/SynapseArtifactInitUtils.java
  1-254
  255      // If the entry is a file, write the file
  256      copyInputStream(zipFile.getInputStream(entry),
  257          new BufferedOutputStream(new FileOutputStream(destPath + entry.getName())));
  258  }
  259 }
  1  258-276

<apache/druid>/indexing-hadoop./indexer/JobHelper.java
  1-768
  769      try (ZipInputStream in = new ZipInputStream(fileSystem.open(zip, 1 << 13))) {
  770          for (ZipEntry entry = in.getNextEntry(); entry != null; entry = in.getNextEntry()) {
  771              final String filename = entry.getName();
  772
  773              final String outputPath = new File(outDir, filename).getAbsolutePath();
  774
  775          }
  1  774-864

<HongZhaohua/jstarcraft-core>/jstarcraft-core-common./utility/PressUtility.java
  1-179
  180      ArchiveEntry archiveEntry;
  181      while (null != (archiveEntry = archiveInputStream.getNextEntry())) {
  182          File file = new File(toDirectory, archiveEntry.getName());
  183
  184          try (FileOutputStream fileOutputStream = new FileOutputStream(file)) {
  185              int length = -1;
  186
  187          }
  188
  189          ArchiveEntry archiveEntry;
  190          while (null != (archiveEntry = archiveInputStream.getNextEntry())) {
  191              File file = new File(toDirectory, archiveEntry.getName());
  192
  193              try (FileOutputStream fileOutputStream = new FileOutputStream(file)) {
  194                  int length = -1;
  195
  196          }
  197
  198          ArchiveEntry archiveEntry;
  199          while (null != (archiveEntry = archiveInputStream.getNextEntry())) {
  200              File file = new File(toDirectory, archiveEntry.getName());
  201
  202              try (FileOutputStream fileOutputStream = new FileOutputStream(file)) {
  203                  int length = -1;
  204
  205          }
  206
  207          ArchiveEntry archiveEntry;
  208          while (null != (archiveEntry = archiveInputStream.getNextEntry())) {
  209              File file = new File(toDirectory, archiveEntry.getName());
  210
  211              try (FileOutputStream fileOutputStream = new FileOutputStream(file)) {
  212                  int length = -1;
  213
  214          }
  215
  216          ArchiveEntry archiveEntry;
  217          while (null != (archiveEntry = archiveInputStream.getNextEntry())) {
  218              File file = new File(toDirectory, archiveEntry.getName());
  219
  220              try (FileOutputStream fileOutputStream = new FileOutputStream(file)) {
  221                  int length = -1;
  222
  223          }
  1  223-233

<deepjavavlibrary/djl>/api./repository/AbstractRepository.java
  1-245
  246      TarArchiveEntry entry;
  247      while ((entry = tis.getNextTarEntry()) != null) {
  248          String entryName = entry.getName();
  249
  250          if (entryName.contains("..")) {
  251              throw new IOException("Malicious zip entry: " + entryName);
  1  251-373

<deepjavavlibrary/djl>/api./util/ZipUtils.java
  1-39
  40      ZipEntry entry;
  41      while ((entry = zis.getNextEntry()) != null) {
  42          String name = entry.getName();
  43
  44          if (name.contains("..")) {
  45              throw new IOException("Malicious zip entry: " + name);
  1  45-180

<cybertaxonomy/cdllib>/cdllib-ext./scratchpads/ScratchpadsService.java
  1-159
  160      System.out.println("Extracting: " + ze);
  161
  162      FileOutputStream fos = new FileOutputStream(ze.getName());
  163
  164      Unsanitized archive entry, which may contain '..' is used in a file system operation. Show paths
  165
  166
  167
  168
  169
  170
  171
  172
  173
  174
  175
  176
  177
  178
  179
  180
  181
  182
  183
  184
  185
  186
  187
  188
  189
  190
  191
  192
  193
  194
  195
  196
  197
  198
  199
  200
  201
  202
  203
  204
  205
  206
  207
  208
  209
  210
  211
  212
  213
  214
  215
  216
  217
  218
  219
  220
  221
  222
  223
  224
  225
  226
  227
  228
  229
  230
  231
  232
  233
  234
  235
  236
  237
  238
  239
  240
  241
  242
  243
  244
  245
  246
  247
  248
  249
  250
  251
  252
  253
  254
  255
  256
  257
  258
  259
  260
  261
  262
  263
  264
  265
  266
  267
  268
  269
  270
  271
  272
  273
  274
  275
  276
  277
  278
  279
  280
  281
  282
  283
  284
  285
  286
  287
  288
  289
  290
  291
  292
  293
  294
  295
  296
  297
  298
  299
  300
  301
  302
  303
  304
  305
  306
  307
  308
  309
  310
  311
  312
  313
  314
  315
  316
  317
  318
  319
  320
  321
  322
  323
  324
  325
  326
  327
  328
  329
  330
  331
  332
  333
  334
  335
  336
  337
  338
  339
  340
  341
  342
  343
  344
  345
  346
  347
  348
  349
  350
  351
  352
  353
  354
  355
  356
  357
  358
  359
  360
  361
  362
  363
  364
  365
  366
  367
  368
  369
  370
  371
  372
  373
  374
  375
  376
  377
  378
  379
  380
  381
  382
  383
  384
  385
  386
  387
  388
  389
  390
  391
  392
  393
  394
  395
  396
  397
  398
  399
  400
  401
  402
  403
  404
  405
  406
  407
  408
  409
  410
  411
  412
  413
  414
  415
  416
  417
  418
  419
  420
  421
  422
  423
  424
  425
  426
  427
  428
  429
  430
  431
  432
  433
  434
  435
  436
  437
  438
  439
  440
  441
  442
  443
  444
  445
  446
  447
  448
  449
  450
  451
  452
  453
  454
  455
  456
  457
  458
  459
  460
  461
  462
  463
  464
  465
  466
  467
  468
  469
  470
  471
  472
  473
  474
  475
  476
  477
  478
  479
  480
  481
  482
  483
  484
  485
  486
  487
  488
  489
  490
  491
  492
  493
  494
  495
  496
  497
  498
  499
  500
  501
  502
  503
  504
  505
  506
  507
  508
  509
  510
  511
  512
  513
  514
  515
  516
  517
  518
  519
  520
  521
  522
  523
  524
  525
  526
  527
  528
  529
  530
  531
  532
  533
  534
  535
  536
  537
  538
  539
  540
  541
  542
  543
  544
  545
  546
  547
  548
  549
  550
  551
  552
  553
  554
  555
  556
  557
  558
  559
  560
  561
  562
  563
  564
  565
  566
  567
  568
  569
  570
  571
  572
  573
  574
  575
  576
  577
  578
  579
  580
  581
  582
  583
  584
  585
  586
  587
  588
  589
  590
  591
  592
  593
  594
  595
  596
  597
  598
  599
  600
  601
  602
  603
  604
  605
  606
  607
  608
  609
  610
  611
  612
  613
  614
  615
  616
  617
  618
  619
  620
  621
  622
  623
  624
  625
  626
  627
  628
  629
  630
  631
  632
  633
  634
  635
  636
  637
  638
  639
  640
  641
  642
  643
  644
  645
  646
  647
  648
  649
  650
  651
  652
  653
  654
  655
  656
  657
  658
  659
  660
  661
  662
  663
  664
  665
  666
  667
  668
  669
  670
  671
  672
  673
  674
  675
  676
  677
  678
  679
  680
  681
  682
  683
  684
  685
  686
  687
  688
  689
  690
  691
  692
  693
  694
  695
  696
  697
  698
  699
  700
  701
  702
  703
  704
  705
  706
  707
  708
  709
  710
  711
  712
  713
  714
  715
  716
  717
  718
  719
  720
  721
  722
  723
  724
  725
  726
  727
  728
  729
  730
  731
  732
  733
  734
  735
  736
  737
  738
  739
  740
  741
  742
  743
  744
  745
  746
  747
  748
  749
  750
  751
  752
  753
  754
  755
  756
  757
  758
  759
  760
  761
  762
  763
  764
  765
  766
  767
  768
  769
  770
  771
  772
  773
  774
  775
  776
  777
  778
  779
  780
  781
  782
  783
  784
  785
  786
  787
  788
  789
  790
  791
  792
  793
  794
  795
  796
  797
  798
  799
  800
  801
  802
  803
  804
  805
  806
  807
  808
  809
  810
  811
  812
  813
  814
  815
  816
  817
  818
  819
  820
  821
  822
  823
  824
  825
  826
  827
  828
  829
  830
  831
  832
  833
  834
  835
  836
  837
  838
  839
  840
  841
  842
  843
  844
  845
  846
  847
  848
  849
  850
  851
  852
  853
  854
  855
  856
  857
  858
  859
  860
  861
  862
  863
  864
  865
  866
  867
  868
  869
  870
  871
  872
  873
  874
  875
  876
  877
  878
  879
  880
  881
  882
  883
  884
  885
  886
  887
  888
  889
  890
  891
  892
  893
  894
  895
  896
  897
  898
  899
  900
  901
  902
  903
  904
  905
  906
  907
  908
  909
  910
  911
  912
  913
  914
  915
  916
  917
  918
  919
  920
  921
  922
  923
  924
  925
  926
  927
  928
  929
  930
  931
  932
  933
  934
  935
  936
  937
  938
  939
  940
  941
  942
  943
  944
  945
  946
  947
  948
  949
  950
  951
  952
  953
  954
  955
  956
  957
  958
  959
  960
  961
  962
  963
  964
  965
  966
  967
  968
  969
  970
  971
  972
  973
  974
  975
  976
  977
  978
  979
  980
  981
  982
  983
  984
  985
  986
  987
  988
  989
  990
  991
  992
  993
  994
  995
  996
  997
  998
  999
  1000
  1001
  1002
  1003
  1004
  1005
  1006
  1007
  1008
  1009
  1010
  1011
  1012
  1013
  1014
  1015
  1016
  1017
  1018
  1019
  1020
  1021
  1022
  1023
  1024
  1025
  1026
  1027
  1028
  1029
  1030
  1031
  1032
  1033
  1034
  1035
  1036
  1037
  1038
  1039
  1040
  1041
  1042
  1043
  1044
  1045
  1046
  1047
  1048
  1049
  1050
  1051
  1052
  1053
  1054
  1055
  1056
  1057
  1058
  1059
  1060
  1061
  1062
  1063
  1064
  1065
  1066
  1067
  1068
  1069
  1070
  1071
  1072
  1073
  1074
  1075
  1076
  1077
  1078
  1079
  1080
  1081
  1082
  1083
  1084
  1085
  1086
  1087
  1088
  1089
  1090
  1091
  1092
  1093
  1094
  1095
  1096
  1097
  1098
  1099
  1100
  1101
  1102
  1103
  1104
  1105
  1106
  1107
  1108
  1109
  1110
  1111
  1112
  1113
  1114
  1115
  1116
  1117
  1118
  1119
  1120
  1121
  1122
  1123
  1124
  1125
  1126
  1127
  1128
  1129
  1130
  1131
  1132
  1133
  1134
  1135
  1136
  1137
  1138
  1139
  1140
  1141
  1142
  1143
  1144
  1145
  1146
  1147
  1148
  1149
  1150
  1151
  1152
  1153
  1154
  1155
  1156
  1157
  1158
  1159
  1160
  1161
  1162
  1163
  1164
  1165
  1166
  1167
  1168
  1169
  1170
  1171
  1172
  1173
  1174
  1175
  1176
  1177
  1178
  1179
  1180
  1181
  1182
  1183
  1184
  1185
  1186
  1187
  1188
  1189
  1190
  1191
  1192
  1193
  1194
  1195
  1196
  1197
  1198
  1199
  1200
  1201
  1202
  1203
  1204
  1205
  1206
  1207
  1208
  1209
  1210
  1211
  1212
  1213
  1214
  1215
  1216
  1217
  1218
  1219
  1220
  1221
  1222
  1223
  1224
  1225
  1226
  1227
  1228
  1229
  1230
  1231
  1232
  1233
  1234
  1235
  1236
  1237
  1238
  1239
  1240
  1241
  1242
  1243
  1244
  1245
  1246
  1247
  1248
  1249
  1250
  1251
  1252
  1253
  1254
  1255
  1256
  1257
  1258
  1259
  1260
  1261
  1262
  1263
  1264
  1265
  1266
  1267
  1268
  1269
  1270
  1271
  1272
  1273
  1274
  1275
  1276
  1277
  1278
  1279
  1280
  1281
  1282
  1283
  1284
  1285
  1286
  1287
  1288
  1289
  1290
  1291
  1292
  1293
  1294
  1295
  1296
  1297
  1298
  1299
  1300
  1301
  1302
  1303
  1304
  1305
  1306
  1307
  1308
  1309
  1310
  1311
  1312
  1313
  1314
  1315
  1316
  1317
  1318
  1319
  1320
  1321
  1322
  1323
  1324
  1325
  1326
  1327
  1328
  1329
  1330
  1331
  1332
  1333
  1334
  1335
  1336
  1337
  1338
  1339
  1340
  1341
  1342
  1343
  1344
  1345
  1346
  1347
  1348
  1349
  1350
  1351
  1352
  1353
  1354
  1355
  1356
  1357
  1358
  1359
  1360
  1361
  1362
  1363
  1364
  1365
  1366
  1367
  1368
  1369
  1370
  1371
  1372
  1373
  1374
  1375
  1376
  1377
  1378
  1379
  1380
  1381
  1382
  1383
  1384
  1385
  1386
  1387
  1388
  1389
  1390
  1391
  1392
  1393
  1394
  1395
  1396
  1397
  1398
  1399
  1400
  1401
  1402
  1403
  1404
  1405
  1406
  1407
  1408
  1409
  1410
  1411
  1412
  1413
  1414
  1415
  1416
  1417
  1418
  1419
  1420
  1421
  1422
  1423
  1424
  1425
  1426
  1427
  1428
  1429
  1430
  1431
  1432
  1433
  1434
  1435
  1436
  1437
  1438
  1439
  1440
  1441
  1442
  1443
  1444
  1445
  1446
  1447
  1448
  1449
  1450
  1451
  1452
  1453
  1454
  1455
  1456
  1457
  1458
  1459
  1460
  1461
  1462
  1463
  1464
  1465
  1466
  1467
  1468
  1469
  1470
  1471
  1472
  1473
  1474
  1475
  1476
  1477
  1478
  1479
  1480
  1481
  1482
  1483
  1484
  1485
  1486
  1487
  1488
  1489
  1490
  1491
  1492
  1493
  1494
  1495
  1496
  1497
  1498
  1499
  1500
  1501
  1502
  1503
  1504
  1505
  1506
  1507
  1508
  1509
  1510
  1511
  1512
  1513
  1514
  1515
  1516
  1517
  1518
  1519
  1520
  1521
  1522
  1523
  1524
  1525
  1526
  1527
  1528
  1529
  1530
  1531
  1532
  1533
  1534
  1535
  1536
  1537
  1538
  1539
  1540
  1541
  1542
  1543
  1544
  1545
  1546
  1547
  1548
  1549
  1550
  1551
  1552
  1553
  1554
  1555
  1556
  1557
  1558
  1559
  1560
  1561
  1562
  1563
  1564
  1565
  1566
  1567
  1568
  1569
  1570
  1571
  1572
  1573
  1574
  1575
  1576
  1577
  1578
  1579
  1580
  1581
  1582
  1583
  1584
  1585
  1586
  1587
  1588
  1589
  1590
  1591
  1592
  1593
  1594
  1595
  1596
  1597
  1598
  1599
  1600
  1601
  1602
  1603
  1604
  1605
  1606
  1607
  1608
  1609
  1610
  1611
  1612
  1613
  1614
  1615
  1616
  1617
  1618
  1619
  1620
  1621
  1622
  1623
  1624
  1625
  1626
  1627
  1628
  1629
  1630
  1631
  1632
  1633
  1634
  1635
  1636
  1637
  1638
  1639
  1640
  1641
  1642
  1643
  1644
  1645
  1646
  1647
  1648
  1649
  1650
  1651
  1652
  1653
  1654
  1655
  1656
  1657
  1658
  1659
  1660
  1661
  1662
  1663
  1664
  1665
  1666
  1667
  1668
  1669
  1670
  1671
  1672
  1673
  1674
  1675
  1676
  1677
  1678
  1679
  1680
  1681
  1682
  1683
  1684
  1685
  1686
  1687
  1688
  1689
  1690
  1691
  1692
  1693
  1694
  1695
  1696
  1697
  1698
  1699
  1700
  1701
  1702
  1703
  1704
  1705
  1706
  1707
  1708
  1709
  1710
  1711
  1712
  1713
  1714
  1715
  1716
  1717
  1718
  1719
  1720
  1721
  1722
  1723
  1724
  1725
  1726
  1727
  1728
  1729
  1730
  1731
  1732
  1733
  1734
  1735
  1736
  1737
  1738
  1739
  1740
  1741
  1742
  1743
  1744
  1745
  1746
  1747
  1748
  1749
  1750
  1751
  1752
  1753
  1754
  1755
  1756
  1757
  1758
  1759
  1760
  1761
  1762
  1763
  1764
  1765
  1766
  1767
  1768
  1769
  1770
  1771
  1772
  1773
  1774
  1775
  1776
  1777
  1778
  1779
  1780
  1781
  1782
  1783
  1784
  1785
  1786
  1787
  1788
  1789
  1790
  1791
  1792
  1793
  1794
  1795
  1796
  1797
  1798
  1799
  1800
  1801
  1802
  1803
  1804
  1805
  1806
  1807
  1808
  1809
  1810
  1811
  1812
  1813
  1814
  1815
  1816
  1817
  1818
  1819
  1820
  1821
  1822
  1823
  1824
  1825
  1826
  1827
  1828
  1829
  1830
  1831
  1832
  1833
  1834
  1835
  1836
  1837
  1838
  1839
  1840
  1841
  1842
  1843
  1844
  1845
  1846
  1847
  1848
  1849
  1850
  1851
  1852
  1853
  1854
  1855
  1856
  1857
  1858
  1859
  1860
  1861
  1862
  1863
  1864
  1865
  1866
  1867
  1868
  1869
  1870
  1871
  1872
  1873
  1874
  1875
  1876
  1877
  1878
  1879
  1880
  1881
  1882
  1883
 
```

I was finding too many security vulnerabilities!

I was finding too many security vulnerabilities!

I needed automation!

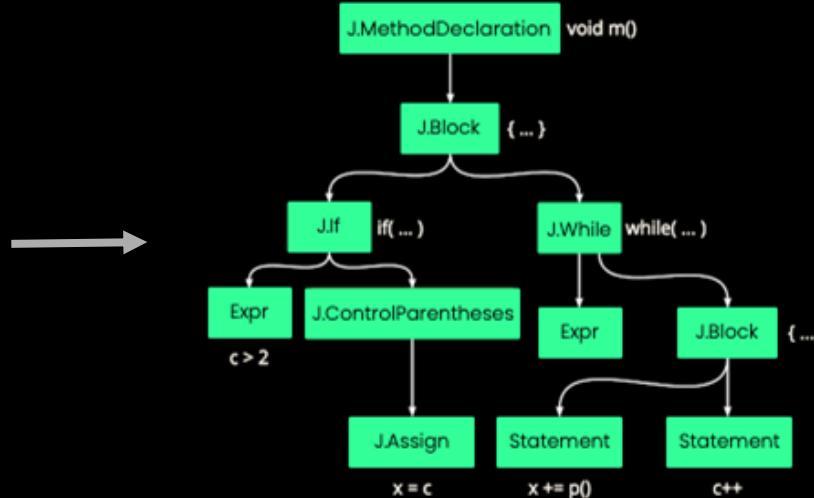
Automated Accurate Transformations at a Massive Scale



OpenRewrite

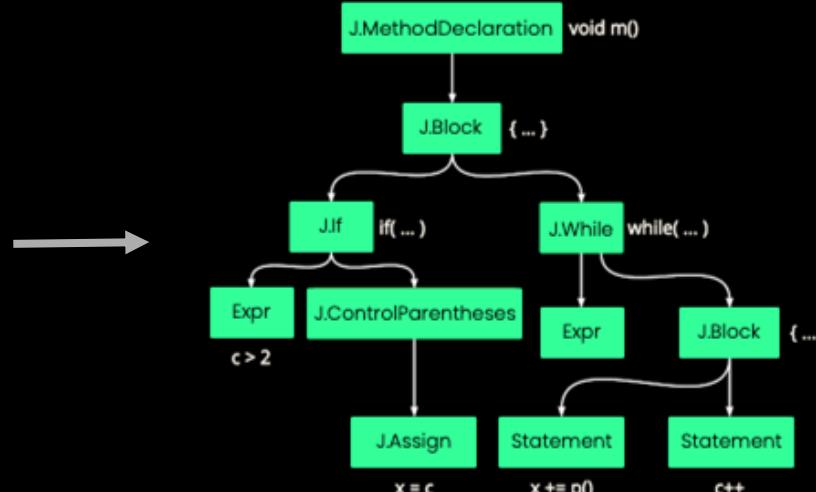
Abstract Syntax Tree (AST)

```
/** myMethod */
void m(){
    if (c > 2) {
        // c is more than 2
        x = c;
    }
    while(c < 10) { // increment x
        x += p();
        c++;
    }
}
```



Abstract Syntax Tree (AST)

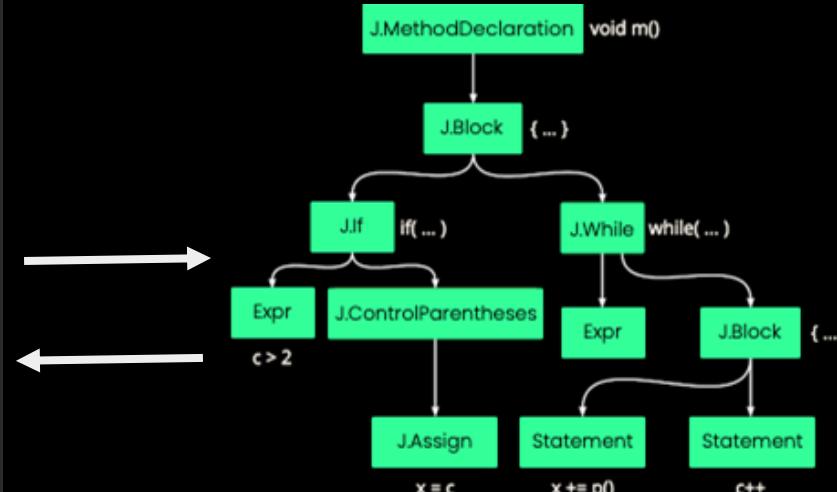
```
/** myMethod */
void m(){
    if (c > 2) {
        // c is more than 2
        x = c;
    }
    while(c < 10) { // increment x
        x += p();
        c++;
    }
}
```



```
void m(){if(c>2){x=c;}while(c<10){x+=p();c++;}}
```

Format Preserving AST

```
/** myMethod */
void m(){
    if (c > 2) {
        // c is more than 2
        x = c;
    }
    while(c < 10) { // increment x
        x += p();
        c++;
    }
}
```



Whitespace and comments are preserved

Generated code matches the Surrounding Formatting

Spaces

```
String name = entry.getName();
Path path = dir.resolve(name);
if (!path.normalize().startsWith(dir)) {
    throw new RuntimeException("Bad zip entry");
}
OutputStream os = Files.newOutputStream(path);
```

Tabs

```
Path path = dir.resolve(name);
if (!path.normalize().startsWith(dir)) {
    throw new RuntimeException("Bad zip entry");
}
OutputStream os = Files.newOutputStream(path);
```

Braces on new line

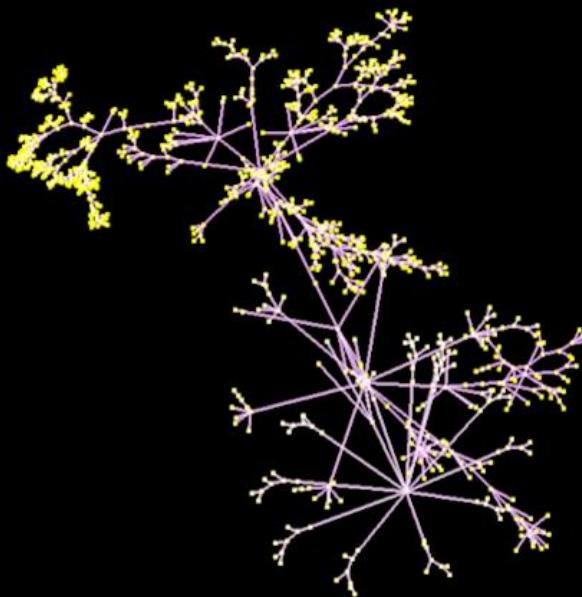
```
String name = entry.getName();
Path path = dir.resolve(name);
if (!path.normalize().startsWith(dir))
{
    throw new RuntimeException("Bad zip entry");
}
OutputStream os = Files.newOutputStream(path);
```

Accurate Transformations Require Fully Type-attributed ASTs

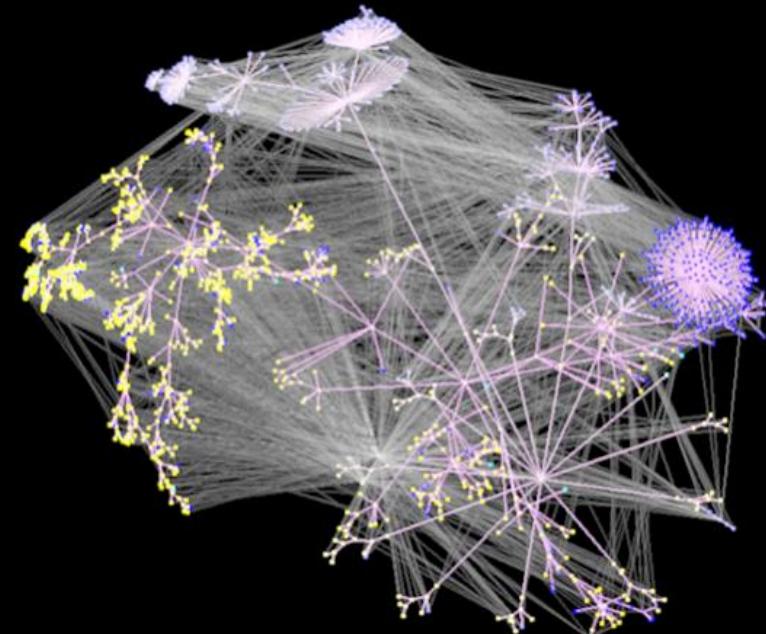
```
log.info("...");
```

Is that log4j, slf4j, LogBack?

The OpenRewrite AST is both Syntactically and Semantically aware.



Syntax alone



With type attribution and formatting

Even simple code produces complex AST

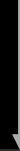


```
if (!path.normalize().startsWith(dir)) {  
    ... throw new RuntimeException("Bad zip entry");  
}
```

```
final JavaTemplate noZipSlipPathStartsWithPathTemplate =  
JavaTemplate.builder(this::getCursor, code: " " +  
"if (!#{any(java.nio.file.Path)}.normalize()" +  
".....".startsWith("#{any(java.nio.file.Path)})){\n" +  
".....throw new RuntimeException(\"Bad zip entry\");\n" +  
"}").build();
```

```
final JavaTemplate noZipSlipPathStartsWithPathTemplate =  
JavaTemplate.builder(this::getCursor, code: " " +  
"if (!#{any(java.nio.file.Path)}.normalize()" +  
" .startsWith(#{any(java.nio.file.Path)})) {\n" +  
"     throw new RuntimeException(\"Bad zip entry\");\n" +  
" }").build();  
  
return b.withTemplate(  
    noZipSlipPathStartsWithPathTemplate,  
    zipSlipSimpleInjectGuardInfo.statement.getCoordinates().after(),  
    zipSlipSimpleInjectGuardInfo.zipEntry,  
    zipSlipSimpleInjectGuardInfo.parentDir  
);
```

```
public class MyZipHelper {  
    public void m1(ZipEntry entry, Path dir) throws Exception {  
        String name = entry.getName();  
        Path path = dir.resolve(name);  
        OutputStream os = Files.newOutputStream(path);  
    }  
}
```



```
public class MyZipHelper {  
    public void m1(ZipEntry entry, Path dir) throws Exception {  
        String name = entry.getName();  
        Path path = dir.resolve(name);  
        if (!path.normalize().startsWith(dir)) {  
            throw new RuntimeException("Bad zip entry");  
        }  
        OutputStream os = Files.newOutputStream(path);  
    }  
}
```

What is possible now?

What other vulnerabilities can we fix?

Three Vulnerabilities

1. Temporary Directory Hijacking
2. Partial Path Traversal
3. Zip Slip

Vulnerability #1

Temporary Directory Hijacking

Temporary Directory on
Unix-Like Systems is
Shared between All Users

Temporary Directory Hijacking - Vulnerable

```
File f = File.createTempFile(  
    "prefix",  
    "suffix"  
) ;  
f.delete() ;  
f.mkdir() ;
```

ASK STACK OVERFLOW



GET VULNERABILITIES

imgflip.com

Temporary Directory Hijacking - Vulnerable

```
File f = File.createTempFile(  
    "prefix",  
    "suffix"  
) ;  
f.delete() ;  
f.mkdir() ;
```

Temporary Directory Hijacking - Vulnerable

```
File f = File.createTempFile(  
    "prefix",  
    "suffix"  
) ;  
f.delete() ;  
// 🏁 Race condition  
f.mkdir() ; // Returns `false`
```

Temporary Directory Hijacking - Imperfect Fix

```
File f = File.createTempFile(  
    "prefix",  
    "suffix"  
) ;  
f.delete() ;  
if (!f.mkdir())  
    throw new IOException("Error") ;
```

Temporary Directory Hijacking - Fix

```
// Since Java 1.7  
File f =  
    Files  
        .createTempDirectory("prefix")  
        .toFile();
```

Temporary Directory Hijacking - CVEs

- CVE-2022-27772 - Spring Boot
- CVE-2021-20202 - Keycloak
- CVE-2021-21331 - DataDog API
- CVE-2020-27216 - Eclipse Jetty
- CVE-2020-17521 - Apache Groovy
- CVE-2020-17534 - Apache netbeans-html4j

Temporary Directory Hijacking

Pull Request Statistics

Temporary Directory Hijacking

64 Pull Requests!

Temporary Directory Hijacking - Pull Requests

Temporary Directory Hijacking - Putting it all together

src/main/java/org/jenkinsci/backend/jpicreate/WebAppMain.java

@@ -10,6 +10,7 @@ org.openrewrite.java.security.UseFilesCreateTempDirectory

```
10  10      import javax.sound.midi.SysexMessage;
11  11      import java.io.File;
12  12      import java.io.IOException;
13  +13      + import java.nio.file.Files;
14  14
15  15      /**
16  * 
```

@@ -41,9 +42,7 @@

```
41  42          FileUtils.copyURLToFile(
42  43                  getClass().getClassLoader().getResource("maven.zip"),
43  44                  zip);
44  -          File bin = File.createTempFile("maven","bin");
45  -          bin.delete();
46  -          bin.mkdirs();
46  +          File bin = Files.createTempDirectory("maven" + "bin").toFile();
47  46
48  47          Process unzip = new ProcessBuilder("unzip", zip.getAbsolutePath())
49  48                  .directory(bin).redirectErrorStream(true).start();
```

Temporary Directory Hijacking - Putting it all together

The screenshot shows a code diff interface with the following details:

- File Path:** src/test/java/com/google/jenkins/plugins/credentials/oauth/JsonServiceAccountConfigTestUtil.java
- Diff Range:** @@ -22,6 +22,7 @@ org.openrewrite.java.security.UseFilesCreateTempDirectory
- Added Lines:**

```
22 22 import java.io.IOException;
23 23 import java.io.StringWriter;
24 24 import java.nio.charset.Charset;
25 +import java.nio.file.Files;
26 26 import java.security.KeyPair;
27 27 import java.security.KeyPairGenerator;
28 28 import java.security.NoSuchAlgorithmException;
```
- Deleted Lines:**

```
@@ -64,13 +65,7 @@
64 65
65 66     private static File getTempFolder() throws IOException {
66 67         if (tempFolder == null) {
67 -             tempFolder = File.createTempFile("temp", Long.toString(System.nanoTime()));
68 -             if (!tempFolder.delete()) {
69 -                 throw new IOException("Could not delete temp file: " + tempFolder.getAbsolutePath());
70 -             }
71 -             if (!tempFolder.mkdir()) {
72 -                 throw new IOException("Could not create temp directory: " + tempFolder.getAbsolutePath());
73 -             }
74 +             tempFolder = Files.createTempDirectory("temp" + Long.toString(System.nanoTime())).toFile();
75
76         }
77         tempFolder.deleteOnExit();
78
79         return tempFolder;
80     }
```

Vulnerability #2

Partial Path Traversal

Partial Path Traversal

```
" /user/sam"
```

Partial Path Traversal

```
" /user/sam "
```

```
" /user/samantha "
```

Partial Path Traversal

Allows an attacker access to a sibling directory with the same prefix

Partial Path Traversal

```
" /user/sam "
```

Allows an attacker access to a sibling directory with the same prefix

Partial Path Traversal

```
" /user/sam "
```

Allows an attacker access to a sibling directory with the same prefix

```
" /user/samantha "
```

Partial Path Traversal

```
" /user/sam "
```

Allows an attacker access to a sibling directory with the same prefix

```
" /user/samantha "
```

Partial Path Traversal - Vulnerability

```
File parent = parent();  
File dir = new File(parent, userControlled());  
  
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    throw new IOException(  
        "Detected path traversal attack!"  
    );  
}
```

```
new File("/user/sam/")
```

```
new File("/user/sam/")
```

File.getCanonicalPath()

```
new File("/user/sam/")
```

File.getCanonicalPath()

```
"/user/sam"
```

```
new File("/user/sam/')
```

File.getCanonicalPath()

```
" /user/sam"
```



Partial Path Traversal - Vulnerability

```
File parent = parent();  
File dir = new File(parent, userControlled());  
  
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    throw new IOException(  
        "Detected path traversal attack!"  
    );  
}
```

Partial Path Traversal - Vulnerability

```
File dir = new File(parent, userControlled());  
  
if (!dir.getCanonicalPath()  
    .startsWith("/user/sam")) {  
    ...  
}
```

Partial Path Traversal - Vulnerability

```
File parent = parent();  
File dir = new File(parent, userControlled());  
  
if (!dir.getCanonicalPath()  
    .startsWith("/user/sam")) {  
    ...  
}
```

Partial Path Traversal - Vulnerability

```
File dir = new File(parent, "../samantha/baz");  
  
if (!dir.getCanonicalPath()  
    .startsWith("/user/sam")) {  
    ...  
}
```

Partial Path Traversal - Vulnerability

```
File dir = new File(parent, "../../samantha/baz");  
  
if (!"/user/samantha/baz"  
    .startsWith("/user/sam")) {  
    ...  
}  
}
```

Partial Path Traversal - Vulnerability

```
File dir = new File(parent, "../../samantha/baz");  
  
if (!"/user/samantha/baz"  
    .startsWith("/user/sam")) {  
    throw new IOException(  
        "Detected path traversal attack!"  
    );  
}
```



Partial Path Traversal - Vulnerability

```
File parent = parent();  
File dir = new File(parent, userControlled());  
  
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    throw new IOException(  
        "Detected path traversal attack!"  
    );  
}
```

Partial Path Traversal Fix!

Partial Path Traversal - Vulnerability

```
File parent = parent();  
File dir = new File(parent, userControlled());  
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    throw new IOException(  
        "Detected path traversal attack!"  
    );  
}
```

Partial Path Traversal - Vulnerability

```
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    ...  
}
```

Partial Path Traversal - Fix #1

```
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath() +  
        File.separatorChar)) {  
    ...  
}
```

Partial Path Traversal - Fix #2

```
if (!dir.getCanonicalFile()  
    .toPath().startsWith(  
        parent.getCanonicalFile().toPath())) {  
    ...  
}
```

Partial Path Traversal - Fix #2 - Better

```
if (!dir.getCanonicalFile()  
    .toPath().startsWith(  
        parent.getCanonicalFile().toPath())) {  
    ...  
}
```



How do we find this vulnerability?

Partial Path Traversal - Vulnerability

```
File parent = parent();  
File dir = new File(parent, userControlled());  
  
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    throw new IOException(  
        "Detected path traversal attack!"  
    );  
}
```

Partial Path Traversal - Vulnerability

```
if (!dir.getCanonicalPath()  
    .startsWith(parent.getCanonicalPath())) {  
    ...  
}
```

Partial Path Traversal - Vulnerability

```
if (!dir.getCanonicalPath ()  
    .startsWith (parent.getCanonicalPath () ) ) {  
    ...  
}
```

Partial Path Traversal - Safe

```
if (!dir.getCanonicalPath ()  
    .startsWith (parent.getCanonicalPath () +  
    File.separatorChar) ) {  
    . . .  
}
```

It can't be that easy, can it?

Partial Path Traversal - Vulnerability

```
if (!dir.getCanonicalPath ()  
    .startsWith (parent.getCanonicalPath () ) ) {  
    ...  
}
```

Partial Path Traversal - Vulnerability

```
String dirCanonical = dir.getCanonicalPath();  
  
if (!dirCanonical  
    .startsWith(parent.getCanonicalPath())) {  
    ...  
}
```

Partial Path Traversal - Vulnerability

```
String dirCanonical = dir.getCanonicalPath();  
String pCanonical = parent.getCanonicalPath() +  
    File.separatorChar;  
if (!dirCanonical  
    .startsWith(pCanonical)) {  
    ...  
}
```

We need Data Flow Analysis

Partial Path Traversal - DataFlow

```
String dirCanonical = dir.getCanonicalPath();  
String pCanonical = parent.getCanonicalPath() +  
    File.separatorChar;  
  
if (!dirCanonical  
    .startsWith(pCanonical)) {  
    ...  
}
```

Partial Path Traversal - Data Flow

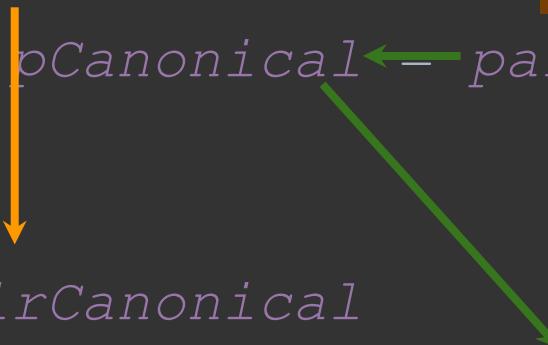
```
String dirCanonical←dir.getCanonicalPath();  
String pCanonical = parent.getCanonicalPath() +  
    File.separatorChar;  
  
if (!dirCanonical  
    .startsWith(pCanonical)) {  
    ...  
}
```

Partial Path Traversal - Data Flow

```
String dirCanonical←dir.getCanonicalPath();  
String pCanonical = parent.getCanonicalPath() +  
                    File.separatorChar;  
  
↓  
if (!dirCanonical  
    .startsWith(pCanonical)) {  
    ...  
}
```

Partial Path Traversal - Data Flow

```
String dirCanonical←dir.getCanonicalPath();  
String pCanonical←parent.getCanonicalPath() +  
File.separatorChar;  
  
if (!dirCanonical  
    .startsWith(pCanonical)) {  
    ...  
}
```



Partial Path Traversal - Data Flow

```
String dirCanonical←dir.getCanonicalPath();  
String pCanonical←parent.getCanonicalPath() +  
File.separatorChar;  
String pCanonical2←pCanonical;  
if (!dirCanonical  
    .startsWith(pCanonical2)) {  
    ...  
}
```

The diagram illustrates the data flow in the code. It starts with three assignments:

- A blue arrow points from `dir.getCanonicalPath()` to `dirCanonical`.
- A green arrow points from `parent.getCanonicalPath()` to `pCanonical`.
- A green arrow points from `File.separatorChar` to the addition operator (+) in the assignment of `pCanonical`.

Then, there is a blue arrow pointing from `pCanonical` to the assignment of `pCanonical2`. Finally, two green arrows point from `pCanonical2` to the check inside the if-block: one to the opening parenthesis and one to the comparison operator (`==`).

Data Flow

Uncovers hard to find Vulnerabilities
and prevents
False Positives

Data Flow Analysis

```
class GetCanonicalPathToStartsWithLocalFlow extends LocalFlowSpec<J.MethodInvocation, Expression> {

    @Override
    public boolean isSource(J.MethodInvocation methodInvocation, Cursor cursor) {
        return new MethodMatcher("java.io.File getCanonicalPath()")
            .matches(methodInvocation);
    }

    @Override
    public boolean isSink(Expression expression, Cursor cursor) {
        return InvocationMatcher
            .fromMethodMatcher(
                new MethodMatcher(
                    "java.lang.String startsWith(java.lang.String)"
                )
            )
            .advanced()
            .isSelect(cursor);
    }
}
```

Partial Path Traversal - Putting it all together

The screenshot shows a code diff interface with the following details:

- File: src/main/java/de/neemann/digital/draw/library/ElementLibrary.java
- Change Type: @@ -412,7 +412,7 @@ org.openrewrite.java.security.PartialPathTraversalVulnerability
- Diff Content:

```
412 412         try {
413 413             String root = rootLibraryPath.getCanonicalPath();
414 414             String path = file.getParentFile().getCanonicalPath();
415 415 -             return path.startsWith(root);
416 416 +             return file.getParentFile().getCanonicalFile().toPath().startsWith(root);
417 417         } catch (IOException e) {
418 418             return false;
}
```

Example Case: AWS Java SDK

CVE-2022-31159

[aws-sdk-java/aws-java-sdk-s3/src/main/java/com/amazonaws/services/s3/transfer/TransferManager.java](#)

Lines 1513 to 1519 in [5be0807](#)

```
1513     private boolean leavesRoot(File localBaseDirectory, String key) {
1514         try {
1515             return !new File(localBaseDirectory, key).getCanonicalPath().startsWith(localBaseDirectory.getCanonicalPath());
1516         } catch (IOException e) {
1517             throw new RuntimeException("Unable to canonicalize paths", e);
1518         }
1519     }
```

[aws-sdk-java/aws-java-sdk-s3/src/main/java/com/amazonaws/services/s3/transfer/TransferManager.java](#)
Lines 1420 to 1423 in ae88c8a

```
1420     if ( leavesRoot(destinationDirectory, s.getKey()) ) {  
1421         throw new RuntimeException("Cannot download key " + s.getKey() +  
1422             ", its relative path resolves outside the parent directory.");  
1423     }
```

Vulnerability #3

Zip Slip

Zip Slip

```
void zipSlip(File destination, ZipFile zip) {  
    Enumeration<? extends ZipEntry> entries = zip.entries();  
    while (entries.hasMoreElements()) {  
        ZipEntry e = entries.nextElement();  
        File f = new File(destination, e.getName());  
        IOUtils.copy(  
            zip.getInputStream(e),  
            new FileOutputStream(f)  
        );  
    }  
}
```

Zip Slip

```
ZipEntry e = entries.nextElement();  
File f = new File(destination, e.getName());  
IOUtils.copy(  
    zip.getInputStream(e),  
    new FileOutputStream(f)  
) ;
```

Zip Slip

```
ZipEntry e = entries.nextElement();  
File f = new File(destination, e.getName());  
IOUtils.copy(  
    zip.getInputStream(e),  
    new FileOutputStream(f)  
) ;
```

Zip Slip is Complicated

Zip Slip

```
File f = new File(destination, e.getName()) ;  
  
IOUtils.copy(  
    zip.getInputStream(e) ,  
    new FileOutputStream(f)  
) ;
```

Zip Slip

```
File f = new File(destination, e.getName()) ;
if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}

IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
```

The Problem with Zip Slip

Zip Slip

```
File f = new File(destination, e.getName()) ;
if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}

IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
```

Zip Slip

```
File f = new File(destination, e.getName()) ;
if (f.toPath().startsWith(destination.toPath())) {
    IOUtils.copy(
        zip.getInputStream(e),
        new FileOutputStream(f)
    );
}
```

Control Flow Analysis

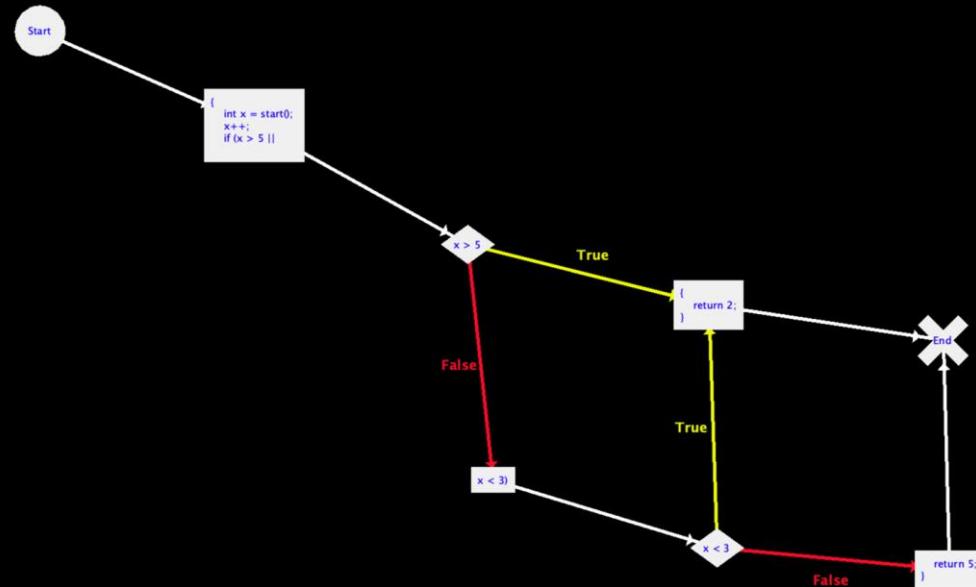
Control Flow Analysis

```
File f = new File(destination, e.getName());  
IOUtils.copy(  
    zip.getInputStream(e),  
    new FileOutputStream(f)  
) ;
```

```
File f = new File(destination, e.getName());  
if  
    (!f.toPath().startsWith(destination.toPath())){  
        throw new IOException("Bad Zip Entry!");  
    }  
IOUtils.copy(  
    zip.getInputStream(e),  
    new FileOutputStream(f)  
) ;
```

Control Flow - OpenRewrite

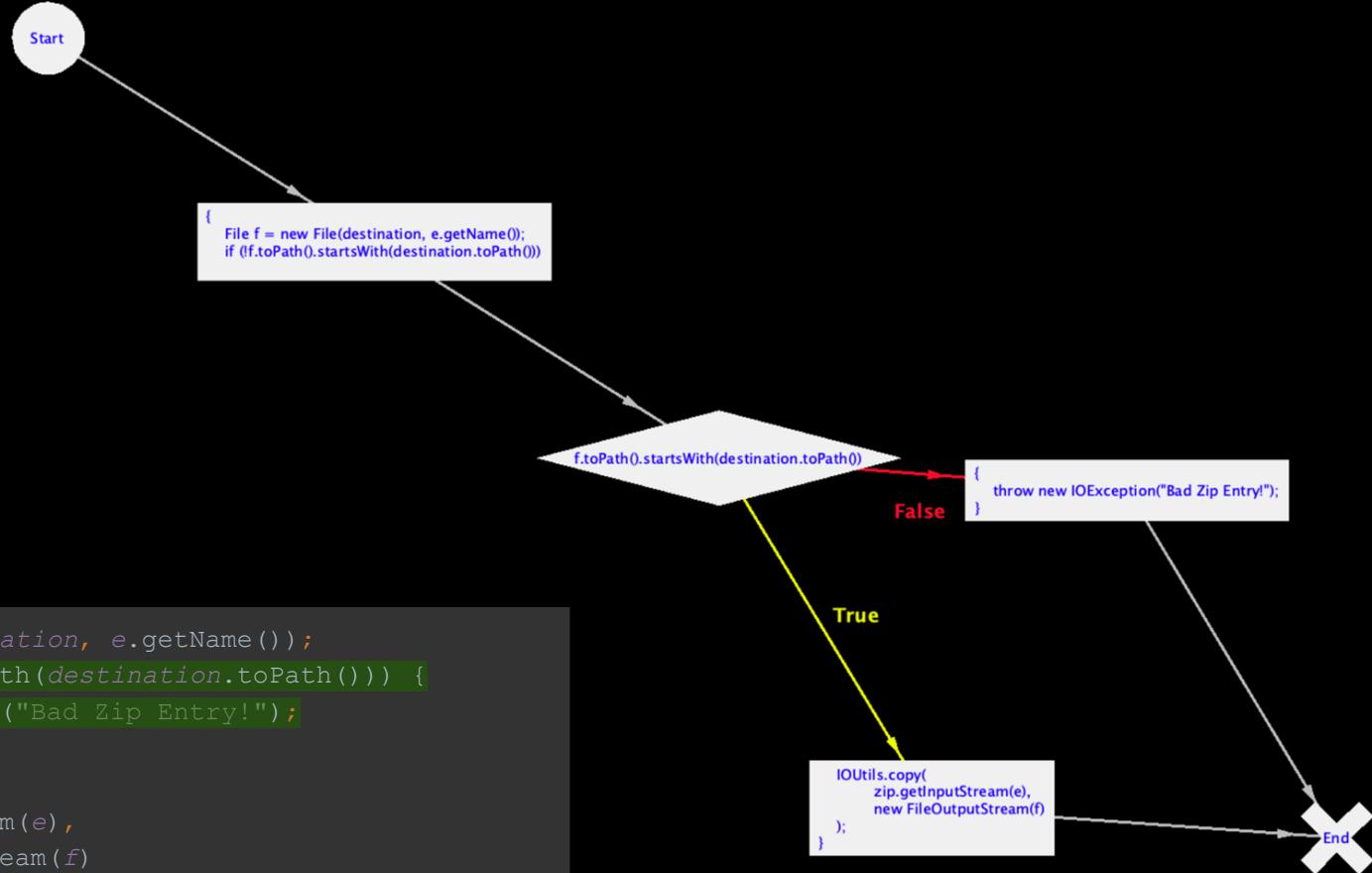
```
abstract class Test {  
    abstract int start();  
    int test() {  
        int x = start();  
        x++;  
        if (x > 5 || x < 3) {  
            return 2;  
        }  
        return 5;  
    }  
}
```



Zip Slip

```
File f = new File(destination, e.getName()) ;
if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}
IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
```

Zip Slip



Zip Slip - Putting it all together

src/main/java/org/owasp/webgoat/lessons/path_traversal/ProfileZipSlip.java

```
@@ -58,6 +58,9 @@ @@ org.openrewrite.java.security.ZipSlip
58  58          while (entries.hasMoreElements()) {
59  59                  ZipEntry e = entries.nextElement();
60  60                  File f = new File(tmpZipDirectory.toFile(), e.getName());
61 +                 if (!f.toPath().normalize().startsWith(tmpZipDirectory.toFile().toPath())) {
62 +                         throw new RuntimeException("Bad zip entry");
63 +                 }
61  64                  InputStream is = zip.getInputStream(e);
62  65                  Files.copy(is, f.toPath(), StandardCopyOption.REPLACE_EXISTING);
63  66 }
```

Zip Slip - Putting it all together

The screenshot shows a code diff interface with the following details:

- File:** jbake-core/src/main/java/org/jbake/app/ZipUtil.java
- Change Type:** @@ -28,7 +28,10 @@ org.openrewrite.java.security.ZipSlip
- Line 28:** byte[] buffer = new byte[1024];
- Line 29:** 29
- Line 30:** while ((entry = zis.getNextEntry()) != null) {
- Line 31:** - File outputFile = new File(outputFolder.getCanonicalPath() + File.separatorChar + entry.getName());
+ File outputFile = new File(outputFolder.getCanonicalPath(), entry.getName());
- Line 32:** + if (!outputFile.toPath().normalize().startsWith(outputFolder.getCanonicalPath())) {
- Line 33:** + throw new RuntimeException("Bad zip entry");
- Line 34:** }
- Line 35:** File outputParent = new File(outputFile.getParent());
- Line 36:** outputParent.mkdirs();

Pull Request Generation!

GOT SECURITY VULNERABILITIES?



YOU GET A PULL REQUEST!
YOU GET A PULL REQUEST!
EVERYBODY GETS A PULL REQUEST!!!

imgflip.com

Problems with Pull Request Generation

How fast can we generate Pull Requests?

Pull Request Generation Steps

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes
6. Create Pull Request on GitHub

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes
6. Create Pull Request on GitHub

File IO

Git Operation

GitHub API

Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes
6. Create Pull Request on GitHub

File IO

Git Operation

GitHub API

**IF YOU COULD STOP
RATE LIMITING YOUR API**

THAT WOULD BE GREAT

We've made it this far

-  Vulnerabilities Detected
-  Style Detected
-  Code Fixed & Diff Generated
-  Rate Limit Bypassed

We've made it this far

-  Vulnerabilities Detected
-  Style Detected
-  Code Fixed & Diff Generated
-  Rate Limit Bypassed

How do we do this for all the repositories?

Moderne

- Free for Open Source Projects!
- ~7,000 Repositories indexed
- Run Open Rewrite Transformations at Scale
- Generates and Update Pull Requests

800+ OpenRewrite Recipes including complete Framework Migrations

The screenshot displays the Moderne platform interface, which is a web-based tool for analyzing and modernizing code. The top navigation bar includes a logo, a search bar with placeholder text "Search recipes like 'java' or 'tags:testing'", and a search icon. The left sidebar features a vertical menu with icons for different categories: Analyze your code, Modernize your code, Java recipes, Spring recipes, Kubernetes recipes, Maven recipes, and a link to "... and much, much more".

Analyze your code

- [Find method usages](#) Find method usages by pattern.
- [Change method name](#) Rename a method.
- [Find types](#) Find type references by name.
- [Find missing configuration](#) Find Kubernetes resources with missing configuration.

Modernize your code

Java recipes »

- [Migrate Java 8 to Java 11](#) This recipe will apply changes commonly needed when migrating...
- [Java security best practices](#) Applies security best practices to Java code.
- [Format Java code](#) Format Java code using a standard comprehensive set of Java formattin...
- [Migrate JUnit asserts to AssertJ](#) AssertJ provides a rich set of assertions, truly helpful error...

Spring recipes »

- [Spring Boot 2.x migration from Spring Boot 1.x](#) Migrates Spring Boot 1.x to 2.x including best practices.
- [Spring Boot 2.x best practices](#) Applies best practices to Spring Boot 2 applications.
- [JUnit Jupiter for Spring Boot 2.x projects](#) Migrates Spring Boot 2.x projects having JUnit 4 tests to JUnit Jupiter.
- [Remove @RequestMapping annotations](#) Replace method declaration @RequestMapping annotations with...

Kubernetes recipes »

- [Kubernetes best practices](#) Applies best practices to Kubernetes manifests.
- [Ensure liveness probe is configured](#) The kubelet uses liveness probes to know when to schedule restarts for...
- [Ensure readiness probe is configured](#) Using the Readiness Probe ensures teams define what actions need to b...
- [Cap exceeds resource value](#) Cap resource values that exceed a specific maximum.

Maven recipes »

- [Manage dependencies](#) Make existing dependencies managed by moving their version to...
- [Maven dependency insight](#) Find direct and transitive dependencies matching a group,...
- [Remove redundant explicit dependency versions](#) Remove explicitly-specified dependency versions when a parent...
- [Upgrade Maven dependency version](#) Upgrade the version of a dependency by specifying a group or group and...

[... and much, much more »](#)

Bulk Pull Request Generation - public.moderne.io

Moderne Search recipes like 'java' or 'tags:testing'

Home > Recent commits > Commit job 1cd92655-ecfd-497f-b51f-94dcb47ef895

Commit results

Recipe: **sulAO** Success: 93% Progress: 100% Started: 1 day ago

Commit title: **vuln-fix: Use HTTPS instead of HTTP to resolve dependencies**

Commit messages:

This fixes a security vulnerability in this project where the `build.gradle` files were configuring Gradle to resolve dependencies over HTTP instead of HTTPS.

Weakness: CWE-829: Inclusion of Functionality from Untrusted Control Sphere Severity: High CVSS: 8.1 Detection: OpenRewrite

Reported-by: Jonathan Leitschuh Jonathan.Leitschuh@gmail.com Signed-off-by: Jonathan Leitschuh Jonathan.Leitschuh@gmail.com

Bug-tracker: <https://github.com/JLLeitschuh/security-research/issues/9>

Search... RERUN FAILED JOBS

Status	Repository	Modified	Result
No changes	lucene-gosen/lucene-gosen	about 22 hours ago	
Completed	sonalake/swagger-changelog-gradle-plugin	1 day ago	View commit
Completed	SmartReceipts/SmartReceiptsLibrary	1 day ago	View commit
Completed	jmad/jmad-core	1 day ago	View commit
Completed	sitewhere/sitewhere	1 day ago	View commit
Completed	nining377/UnblockMusicPro_Xposed	1 day ago	View commit
Completed	Mocha-L/QuJing	1 day ago	View commit

sonalake / swagger-changelog-gradle-plugin Public

Code Issues 1 Pull requests 1 Actions Projects Wiki Security Insights

[SECURITY] Use HTTPS to resolve dependencies in Gradle Build #13

JLLeitschuh wants to merge 1 commit into **sonalake:master** from **JLLeitschuh:fix/3LL/use_https_to_resolve_dependencies_gradle**

Conversation 0 Commits 1 Checks 0 Files changed 1

JLLeitschuh commented 6 days ago

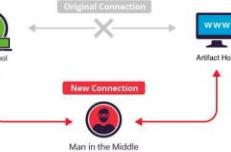
sitewhere / sitewhere Public

[SECURITY] Use HTTPS to resolve dependencies in Gradle Build #982

JLLeitschuh wants to merge 1 commit into **sitewhere:master** from **JLLeitschuh:fix/3LL/use_https_to_resolve_dependencies_gradle**

Conversation 0 Commits 1 Checks 0 Files changed 1

JLLeitschuh commented 6 days ago

Build Tool 

This is a security fix. The build files indicate this project is resolving dependencies over HTTP instead of HTTPS. This leaves your build vulnerable to allowing a Man in the Middle (MitM) attackers to execute arbitrary code on your or your computer or CI/CD system.

This vulnerability has a CVSS v3.0 Base Score of 8.1/10. POC code has existed since 2014 to maliciously compromise a JAR file in-flight. MitM attacks against HTTP are increasingly common, for example Conjur is known to have done it to their own users.

Code Issues 1 Pull requests 1 Actions Projects 2 Wiki Security Insights

Watch 3 Fork 371 Star 889

Reviewers No reviewers

Assignees No one assigned

Labels None yet

Projects None yet

Milestone None milestone

Development Successfully merging this pull request may close these issues. None yet

Notifications Customize Subscribe You're not receiving notifications from this thread.

1 participant

Use Files#createTempDirectory

SUMMARY

Status	Started	Estimated time savings	Repositories searched	Repositories changed	Files searched	Files changed
Finished	about 2 hours ago	17 hours, 50 minutes	6.21k	80	1.26M	107

Search...

Hide no results

SELECT ALL WITH RESULTS

<input type="checkbox"/>	Status	Repository	Branch	Total results	Files searched	Actions
<input type="checkbox"/>	Finished	broadinstitute/picard	master	4	6,501	DIFF ▾
<input type="checkbox"/>	Finished	jenkinsci/google-oauth-plugin	develop	3	218	DIFF ▾
<input type="checkbox"/>	Finished	andyglick/jenkins4java	master	3	279	DIFF ▾
<input type="checkbox"/>	Finished	salesforce/ImageOptimization	master	3	244	DIFF ▾
<input type="checkbox"/>	Finished	sonatype/plexus-archiver	master	3	580	DIFF ▾
<input type="checkbox"/>	Finished	zeroturnaround/zt-zip	master	3	230	DIFF ▾
<input type="checkbox"/>	Finished	jenkinsci/acceptance-test-harness	master	3	786	DIFF ▾
<input type="checkbox"/>	Finished	jenkinsci/jacoco-plugin	master	3	333	DIFF ▾
<input type="checkbox"/>	Finished	smacke/jaydio	master	2	154	DIFF ▾
<input type="checkbox"/>	Finished	jaltekruse/OpenNotebook	master	2	438	DIFF ▾
<input type="checkbox"/>	Finished	nativefibs4java/BridJ	master	2	627	DIFF ▾
<input type="checkbox"/>	Finished	jenkinsci/backend-jpi-create	master	2	134	DIFF ▾
<input type="checkbox"/>	Finished	libgdx/libgdx	master	2	3,718	DIFF ▾
<input type="checkbox"/>	Finished	koraktor/mavanagaiata	master	2	75	DIFF ▾
<input type="checkbox"/>	Finished	Graylog2/JadConfig	master	2	357	DIFF ▾
<input type="checkbox"/>	Finished	jbossas/jboss-vfs	master	2	279	DIFF ▾



Home > Recent commits > Commit job 1449e2d1-7e24-4d78-9798-ba06caa1c1a2

Commit results

Recipe	Success	Progress	Started
e52VD	80%	100%	about 6 hours ago

Commit title

vuln-fix: Temporary Directory Hijacking or Information Disclosure

Commit messages

This fixes either Temporary Directory Hijacking, or Temporary Directory Local Information Disclosure.

Weakness: CWE-379: Creation of Temporary File in Directory with Insecure Permissions Severity: High CVSS: 7.3 Detection: CodeQL & OpenRewrite (<https://public.moderne.io/recipes/org.openrewrite.java.security.UseFilesCreateTempDirectory>)

Reported-by: Jonathan Leitschuh Jonathan.Leitschuh@gmail.com Signed-off-by: Jonathan Leitschuh Jonathan.Leitschuh@gmail.com

Bug-tracker: <https://github.com/JLLeitschuh/security-research/issues/10>

Search...

RERUN FAILED JOBS

Status	Repository	Modified	Result
COMPLETED	sanity/tahir	about 6 hours ago	View commit
COMPLETED	broadinstitute/picard	about 6 hours ago	View commit
COMPLETED	Anuken/Arc	about 5 hours ago	View commit
COMPLETED	talsma-ict/umidoclet	about 5 hours ago	View commit
COMPLETED	jenkinsci/jenkins-test-harness	about 5 hours ago	View commit
COMPLETED	searls/jasmine-maven-plugin	about 5 hours ago	View commit
COMPLETED	vertx-3/vertx-amqp-bridge	about 5 hours ago	View commit
COMPLETED	reactor/reactor-netty	about 5 hours ago	View commit
COMPLETED	libgdx/libgdx	about 5 hours ago	View commit
COMPLETED	Karlatemp/PublicationSign	about 5 hours ago	View commit

But there are more than just 7,000
repositories in the world

How do we find the other vulnerable projects?

CodeQL

CodeQL

100k+ OSS Projects Indexed
35k+ OSS Java Projects

<https://github.com/moderneinc/jenkins-ingest>

main · [jenkins-ingest / repos.csv](#) · [Raw](#) · [Blame](#) · [Edit](#) · [Delete](#)

 tkvangorder Fixing issues with requested repos · Latest commit c6d166d 3 days ago · [History](#)

7 contributors 

9877 lines (9877 sloc) | 433 KB

Search this file...

1	Offz/gpr-for-gradle	master	8		gradle
2	Opslab/opslabUtil	master	8		maven
3	105032013072/javaparser	master	8		maven
4	15189611/jumpAop	master	8		gradlew
5	18824863285/BaseFlutter	master	8		gradlew
6	1and1/cosmo	master	8		maven
7	1and1/reactive	master	8		maven
8	1c-syntax/bslls-dev-tools	develop	8		gradlew
9	275593469/study	master	8		maven
10	2dxgjun/AndroidTagGroup	master	8		gradlew
11	2pure/CodeDesign-Homework1	master	8		maven
12	38leinaD/tdd-gradle-plugin	master	8		gradlew
13	3esi/dotnet-plugin	master	8		gradle
14	3esi/gitversion-plugin	master	8		gradle

Finally!

Let's generate some
Open Source Software
Pull Requests!

Bulk Pull Request Generation Statistics

Project	PR Generator	Pull Requests	Merge Rate
HTTP Download of Dependencies	Python Bot	1,596	40%
CVE-2019-16303: JHipster RNG Vulnerability	Python Bot + Moderne	3,467	2.3%
CVE-2020-8597: rhostname array overflow	Python Bot	1,885	7.6%
Temporary Directory Hijacking	Moderne	64	TBD
Partial Path Traversal	Moderne	24	TBD
Zip Slip	Moderne	86	TBD

Bulk Pull Request Generation Statistics

Project	PR Generator	Pull Requests	Merge Rate
HTTP Download of Dependencies	Python Bot	1,596	40%
CVE-2019-16303: JHipster RNG Vulnerability	Python Bot + Moderne	3,467	2.3%
CVE-2020-8597: rhostname array overflow	Python Bot	1,885	7.6%
Temporary Directory Hijacking	Moderne	64	TBD
Partial Path Traversal	Moderne	24	TBD
Zip Slip	Moderne	86	TBD

New Pull Requests Generated in 2022: 570+

Bulk Pull Request Generation Statistics

Project	PR Generator	Pull Requests	Merge Rate
HTTP Download of Dependencies	Python Bot	1,596	40%
CVE-2019-16303: JHipster RNG Vulnerability	Python Bot + Moderne	3,467	2.3%
CVE-2020-8597: rhostname array overflow	Python Bot	1,885	7.6%
Temporary Directory Hijacking	Moderne	64	TBD
Partial Path Traversal	Moderne	24	TBD
Zip Slip	Moderne	86	TBD

Personally Generated: 5,200+ Pull Requests



zeroturnaround / zt-zip

Public



Watch



Fork



Star 1.3k



<> Code

Issues 27

Pull requests 3

Actions

Projects

Security

Insights

Filters

Q is:pr is:open sort:updated-desc

Labels 2

Milestones 1

New pull request

Clear current search query, filters, and sorts

✓ 3 Open 37 Closed Merged

Open all Author ▾ Label ▾ Projects ▾ Milestones ▾ Reviews ▾ Assignee ▾ Sort ▾

[SECURITY] Fix Zip Slip Vulnerability

#149 opened 2 minutes ago by

[SECURITY] Fix Partial Path Traversal Vulnerability

#148 opened 6 hours ago by updated 6 hours ago

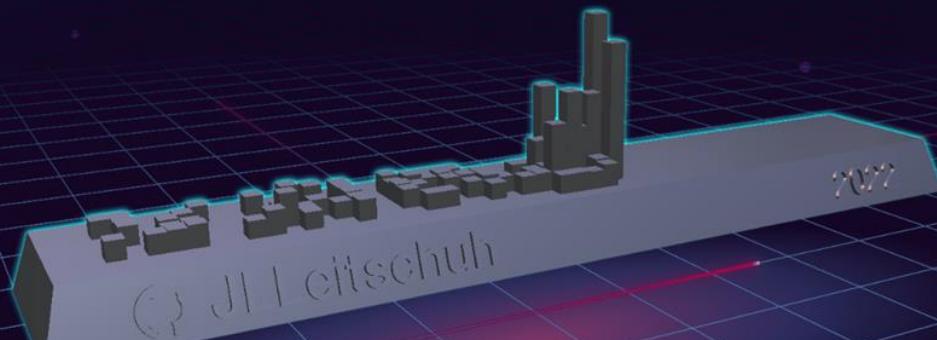
[SECURITY] Fix Temporary Directory Hijacking or Information Disclosure Vulnerability

#147 opened 2 days ago by

💡 **ProTip!** Find all pull requests that aren't related to any open issues with `-linked:issue`.



@JLLeitschuh's 2022 GitHub Skyline



Best Practices for Bulk Pull Request Generation

Messaging!

All Software Problems are
People Problems
In Disguise

Lesson 1

Sign off all Commits

--signoff

Sign off on Commits

Signed-off-by: Jonathan Leitschuh <Jonathan.Leitschuh@gmail.com>

Sign off on Commits

Why?!

Sign off on Commits

“It was introduced in the wake of the SCO lawsuit, (and other accusations of copyright infringement from SCO, most of which they never actually took to court), as a Developers Certificate of Origin. It is used to say that you certify that you have created the patch in question, or that you certify that to the best of your knowledge, it was created under an appropriate open-source license, or that it has been provided to you by someone else under those terms.”

- [Stack Overflow](#)

TL;DR

Lawyers

Lesson 2

Be a good commitizen

Lesson 2

Be a good commitizen

GPG Sign your Commits

-o-



vuln-fix: Use HTTPS instead of HTTP to resolve deps CVE-2021-26291

...

Verified

be60905

Lesson 3

There are risks using your personal GitHub Account

Anyone here familiar with
GitHub's
Angry Unicorn?



This page is taking way too long to load.

Sorry about that. Please try refreshing and contact us if the problem persists.

[Contact Support](#) — [GitHub Status](#) — [@githubstatus](#)



This was my GitHub Profile Page for most of 2020

Lesson 4

Coordinate with GitHub

Before Attempting

Reach out to GitHub!

SecurityLab@github.com

Lesson 5

Consider the Implications



Is this responsible disclosure?

5

#11 opened 4 hours ago

🕒 updated 35 minutes ago

Conclusion

As Security Researchers

We have an obligation to society

We know these vulnerabilities are out there

“For every 500 developers
you have one security
researcher.”

- GitHub 2020

“We can fix it. We have the technology. OK. We need to create the technology. Alright. The policy guys are mucking with the technology. Relax. WE'RE ON IT.

- Dan Kaminsky (1979 – 2021)

Black Hat Sound Bytes

- Learn CodeQL! Seriously! It's an incredibly powerful language!
- Contribute to OpenRewrite! Deploy your security fixes at scale!
- Join the GitHub Security Lab & OpenRewrite Slack Channels!

Thanks



HUMAN
Moderne

Lidia Giuliano

Shyam Mehta