A RULE SCOPE

Table 1. Issues reported by the code analysis, and their respective issue categories. We are actively labeling and enlarging our dataset.

Issue Identifier	Issue Category	# Datapoints	
		Train	Test
AmbiguousConditional	AST	150	58
AngularSceProviderDisabled	AST	2	1
ArgumentsAsParameter	AST	2	3
ArrayConstructor	AST	88	76
AssignmentWithSameVarOnLeftAndRight	AST	8	6
BooleanObjectCreation	AST	31	7
CommaOrSwitch	AST	18	4
ConditionWithAssign	AST	40	15
ContentLengthInCode	AST	2	1
DateMonthDecember	AST	15	6
DuplicateCaseBody	AST	2	1
DuplicateIfBody	AST	2	1
DuplicateObjectProperty	AST	6	1
DuplicateVueProperty	AST	5	1
EmberInitializerDeprecation	AST	4	1
FirefoxImageNonStandard	AST	2	1
FunctionDeclarationInBlock	AST	2	1
HttpToHttps	AST	4	3
MemberExpressionTypo	AST	18	2
NonLocalLoopVar	AST	2	1
ObjectConstructor	AST	35	30
OperatorPrecedence	AST	78	20
ReactApiTypo	AST	80	12
ReactDeprecatedElementProp	AST	2	1
ReactLegacyLifecycleMethod	AST	2	1
ReactMissingArrayKeys	AST	117	42
ReactPropsInGetDefault	AST	2	1
ReactSetInnerHtml	AST	3	2
ReactThisInFunctionComponent	AST	2	1
RedeclarationVars	AST	3	1
RegExpBadCharRange	AST	31	4
RegExpStringInReplace	AST	8	4
RepeatFuncArg	AST	3	2
ShiftOverflow	AST	9	1
SuperDuplicated	AST	7	1
UpdateApi	AST	4	1
UseArrowFunction	AST	3	1
UseGetFullYear	AST	2	1
UseLowercaseTagsForXHTML	AST	8	2

(table continued)

(table continued)

Issue Identifier	Issue Category	# Datapoints	
		Train	Test
UseStrictWrong	AST	2	1
Utf8Literal	AST	53	6
VarDeclConflict	AST	2	2
Stats for AST: # Issues: 42		**859**	**327**
ArrayMethodOnNonArray	Local	4	5
AssignToExports	Local	80	8
BadOperandForBitwiseOperation	Local	54	18
BadWrapperObjectCreation	Local	11	2
BitwiseOperationSignChecked	Local	2	1
CallbackShouldReturn	Local	2	1
CollectionArraySizeMatch	Local	19	16
CollectionUpdatedButNeverQueried	Local	2	1
CompareTypeofToString	Local	2	1
ComparisonToNaN	Local	4	2
ConfusedRegex	Local	9	3
ContentTypeNoCharset	Local	2	2
CopyPasteError	Local	3	1
DateMonthIndex	Local	6	1
DeleteOfNonProperty	Local	3	1
DuplicateCaseSwitch	Local	3	1
EqualityMisplacedParentheses	Local	5	1
ExceptionIsNotThrown	Local	4	1
ForEachReturns	Local	2	1
GlobalReplacementRegex	Local	3	1
ImplicitCoercionInOperator	Local	28	4
InOperatorBadLHS	Local	3	1
IncompatibleTypesInComparison	Local	5	1
IncompleteRegex	Local	14	2
IncorrectHtmlEscape	Local	28	9
LoopConditionLengthMissing	Local	63	21
MissingApiCallGet	Local	2	1
MissingApiCallReject	Local	3	1
MissingClose	Local	27	5
MissingCloseOnSomePath	Local	2	1
NoEffectExpression	Local	7	1
NoZeroReturnedInSort	Local	4	1
NonYieldingGenerator	Local	3	1
OverwriteAssignment	Local	2	1
PrimitiveInstanceOf	Local	11	3
PromiseNotCaughtNode	Local	3	1
RHSPrimitiveType	Local	22	13

(table continued)

(table continued)

Issue Identifier	Issue Category	# Datapoints	
		Train	Test
ReactContextTypes	Local	2	1
ReactForwardPropsToSuper	Local	24	16
ReactIdentifierTypo	Local	41	16
ReactIncorrectReturnValue	Local	6	4
ReactInvalidEventHandlerType	Local	3	1
ReactMissingEventHandlerCleanup	Local	3	2
ReactReservedPropsUsed	Local	4	2
ReactShouldConstructProps	Local	61	25
ReactThisInStaticLifecycleMethod	Local	2	2
ReactWrongStyleProperty	Local	50	40
ReactWrongStyleType	Local	31	42
ReplaceWithSameString	Local	48	35
ReplacementRegex	Local	2	1
RequireAsConstructor	Local	3	1
ThisBeforeSuper	Local	31	9
TypeofNotComparedToAnything	Local	5	1
UnusedLoopVar	Local	4	1
UsageOfUninitializedVariable	Local	98	35
UseIsArrayToCheckForArrays	Local	2	2
UseNumberIsNan	Local	3	1
WrongComparisonOperatorInSort	Local	2	1
WrongCsrfTokenHeader	Local	2	1
Stats for Local: # Issues: 59		**874**	**374**
BadAwaitExpression	FileWide	14	5
ExpectsObjectDislikesPrimitive	FileWide	2	1
PureFunctionReturnValueIgnored	FileWide	3	1
PureMethodReturnValueIgnored	FileWide	5	2
ReactControlledUncontrolledFormElement	FileWide	29	28
ReactMissingCleanup	FileWide	59	12
ReactModifyState	FileWide	20	8
ReactTypeCreatorUsedAsType	FileWide	38	12
ReactUnusedSnapshot	FileWide	2	1
ReactWrongApiArgumentType	FileWide	2	4
UseInstead	FileWide	7	1
WrongNumberOfArguments	FileWide	3	1
Stats for FileWide: # Issues: 12		**184**	**76**
DisablePoweredBy	SecurityLocal	367	142
ElectronInsecureWebPreferences	SecurityLocal	38	48
ElectronLoadInsecureContent	SecurityLocal	44	25
HardcodedNonCryptoSecret	SecurityLocal	37	28
HardcodedSecret	SecurityLocal	44	51

(table continued)

(table continued)

Issue Identifier	Issue Category	# Datapoints	
		Train	Test
InsecureCipherNoIntegrity	SecurityLocal	24	5
InsecureECB	SecurityLocal	7	2
InsecureHash	SecurityLocal	56	27
InsecureTLSConfig	SecurityLocal	77	8
InsufficientPostmessageValidation	SecurityLocal	14	3
IntrospectionEnabled	SecurityLocal	7	9
LimitGraphqlDepth	SecurityLocal	5	2
LoopDOS	SecurityLocal	3	6
NoCryptoTimingAttacks	SecurityLocal	2	1
NoHardcodedCredentials	SecurityLocal	38	21
NoHardcodedPasswords	SecurityLocal	31	21
NodeBufferNoOffset	SecurityLocal	3	1
TooPermissiveCorsHeader	SecurityLocal	5	9
TooPermissiveCorsPostMessage	SecurityLocal	10	7
TooSmallRsaKeySizeUsed	SecurityLocal	3	1
UseCsurfForExpress	SecurityLocal	46	30
UseHelmetForExpress	SecurityLocal	36	18
UseSecureWebsockets	SecurityLocal	24	27
WebCookieHttpOnlyDisabledByDefault	SecurityLocal	14	12
WebCookieHttpOnlyDisabledExplicitly	SecurityLocal	12	6
WebCookieSecureDisabledByDefault	SecurityLocal	54	26
WebCookieSecureDisabledExplicitly	SecurityLocal	8	3
Stats for SecurityLocal: # Issues: 27		**1009**	**539**
CodeInjection	SecurityFlow	50	67
CommandInjection	SecurityFlow	45	23
DOMXSS	SecurityFlow	81	51
FormatString	SecurityFlow	2	2
HTTPSourceWithUncheckedType	SecurityFlow	33	21
IndirectCommandInjection	SecurityFlow	11	7
NoRateLimitingForExpensiveWebOperation	SecurityFlow	37	9
NoRateLimitingForLogin	SecurityFlow	4	10
NoSqli	SecurityFlow	7	4
OR	SecurityFlow	38	57
PT	SecurityFlow	36	18
PrototypePollution	SecurityFlow	3	2
ServerLeak	SecurityFlow	61	46
Sqli	SecurityFlow	43	65
XSS	SecurityFlow	128	95
reDOS	SecurityFlow	27	25
Stats for SecurityFlow: # Issues: 16		**606**	**502**
Total stats: # Issues: 156		**3532**	**1818**

B PROMPTS FOR GPT-3.5 & GPT-4

Let RULE and DESCRIPTION denote the name and the description of the issue reported by the static analyzer in a code snippet. We will query the model to generate a fix for this code snippet and denote it by $\mathtt{CODE}^{q}_{\mathtt{PRE}}$. Let f be the number of few-shot examples provided in the prompt and the pair of code snippets ($\mathtt{CODE}^{i}_{\mathtt{PRE}}$, $\mathtt{CODE}^{i}_{\mathtt{POST}}$) denote the ith example fix for the issue RULE in the prompt.

GPT-3.5 and GPT-4 were designed to make conversations and we followed the best practices [**?**] shared by OpenAI to build the initial conversation. A conversation can contain three different roles, namely system, user and assistant. It is advised to start the conversation with a system content where one defines the role of the assistant (AI model) and gives instructions on the desired output structure. After that, we provide the few-shot examples as a conversation between the user and the assistant. The conversation is finished by the user providing the vulnerable code $\mathtt{CODE}^{q}_{\mathtt{PRE}}$ so that the next turn belongs to the assistant. The asistant completes the conversation by generating the fix to the last user query. Precisely, the following prompt is fed into GPT models. (The last sentence of the system prompt was not provided when the full file was fed into the model.)

SYSTEM: Assistant is a code assistant designed to fix issues in given code snippets. Instructions: Do not generate additional text or code. Output only the fixed code snippet. Do not generate explanations, comments, notes. Note that the code we provide is incomplete, it is intentionally reduced to a smaller snippet, do not try to complete it in anyway. Leave evertything as it is and just apply the changes related to the fix.

USER: Generate the fixed code for the bug RULE with the error message DESCRIPTION. $\mathtt{CODE}^{1}_{\mathtt{PRE}}$

ASSISTANT: $\mathtt{CODE}^{1}_{\mathtt{POST}}$

. . .

USER: Generate the fixed code for the bug RULE with the error message DESCRIPTION. $\mathtt{CODE}^{f}_{\mathtt{PRE}}$

ASSISTANT: $\mathtt{CODE}^{f}_{\mathtt{POST}}$

USER: Generate the fixed code for the bug RULE with the error message DESCRIPTION. $\mathtt{CODE}^{q}_{\mathtt{PRE}}$

C EXAMPLES FIXES

(a) Input: vulnerable pre-version

```javascript
const express = require('express')
const router = express.Router()

const dbConfig = require('../db/dbConfig')
const mysql = require('mysql')
const pool = mysql.createPool(dbConfig.mysql)

let responseJSON = function (res, ret) {
  if (typeof ret === 'undefined') {
    res.json({
      code: '-200', msg: 'failed operation'
    })
  } else {
    res.json(ret)
  }
}

// ... <REDACTED>

router.get('/api', (req, res, next) => {
  // ... <REDACTED>
  var obj = { name: 'huangming', age: 1 }
  res.json(obj)
})

// ... <REDACTED>

router.get('/postAdvice', (req, res, next) => {
  res.header('Access-Control-Allow-Origin', '*')
  res.header('Access-Control-Allow-Methods', 'PUT, GET, POST, DELETE, OPTIONS')
  res.header('Access-Control-Allow-Headers', 'X-Requested-With')
  res.header('Access-Control-Allow-Headers', 'Content-Type')
  let ip = req.headers['x-forwarded-for'] ||
    req.connection.remoteAddress ||
    req.socket.remoteAddress ||
    (req.connection.socket ? req.connection.socket.remoteAddress : null)
  pool.getConnection((err, connection) => {
    let param = req.query
    let sql = 'INSERT INTO w_advice (username,advice,email,ip) \
            VALUES ("' \
            + param.username + '","' + param.advice \
            + '","' + param.email + '","' + ip + '")'

    connection.query(sql, (err, result) => {


      responseJSON(res, result)
    })
    connection.release()
  })
})
```

(b) Output: non-vulnerable full file

```javascript
const express = require('express')
const router = express.Router()

const dbConfig = require('../db/dbConfig')
const mysql = require('mysql')
const pool = mysql.createPool(dbConfig.mysql)

let responseJSON = function (res, ret) {
  if (typeof ret === 'undefined') {
    res.json({
      code: '-200', msg: 'failed operation'
    })
  } else {
    res.json(ret)
  }
}

// ... <REDACTED>

router.get('/api', (req, res, next) => {
  // ... <REDACTED>
  var obj = { name: 'huangming', age: 1 }
  res.json(obj)
})

// ... <REDACTED>

router.get('/postAdvice', (req, res, next) => {
  res.header('Access-Control-Allow-Origin', '*')
  res.header('Access-Control-Allow-Methods', 'PUT, GET, POST, DELETE, OPTIONS')
  res.header('Access-Control-Allow-Headers', 'X-Requested-With')
  res.header('Access-Control-Allow-Headers', 'Content-Type')
  let ip = req.headers['x-forwarded-for'] ||
    req.connection.remoteAddress ||
    req.socket.remoteAddress ||
    (req.connection.socket ? req.connection.socket.remoteAddress : null)
  pool.getConnection((err, connection) => {
    let param = req.query
    let sql = 'INSERT INTO w_advice (username,advice,email,ip) \
            VALUES (?, ?, ?, ?)'


    connection.query(
                sql,
                [param.username, param.advice, param.email, ip],
                (err, result) => {
      responseJSON(res, result)
    })
    connection.release()
  })
})
```

Fig. 1. DeepCode AI Fix fixes *Sql Injection*, one of the most common and critical vulnerabilities.

(a) Input: vulnerable pre-version

```
// ... <REDACTED>

import fs = require('fs')
import { type Request, type Response, type NextFunction } from 'express'

// ... <REDACTED>

module.exports = function profileImageUrlUpload () {
  return (req: Request, res: Response, next: NextFunction) => {
    if (req.body.imageUrl !== undefined) {
      const url = req.body.imageUrl
      if (url.match(/(.)*solve\/challenges\/server-side(.)*/) !== null)
        req.app.locals.abused_ssrf_bug = true
      const loggedInUser = security.authenticatedUsers.get(req.cookies.token)
      // ... <REDACTED>
      if (loggedInUser) {
        const imageRequest = request
          .get(url)
          .on('error', function (err: unknown) {
            // ... <REDACTED>
          })
          .on('response', function (res: Response) {
            // ... <REDACTED>
          }
          // ... <REDACTED>
      }
    }
  }
}
```

(b) Output: non-vulnerable full file

```
// ... <REDACTED>

import fs = require('fs')
import { type Request, type Response, type NextFunction } from 'express'

// ... <REDACTED>

module.exports = function profileImageUrlUpload () {
  return (req: Request, res: Response, next: NextFunction) => {
    if (req.body.imageUrl !== undefined) {
      const url = typeof req.body.imageUrl === 'string' ? req.body.imageUrl : ''
      if (url.match(/(.)*solve\/challenges\/server-side(.)*/) !== null)
        req.app.locals.abused_ssrf_bug = true
      const loggedInUser = security.authenticatedUsers.get(req.cookies.token)
      // ... <REDACTED>
      if (loggedInUser) {
        const imageRequest = request
          .get(url)
          .on('error', function (err: unknown) {
            // ... <REDACTED>
          })
          .on('response', function (res: Response) {
            // ... <REDACTED>
          }
          // ... <REDACTED>
      }
    }
  }
}
```

Fig. 2. DeepCode AI Fix fixes a *HTTPSourceWithUncheckedType* vulnerability inside *Juice-Shop*, one of the intentionally vulnerable benchmark repositories.

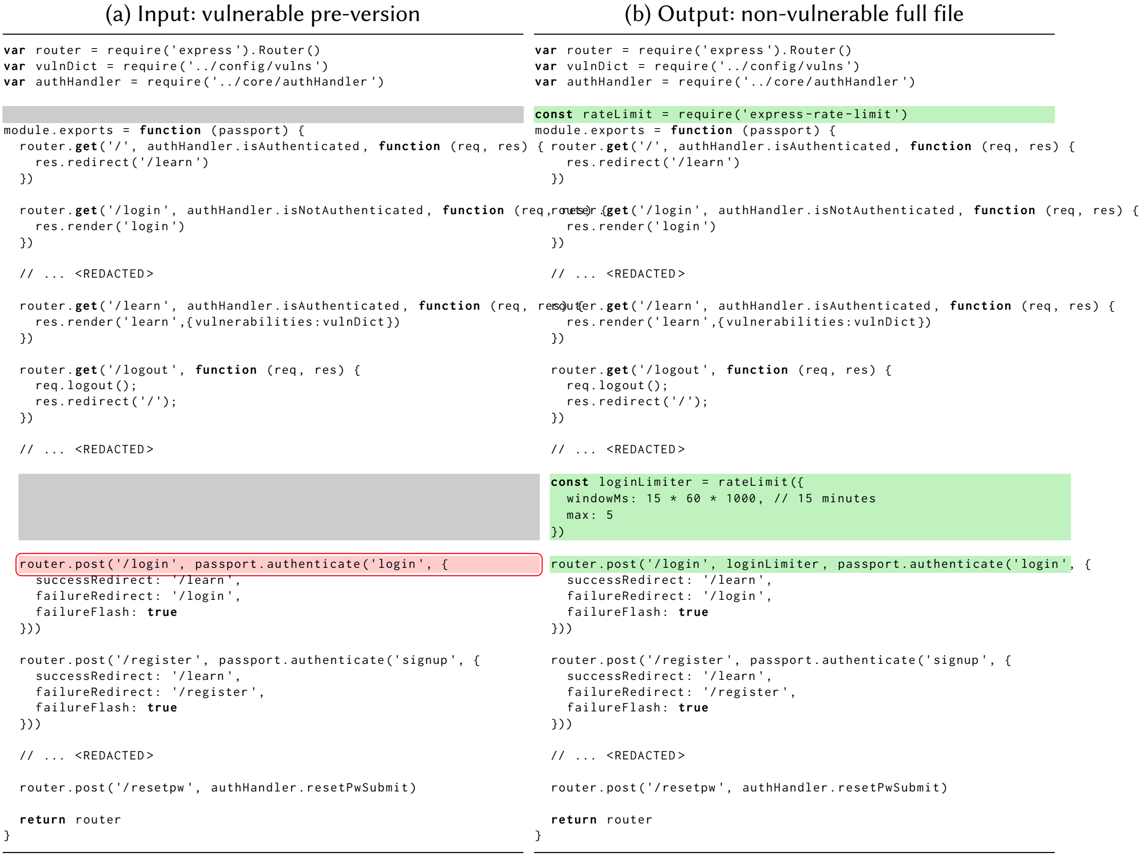
(a) Input: vulnerable pre-version

```
// ... <REDACTED>
const finale = require('finale-rest')

const express = require('express')
const compression = require('compression')
// ... <REDACTED>
const models = require('./models')
const datacreator = require('./data/datacreator')
const app = express()

// ... <REDACTED>
const collectDurationPromise = (name, func) => {
  return async (...args) => {
    const end = startupGauge.startTimer({ task: name })
    const res = await func(...args)
    end()
    return res
  }
}

// ... <REDACTED>

/* Sets view engine to hbs */
app.set('view engine', 'hbs')

restoreOverwrittenFilesWithOriginals().then(() => {
  // ... <REDACTED>
  app.use(errorhandler())
}).catch((err) => {
  // ... <REDACTED>
})

// ... <REDACTED>
```

(b) Output: non-vulnerable full file

```
// ... <REDACTED>
const finale = require('finale-rest')
import csrf = require('csurf')
const express = require('express')
const compression = require('compression')
// ... <REDACTED>
const models = require('./models')
const datacreator = require('./data/datacreator')
const app = express()
app.use(csrf({ cookie: true }))
// ... <REDACTED>
const collectDurationPromise = (name, func) => {
  return async (...args) => {
    const end = startupGauge.startTimer({ task: name })
    const res = await func(...args)
    end()
    return res
  }
}

// ... <REDACTED>

/* Sets view engine to hbs */
app.set('view engine', 'hbs')

restoreOverwrittenFilesWithOriginals().then(() => {
  // ... <REDACTED>
  app.use(errorhandler())
}).catch((err) => {
  // ... <REDACTED>
})

// ... <REDACTED>
```

Fig. 3. DeepCode AI Fix fixes a *UseCsrfForExpress* vulnerability inside *Juice-Shop*. Note that the import statement for *express*, the definition of the app and its usage can be arbitarly far away from each other. DeepCode AI Fix brings them all in the same range and achieves to modify several places in the file without any issue.



(a) Input: vulnerable pre-version

```
var router = require('express').Router()
var vulnDict = require('../config/vulns')
var authHandler = require('../core/authHandler')

module.exports = function (passport) {
  router.get('/', authHandler.isAuthenticated, function (req, res) {
    res.redirect('/learn')
  })

  router.get('/login', authHandler.isNotAuthenticated, function (req, res) {
    res.render('login')
  })

  // ... <REDACTED>

  router.get('/learn', authHandler.isAuthenticated, function (req, res) {
    res.render('learn',{vulnerabilities:vulnDict})
  })

  router.get('/logout', function (req, res) {
    req.logout();
    res.redirect('/');
  })

  // ... <REDACTED>




  router.post('/login', passport.authenticate('login', {
    successRedirect: '/learn',
    failureRedirect: '/login',
    failureFlash: true
  }))

  router.post('/register', passport.authenticate('signup', {
    successRedirect: '/learn',
    failureRedirect: '/register',
    failureFlash: true
  }))

  // ... <REDACTED>

  router.post('/resetpw', authHandler.resetPwSubmit)

  return router
}
```

(b) Output: non-vulnerable full file

```
var router = require('express').Router()
var vulnDict = require('../config/vulns')
var authHandler = require('../core/authHandler')

const rateLimit = require('express-rate-limit')
module.exports = function (passport) {
  router.get('/', authHandler.isAuthenticated, function (req, res) {
    res.redirect('/learn')
  })

  router.get('/login', authHandler.isNotAuthenticated, function (req, res) {
    res.render('login')
  })

  // ... <REDACTED>

  router.get('/learn', authHandler.isAuthenticated, function (req, res) {
    res.render('learn',{vulnerabilities:vulnDict})
  })

  router.get('/logout', function (req, res) {
    req.logout();
    res.redirect('/');
  })

  // ... <REDACTED>

  const loginLimiter = rateLimit({
    windowMs: 15 * 60 * 1000, // 15 minutes
    max: 5
  })

  router.post('/login', loginLimiter, passport.authenticate('login', {
    successRedirect: '/learn',
    failureRedirect: '/login',
    failureFlash: true
  }))

  router.post('/register', passport.authenticate('signup', {
    successRedirect: '/learn',
    failureRedirect: '/register',
    failureFlash: true
  }))

  // ... <REDACTED>

  router.post('/resetpw', authHandler.resetPwSubmit)

  return router
}
```

Fig. 4. DeepCode AI Fix fixes a *NoRateLimiting* vulnerability inside *appsecco/dvna*, one of the intentionally vulnerable benchmark repositories. This is a hard to fix vulnerability because the fix requires changes in 3 different locations of the file and ssome of those changes involve multiple lines.

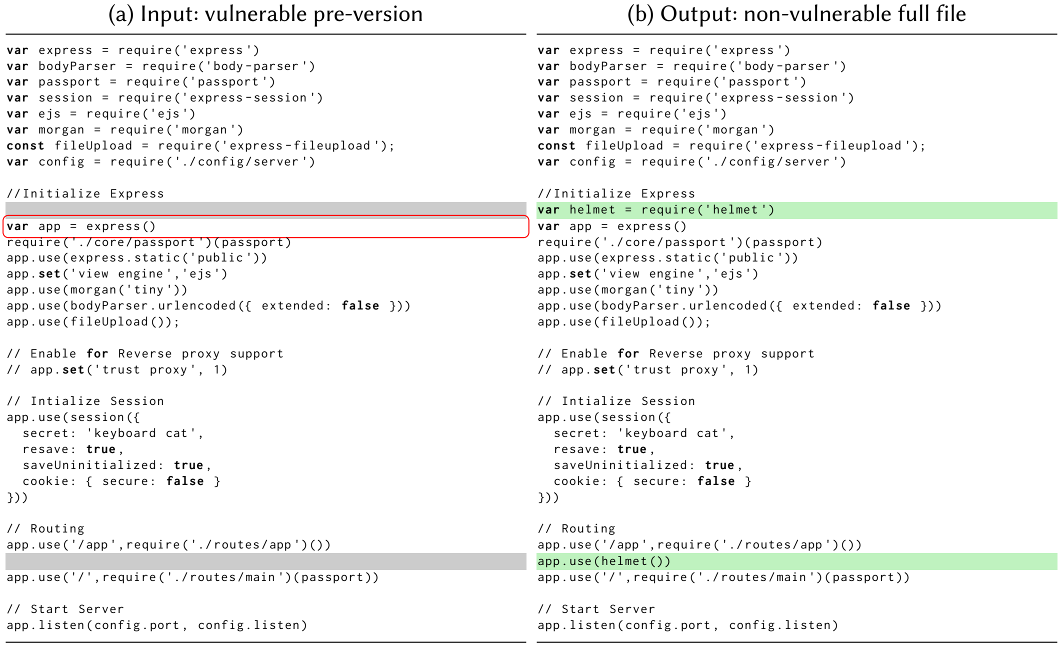
(a) Input: vulnerable pre-version

```
// ... <REDACTED>
const exec = require('child_process').exec;
var mathjs = require('mathjs')
var libxmljs = require("libxmljs");
var serialize = require("node-serialize")
const Op = db.Sequelize.Op

// ... <REDACTED>

module.exports.ping = function (req, res) {
    exec('ping -c 2 ' + req.body.address, function (err, stdout, st
    output = stdout + stderr
    res.render('app/ping', {
      output: output
    })
  })
}

// ... <REDACTED>

module.exports.listProducts = function (req, res) {
  db.Product.findAll().then(products => {
    output = {
      products: products
    }
    res.render('app/products', {
      output: output
    })
  })
}

// ... <REDACTED>

module.exports.listUsersAPI = function (req, res) {
  db.User.findAll({}).then(users => {
    res.status(200).json({
      success: true,
      users: users
    })
  })
}
// ... <REDACTED>
```

(b) Output: non-vulnerable full file

```
// ... <REDACTED>
const exec = require('child_process').exec;
var mathjs = require('mathjs')
var libxmljs = require("libxmljs");
var serialize = require("node-serialize")
const Op = db.Sequelize.Op

// ... <REDACTED>

module.exports.ping = function (req, res) {
    execFile('ping', ['-c', '2', req.body.address], function (err, stdout, stderr
    output = stdout + stderr
    res.render('app/ping', {
      output: output
    })
  })
}

// ... <REDACTED>

module.exports.listProducts = function (req, res) {
  db.Product.findAll().then(products => {
    output = {
      products: products
    }
    res.render('app/products', {
      output: output
    })
  })
}

// ... <REDACTED>

module.exports.listUsersAPI = function (req, res) {
  db.User.findAll({}).then(users => {
    res.status(200).json({
      success: true,
      users: users
    })
  })
}
// ... <REDACTED>
```

Fig. 5. DeepCode AI Fix fixes a *CommandInjection* vulnerability inside *appsecco/dvna*, one of the intentionally vulnerable benchmark repositories. Note how DeepCode AI Fix keeps the required import during reduction and the significant compression rate.



(a) Input: vulnerable pre-version

```
var express = require('express')
var bodyParser = require('body-parser')
var passport = require('passport')
var session = require('express-session')
var ejs = require('ejs')
var morgan = require('morgan')
const fileUpload = require('express-fileupload');
var config = require('./config/server')

//Initialize Express

var app = express()
require('./core/passport')(passport)
app.use(express.static('public'))
app.set('view engine','ejs')
app.use(morgan('tiny'))
app.use(bodyParser.urlencoded({ extended: false }))
app.use(fileUpload());

// Enable for Reverse proxy support
// app.set('trust proxy', 1)

// Intialize Session
app.use(session({
  secret: 'keyboard cat',
  resave: true,
  saveUninitialized: true,
  cookie: { secure: false }
}))

// Routing
app.use('/app',require('./routes/app')())

app.use('/',require('./routes/main')(passport))

// Start Server
app.listen(config.port, config.listen)
```

(b) Output: non-vulnerable full file

```
var express = require('express')
var bodyParser = require('body-parser')
var passport = require('passport')
var session = require('express-session')
var ejs = require('ejs')
var morgan = require('morgan')
const fileUpload = require('express-fileupload');
var config = require('./config/server')

//Initialize Express
var helmet = require('helmet')
var app = express()
require('./core/passport')(passport)
app.use(express.static('public'))
app.set('view engine','ejs')
app.use(morgan('tiny'))
app.use(bodyParser.urlencoded({ extended: false }))
app.use(fileUpload());

// Enable for Reverse proxy support
// app.set('trust proxy', 1)

// Intialize Session
app.use(session({
  secret: 'keyboard cat',
  resave: true,
  saveUninitialized: true,
  cookie: { secure: false }
}))

// Routing
app.use('/app',require('./routes/app')())
app.use(helmet())
app.use('/',require('./routes/main')(passport))

// Start Server
app.listen(config.port, config.listen)
```

Fig. 6. DeepCode AI Fix fixes a *UseHelmetForExpress* vulnerability inside *appsecco/dvna*. The fix is seemingly simple as one can add *helmet* with a single line. However, without adding the right import statement, the code will be broken. A great bug-fixing tool must apply imports correctly. This makes even the seemingly simple fix patterns much harder as the import statements and their usages can be arbitaly far away from each other. The rule *UseHelmetForExpress* belongs to the category *SecurityLocal* but it still requires changes in several different places of the file, just like other "*Local*" rules.

(a) Input: vulnerable pre-version

```
'user strcit';
const config = require('./../../config')
var jwt = require("jsonwebtoken");
const { user } = require('../../orm');

module.exports = (app,db) => {
  app.post('/v1/user/token', (req,res) =>{
    // ... <REDACTED>
  });
  app.post('/v1/user/login', (req,res) =>{
    // ... REDACTED
    const user = db.user.findAll({
      where: {
        email: userEmail
      }}).then(user => {
      })
    // ... REDACTED
  });
  // ... REDACTED
  app.put('/v1/admin/promote/:id', (req,res) =>{
    const userId = req.params.id;
    const user = db.user.update({role:'admin'}, {
      where: {
        id : userId
      }}
    )
    .then((user)=>{
      res.send(user)
    })
  });
  app.post('/v1/user/:id/validate-otp', (req,res) =>{
    // ... REDACTED
    const user = db.user.findOne({
      where: {
        id: userId
      }}).then(user => {
        // ... REDACTED
      })
    // ... REDACTED
  });
};
```

(b) Output: non-vulnerable full file

```
'user strcit';
const config = require('./../../config')
var jwt = require("jsonwebtoken");
const { user } = require('../../orm');

module.exports = (app,db) => {
  app.post('/v1/user/token', (req,res) =>{
    // ... <REDACTED>
  });
  app.post('/v1/user/login', (req,res) =>{
    // ... REDACTED
    const user = db.user.findAll({
      where: {
        email: userEmail
      }}).then(user => {
      })
    // ... REDACTED
  });
  // ... REDACTED
  app.put('/v1/admin/promote/:id', (req,res) =>{
    const userId = req.params.id;
    const user = db.user.update({role:'admin'}, {
      where: {
        id : userId
      }}
    )
    .then((user)=>{
      res.status(200).json(user)
    })
  });
  app.post('/v1/user/:id/validate-otp', (req,res) =>{
    // ... REDACTED
    const user = db.user.findOne({
      where: {
        id: userId
      }}).then(user => {
        // ... REDACTED
      })
    // ... REDACTED
  });
};
```

Fig. 7. DeepCode AI Fix fixes a *XSS* vulnerability inside *SirAppSec/vuln-node.js-express.js-app*, one of the intentionally vulnerable benchmark repositories.

(a) Input: vulnerable pre-version

```
var requirejs = require('requirejs');
var config = requirejs('./config');

// ... <REDACTED>

var express = require('express'),
    http = require('http'),
    path = require('path');

// ... <REDACTED>

var rootDir = path.join(__dirname, '..');

var app = express();

app.configure(function(){
  app.set('port', process.env.PORT || 3000);
  app.use(express.logger('dev'));
  // ... <REDACTED>
});

app.configure('development', function(){
  app.use('/app',
    express.static(path.join(rootDir, 'app')));
  // ... <REDACTED>
});

var server = http.createServer(app)
.listen(app.get('port'), function(){
  console.log("<REDACTED> " + app.get('port'));
});
```

(b) Output: non-vulnerable full file

```
var requirejs = require('requirejs');
var config = requirejs('./config');

// ... <REDACTED>

var express = require('express'),
    https = require('https'),
    path = require('path');

// ... <REDACTED>

var rootDir = path.join(__dirname, '..');

var app = express();

app.configure(function(){
  app.set('port', process.env.PORT || 3000);
  app.use(express.logger('dev'));
  // ... <REDACTED>
});

app.configure('development', function(){
  app.use('/app',
    express.static(path.join(rootDir, 'app')));
  // ... <REDACTED>
});

var server = https.createServer(app)
.listen(app.get('port'), function(){
  console.log("<REDACTED> " + app.get('port'));
});
```

Fig. 8. DeepCode AI Fix fixes a *HttpToHttps* vulnerability requiring multiple changes in different locations of the file.

D EVALUATION OF MODEL SIZE AND ARCHITECHTURE WITH CODEREDUCE

Table 2. Effects of the model size and architecture for CodeReduced data, w.r.t. Pass@k for $k = 1$.

Issue Category	Pass@k (%), $k = 1$		
	StarCoderBase-1B	StarCoderBase-7B	Mistral-7B-Instruct
AST	59.00	**70.05**	63.78
Local	72.02	**82.75**	70.09
FileWide	54.28	**77.38**	51.48
SecurityLocal	67.06	**74.68**	56.33
SecurityFlow	39.64	**49.15**	27.47

E EVALUATION OF DIFFERENT MODELS WITH CODEREDUCE

Table 3. Evaluation of Pass@k and ExactMatch@k metrics for models that use CodeReduce (marked as ‡)

Issue Category	Model	Pass@k (%)			ExactMatch@k (%)		
		$k = 1$	$k = 3$	$k = 5$	$k = 1$	$k = 3$	$k = 5$
AST	Mixtral-8x7B-CodeReduced‡	81.01	88.96	89.16	39.29	52.25	53.5
	StarCoder-7B-CodeReduced‡	70.05	80.84	82.31	41.19	48.15	48.56
	StarCoder-3B-CodeReduced‡	68.2	78.86	81.84	40.51	43.79	43.79
	StableCode-3B-CodeReduced‡	67.21	79.77	80.6	37.69	46.51	47.02
	DeepSeekCoder-1.3B-CodeReduced‡	57.39	75.41	79.27	30.75	39.67	45.48
Local	Mixtral-8x7B-CodeReduced‡	69.77	90.19	92.71	31.37	41.77	44.11
	StarCoder-7B-CodeReduced‡	82.75	89.72	90.19	34.23	42.02	42.73
	StarCoder-3B-CodeReduced‡	81.87	85.82	87.63	33.99	40.79	44.02
	StableCode-3B-CodeReduced‡	77.51	83.39	83.93	32.18	39.72	40.72
	DeepSeekCoder-1.3B-CodeReduced‡	72.72	82.24	85.05	33.98	42.97	45.4
FileWide	Mixtral-8x7B-CodeReduced‡	50.37	67.61	85.66	41.66	47.91	48.95
	StarCoder-7B-CodeReduced‡	77.38	80.16	85.02	44.59	54.5	54.5
	StarCoder-3B-CodeReduced‡	61.88	81.9	82.59	38.69	57.91	73.11
	StableCode-3B-CodeReduced‡	69.74	76.34	76.34	38.4	40.98	40.98
	DeepSeekCoder-1.3B-CodeReduced‡	64.98	89.29	94.89	34.53	62.91	65.27
SecurityLocal	Mixtral-8x7B-CodeReduced‡	63.58	92.2	94.41	15.77	23.46	25.99
	StarCoder-7B-CodeReduced‡	74.68	89.08	91.76	20.18	29.53	32.23
	StarCoder-3B-CodeReduced‡	68.04	88.51	91.41	17.86	25.6	27.43
	StableCode-3B-CodeReduced‡	70.94	84.23	86.49	15.31	18.71	23.36
	DeepSeekCoder-1.3B-CodeReduced‡	61.69	84.18	88.66	15.99	26.84	28.82
SecurityFlow	Mixtral-8x7B-CodeReduced‡	41.93	72.97	81.62	9.26	17.9	20.27
	StarCoder-7B-CodeReduced‡	49.15	65.9	68.53	10.63	13.49	15.7
	StarCoder-3B-CodeReduced‡	45.65	58.82	65.99	9.72	16.17	21.07
	StableCode-3B-CodeReduced‡	41.28	55.32	58.58	9.86	15.37	15.78
	DeepSeekCoder-1.3B-CodeReduced‡	33.26	59.61	66.88	15.22	19.27	19.77