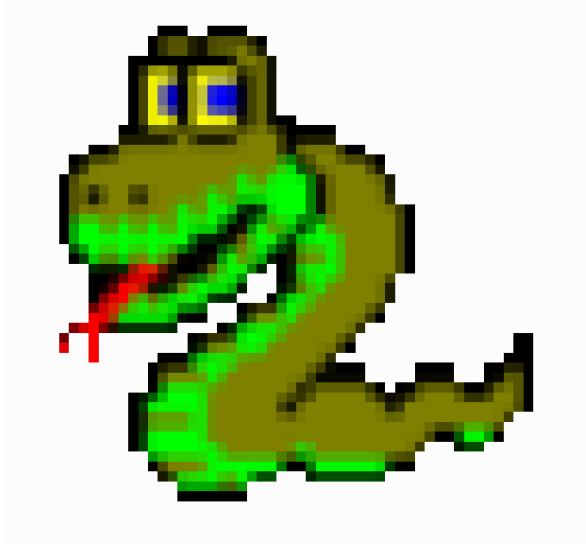
Reverse Engineering Dynamic Languages

A Focus on Python



Aaron Portnoy , Ali Rizvi-Santiago

aportnoy@tippingpoint.com arizvisa@tippingpoint.com

About Us

Work in TippingPoint DVLabs (http://dvlabs.tippingpoint.com)

Responsible for bughunting, patch analysis, vuln-dev

Authors and contributors to…

- Sulley Fuzzing Framework
- PaiMei
- PyMSRPC
- OpenRCE.org

Talk Outline

We will be focusing on Python in its binary forms
- Disassembling code
- Code object modification
- Runtime stuff

An example of reversing Python
- Cheating at an MMORPG

Introduction to Dynamic Languages

What are the characteristics of a dynamic language?

- Most tasks performed at runtime rather than during compilation

Advantages to dynamic languages

- Development speed
- Portability
- Flexibility

Great for lazy coders (like us)

Why Python?

Implements many dynamic features

Rapidly gaining popularity

We were already familiar with its internals

Multiplayer Online Role Playing Game

10,000+ subscribers

Written in Python

Distributed in a binary form

Why this game?

Its TV commercial interrupted Robot Chicken

Pedram wanted to cheat at it

P

First Look

python24.dll

safe to assume, written in Python

What is this 130mb PYD file?

Google says frozen Python objects

Grepping tells us this is likely the source of interesting stuff

Panda3D Library

Made by Disney

What do we know about Python?

Source code compiled to objects
    Interpreted

Python is a dynamic language
    Type information must be present somewhere

Python implements a virtual machine
    Byte code must also be present somewhere

Structure of a PYD

Let’s check it out in IDA

```
dd offset unk_113B0F10
dd 55Bh
dd offset aPirates_shi_36 ; "pirates.ship.PlayerShip"
dd offset unk_113B1470
dd 1262h
dd offset aPirates_ship_p ; "pirates.ship.PlayerShipOV"
dd offset unk_113B26D8
dd 2356h
dd offset aPirates_shi_35 ; "pirates.ship.ShipCameraParams"
dd offset unk_113B4A30
dd 7A4h
dd offset aPirates_shi_34 ; "pirates.ship.ShipGlobals"
dd offset unk_113B51D8
dd 155A9h
dd offset aPirates_shi_33 ; "pirates.ship.ShipInfo"
dd offset unk_113CA788
dd 0ED0h
dd offset aPirates_shi_32 ; "pirates.ship.ShipMeter"
dd offset unk_113CB658
dd 3950h
dd offset aPirates_shi_31 ; "pirates.ship.ShipModel"
dd offset unk_113CEFA8
dd 5DE0h
```

Python Serialization

Python's 'marshal' module

Kind of like pickle, but handles internal types

What is this currently used for?

.pyc – cached code objects (for avoiding having to re-parse)

.pyz – squeezed code objects

.pyd – marshalled code objects stored in a shared object (.dll, .so, etc)

Python Code Object

What do we get when we deserialize?

An object of type ‘code’

Code object properties:

co_argcount, co_nlocals, co_stacksize, co_flags, co_code, co_consts, co_names, co_varnames, co_filename, *co_name*, *co_firstlineno*, *co_lnotab*, *co_freevars*, *co_cellvars*

Which is the most interesting to a reverser?

co_code – string representation of object’s byte code

Byte Code Primer

Instruction consists of a 1-byte opcode followed by an argument when required

Arguments are 16-bits

Has support for extended args

Used if your code has more than 64k of defined constants

Ridiculous getopt implementation?

Like gcc?

Data is not part of byte code

Index references into other code object properties

co_consts

co_names

co_varnames

Byte Code Example

Byte Code Example (cont.)

```
LOAD_CONST  2
LOAD_CONST  78
LOAD_CONST  23
BUILD_TUPLE  3
RETURN_VALUE
```

Code Object Modification

Code objects are immutable

BUT, you can clone an object, optionally modifying attributes

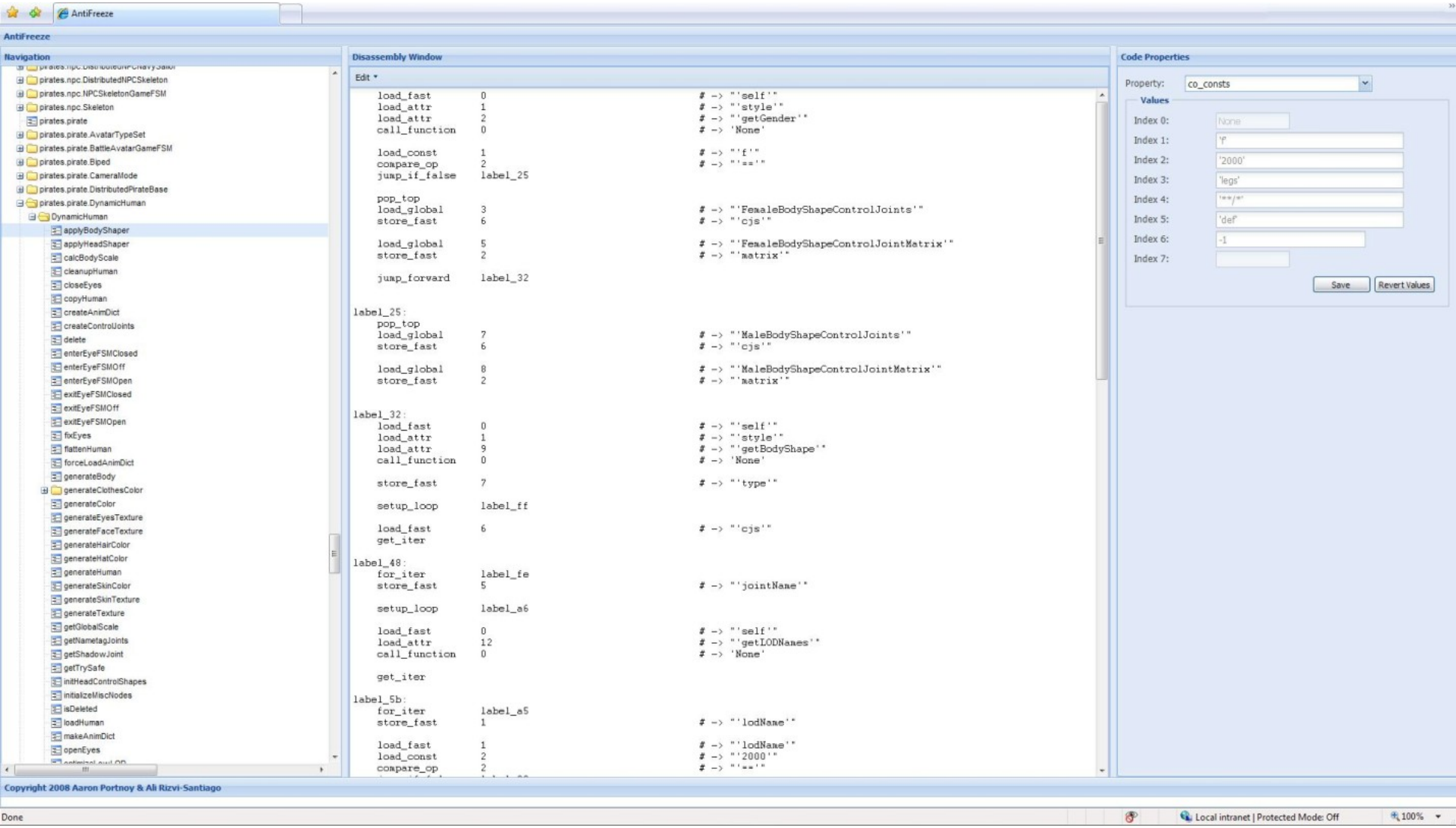
We call this “sneaking the type”™

```
>>> code = type(eval('lambda:x').func_code)
>>> help(code)
Help on class code in module __builtin__:

class code(object)
|  code(argcount, nlocals, stacksize, flags, codestring,
|        constants, names, varnames, filename, name,
|        firstlineno, lnotab[, freevars[, cellvars]])
|
|  Create a code object.  Not for the faint of heart.
```

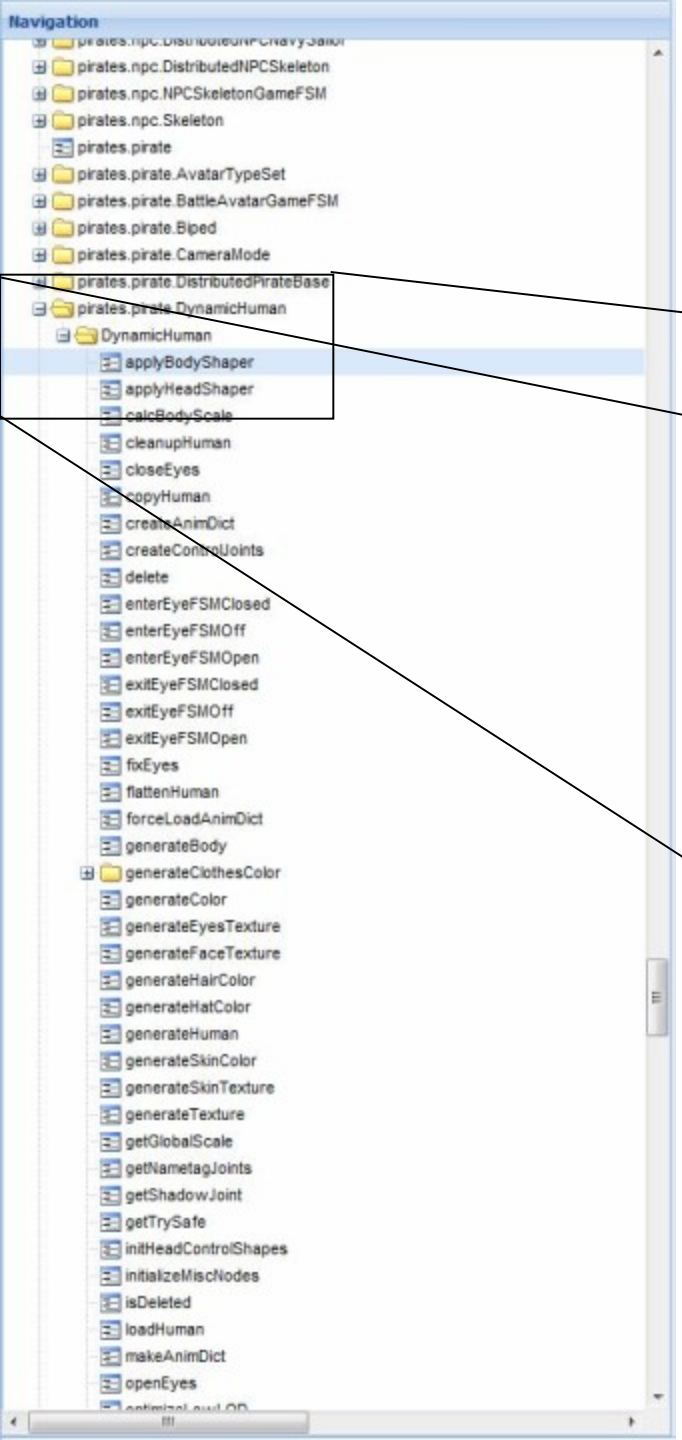
Introducing AntiFreeze

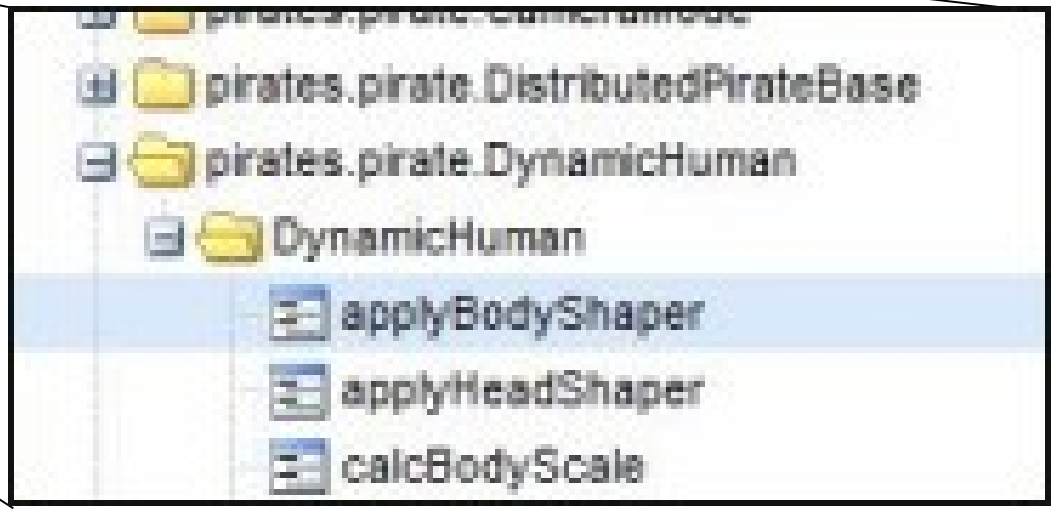
Tool for statically modifying code objects within a PYD
- Web-based
- Interface utilizes Ext-js javascript library

Components
- Disassembly Engine
- Assembler
- Functionality for extracting code objects from a PYD
  - PE Parser
  - Intel Disassembler


AntiFreeze
AntiFreeze
Navigation
pirates.npc.DistributedNPCSkeleton
pirates.npc.NPCSkeletonGameFSM
pirates.npc.Skeleton
pirates.pirate
pirates.pirate.AvatarTypeSet
pirates.pirate.BattleAvatarGameFSM
pirates.pirate.Biped
pirates.pirate.CameraMode
pirates.pirate.DistributedPirateBase
pirates.pirate.DynamicHuman
DynamicHuman
applyBodyShaper
applyHeadShaper
calcBodyScale
cleanupHuman
closeEyes
copyHuman
createAnimDict
createControlJoints
delete
enterEyeFSMClosed
enterEyeFSMOff
enterEyeFSMOpen
exitEyeFSMClosed
exitEyeFSMOff
exitEyeFSMOpen
fixEyes
flattenHuman
forceLoadAnimDict
generateBody
generateClothesColor
generateColor
generateEyesTexture
generateFaceTexture
generateHairColor
generateHatColor
generateHuman
generateSkinColor
generateSkinTexture
generateTexture
getGlobalScale
getNametagJoints
getShadowJoint
getTrySafe
initHeadControlShapes
initializeMiscNodes
isDeleted
loadHuman
makeAnimDict
openEyes
Disassembly Window
Edit
load_fast 0 # -> "'self'"
load_attr 1 # -> "'style'"
load_attr 2 # -> "'getGender'"
call_function 0 # -> 'None'
load_const 1 # -> "'f'"
compare_op 2 # -> "'=='"
jump_if_false label_25
pop_top
load_global 3 # -> "'FemaleBodyShapeControlJoints'"
store_fast 6 # -> "'cjs'"
load_global 5 # -> "'FemaleBodyShapeControlJointMatrix'"
store_fast 2 # -> "'matrix'"
jump_forward label_32
label_25:
pop_top
load_global 7 # -> "'MaleBodyShapeControlJoints'"
store_fast 6 # -> "'cjs'"
load_global 8 # -> "'MaleBodyShapeControlJointMatrix'"
store_fast 2 # -> "'matrix'"
label_32:
load_fast 0 # -> "'self'"
load_attr 1 # -> "'style'"
load_attr 9 # -> "'getBodyShape'"
call_function 0 # -> 'None'
store_fast 7 # -> "'type'"
setup_loop label_ff
load_fast 6 # -> "'cjs'"
get_iter
label_48:
for_iter label_fe
store_fast 5 # -> "'jointName'"
setup_loop label_a6
load_fast 0 # -> "'self'"
load_attr 12 # -> "'getLODNames'"
call_function 0 # -> 'None'
get_iter
label_5b:
for_iter label_a5
store_fast 1 # -> "'lodName'"
load_fast 1 # -> "'lodName'"
load_const 2 # -> "'2000'"
compare_op 2 # -> "'=='"
Code Properties
Property: co_consts
Values
Index 0: None
Index 1: 'f'
Index 2: '2000'
Index 3: 'legs'
Index 4: '**/*'
Index 5: 'def'
Index 6: -1
Index 7:
Save
Revert Values
Copyright 2008 Aaron Portnoy & Ali Rizvi-Santiago
Done
Local intranet | Protected Mode: Off
100%

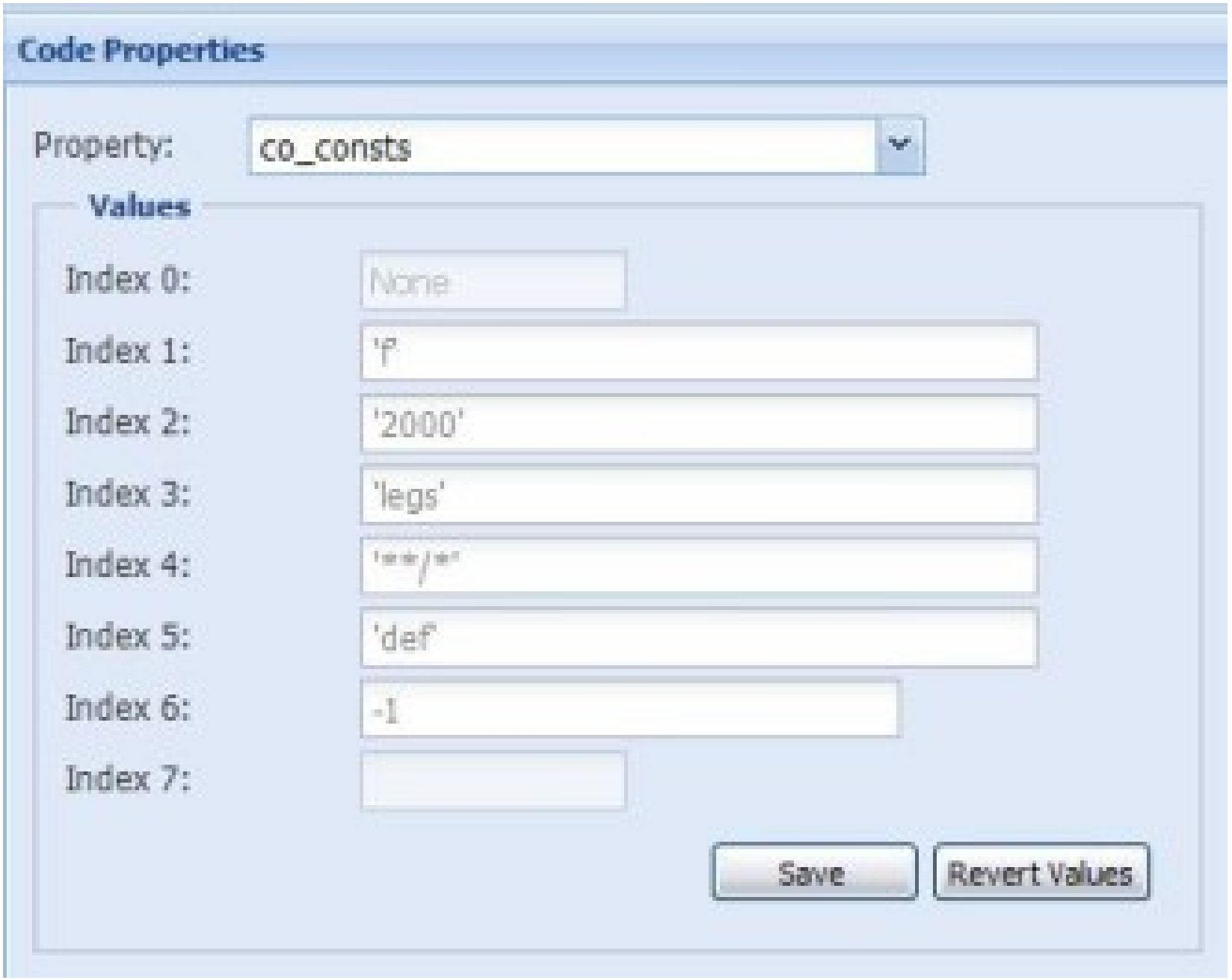
P


Navigation
pirates.npc.DistributedNPCSkeleton
pirates.npc.NPCSkeletonGameFSM
pirates.npc.Skeleton
pirates.pirate
pirates.pirate.AvatarTypeSet
pirates.pirate.BattleAvatarGameFSM
pirates.pirate.Biped
pirates.pirate.CameraMode
pirates.pirate.DistributedPirateBase
pirates.pirate.DynamicHuman
DynamicHuman
applyBodyShaper
applyHeadShaper
calcBodyScale
cleanupHuman
closeEyes
copyHuman
createAnimDict
createControlJoints
delete
enterEyeFSMClosed
enterEyeFSMOff
enterEyeFSMOpen
exitEyeFSMClosed
exitEyeFSMOff
exitEyeFSMOpen
fixEyes
flattenHuman
forceLoadAnimDict
generateBody
generateClothesColor
generateColor
generateEyesTexture
generateFaceTexture
generateHairColor
generateHatColor
generateHuman
generateSkinColor
generateSkinTexture
generateTexture
getGlobalScale
getNametagJoints
getShadowJoint
getTrySafe
initHeadControlShapes
initializeMiscNodes
isDeleted
loadHuman
makeAnimDict
openEyes


pirates.pirate.DistributedPirateBase
pirates.pirate.DynamicHuman
DynamicHuman
applyBodyShaper
applyHeadShaper
calcBodyScale

P

```
    load_attr        1              # -> "'style'"
    load_attr        2              # -> "'getGender'"
    call_function    0              # -> 'None'

    load_const       1              # -> "'f'"
    compare_op       2              # -> "'=='"
    jump_if_false    label_25

    pop_top
    load_global      3              # -> "'FemaleBodyShapeControlJoints'"
    store_fast       6              # -> "'cjs'"

    load_global      5              # -> "'FemaleBodyShapeControlJointMatrix'"
    store_fast       2              # -> "'matrix'"

    jump_forward     label_32


label_25:
    pop_top
    load_global      7              # -> "'MaleBodyShapeControlJoints'"
    store_fast       6              # -> "'cjs'"

    load_global      8              # -> "'MaleBodyShapeControlJointMatrix'"
    store_fast       2              # -> "'matrix'"


label_32:
    load_fast        0              # -> "'self'"
    load_attr        1              # -> "'style'"
    load_attr        9              # -> "'getBodyShape'"
```


Code Properties
Property:
co_consts
Values
Index 0:
None
Index 1:
'f'
Index 2:
'2000'
Index 3:
'legs'
Index 4:
'**/*'
Index 5:
'def'
Index 6:
-1
Index 7:
Save
Revert Values

P

Enough About Static Stuff

Time to explore runtime tricks…

((((Objects and Types) of Objects)) and Types) of Types)

In Python, there are objects and types

Every object has a type associated with it

Every object also inherits from the ‘object’ type
This also includes the ‘type’ type
So, all types inherit from the type type
Which also inherits from the object type

If you try to mentally graph those relationships, you may have an aneurism

Python Object Data Structure

All Instantiated Objects are prefixed with the following information:

```
0       int ob_refcnt
4       struct _typeobject* ob_type
8       int ob_size
```

ob_refcnt – is the reference counter for the object which is utilized for garbage collection

ob_type – contains a pointer to the type of the object

ob_size – duh

Python Standard Types

All base types are exported by the python dll. Check your local dependency viewer for all types.

```
0:001> dd 0x1663660      *this is the address of an object
01663660  00000002 1e1959d0 0000001c 0000001c
01663670  0000007f 01706498 1e051f70 dea555d0
01663680  0166c660 0166c630 7d8c4178 0166f598

0:001> ln 0x1e1959d0              *your ob_type goes here
(1e1959d0)   python24!PyDict_Type
Exact matches:
    python24!PyDict_Type (<no parameter info>)
```

Execution of a Code Object

```
PyFrameObject*

PyEval_EvalCode(PyCodeObject* co, PyObject* globals,
PyObject* locals)
```

Binds Code object to globals()/locals() and returns a PyFrameObject

```
PyObject*

PyEval_EvalFrame(PyFrameObject* f)
```

PyEval_EvalFrame takes the new frame object and is responsible for actual execution.

Concurrent execution of code objects

Multiple interpreters can exist in a single process

Each Interpreter has a list of threads associated with it

Concurrency is handled via a lock known as the GIL

Remember FreeBSD?

PyEval_EvalFrame is responsible for releasing the lock

Diving in With a Debugger

Key things we will need to identify

- All existing interpreters
- Threads associated with an interpreter
- What's currently being executed?

Interpreters

The list of interpreters is a plain old stack

Just need to find a reference to the head of the stack.

“interp_head” in *python-src/Python/pystate.c*

```
0:001> u PyInterpreterState_Head    *mad-friendly
python24!PyInterpreterState_Head:
1e08ce90 a1c0871b1e      mov   eax, [python24!1e1b87c0]
1e08ce95 c3              ret
```

Interpreter Data Structure

```
0   struct _is* next
4   struct _ts* tstate_head
8   PyObject* modules
c   PyObject* sysdict
10  PyObject* builtins
14  PyObject* codec_search_path
18  PyObject* codec_search_cache
1c  PyObject* codec_error_registry
```

Threads

The list of interpreters is also just a plain old stack

```
0  struct _ts* next
4  PyInterpreterState* interp
8  struct _frame* frame
c    int recursion_depth
10   int tracing
14   int use_tracing
…
40 PyObject* dict
…
50 long thread_id      ; this is your GetCurrentThreadId()
```

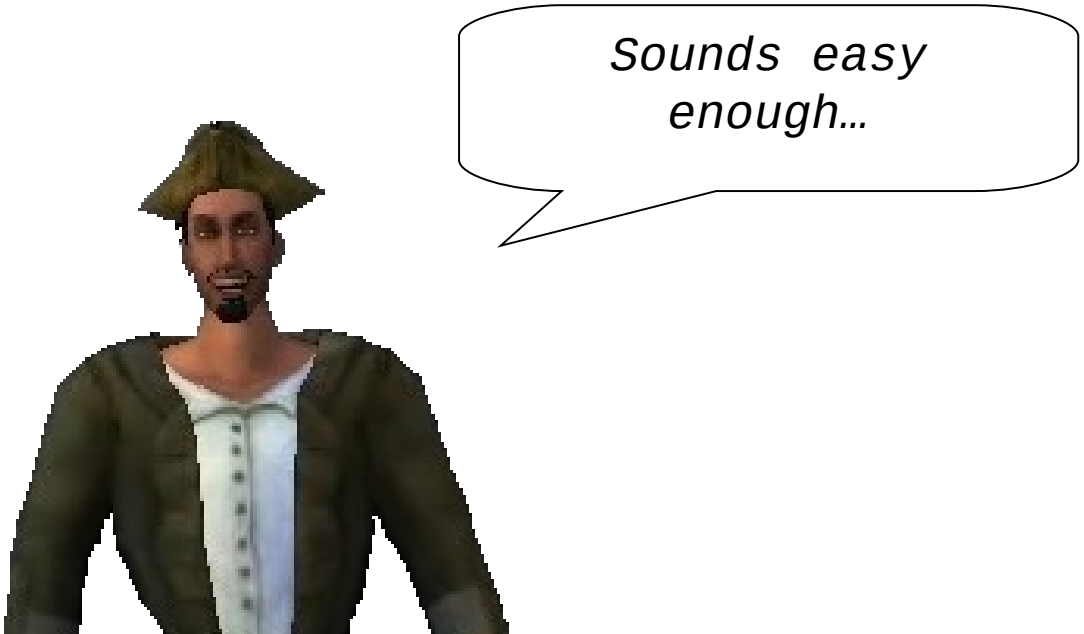
Frame Object

```
0   int ob_refcnt
4   struct _typeobject* ob_type
8   int ob_size

c    struct _frame *f_back   ; calling frame
10   PyCodeObject *f_code
14   PyObject *f_builtins
18   PyDictObject *f_globals
1c   PyDictObject *f_locals
20   PyObject **f_valuestack
24   PyObject **f_stacktop
28   PyObject *f_trace
```

Hooking?

All code must pass through PyEval_EvalCode or PyEval_EvalFrame

Can also hook PyObject_CallFunction or PyObject_CallMethod



Breakpoints

- Breaking on PyEval_EvalFrame
  - Display Name of code object
    - `da poi(poi(poi(@esp+4)+0xc+4)+8+0x2c)+8+0xc`
  - Display Locals
    - `r@$t1=poi(@esp+4);r@$t1=poi(@$t1+0x18);r@$t2=dwo(@$t1+0x10)+1;r@$t1=poi(@$t1+0x14);r@$t3=@$t1+@$t2*@$ptrsize;.while(@$t1<@$t3){r@$t2=poi(@$t1+4);r@$t1=@$t1+@$ptrsize;j(@$t2>0x14)'da@$t2+0x14';''}`
  - Display Globals
    - `r@$t1=poi(@esp+4);r@$t1=poi(@$t1+0x1c);r@$t2=dwo(@$t1+0x10)+1;r@$t1=poi(@$t1+0x14);r@$t3=@$t1+@$t2*@$ptrsize;.while(@$t1<@$t3){r@$t2=poi(@$t1+4);r@$t1=@$t1+@$ptrsize;j(@$t2>0x14)'da@$t2+0x14';''}`
- Breaking on a PyObject_Call*
  - `r@$t1=poi(@esp+4);r@$t2=@$t1;r@$t2=poi(@$t2+0x1c)+0x14;.printf "PyFunction_Type:";da@$t2;r@$t3=@$t1;r@$t3=poi(@$t3+8);r@$t3=poi(@$t3);.printf"PyCFunction_Type";da@$t3;r@$t4=@$t1;r@$t4=poi(@$t4+8);r@$t4=poi(@$t4+0x1c)+0x14;.printf"PyMethod_Type";da@$t4`

Thx WinDBG!

Wait…

Isn’t that a context switch into and out of kernel for execution of EVERY frame?

Userspace Hooking

```
0:000> .dvalloc 1000
Allocated 1000 bytes starting at 00430000

Let's poke around
0:000> u PyEval_EvalFrame
python24!PyEval_EvalFrame:
1e027940 83ec54          sub     esp,54h
1e027943 53              push    ebx
1e027944 8b1dc4871b1e    mov     ebx, [1e1b87c4]
1e02794a 56              push    esi
0:000> a PyEval_EvalFrame
1e027940 jmp 0x430000
1e027945
0:000> u PyEval_EvalFrame
python24!PyEval_EvalFrame:
1e027940 e9bb8640e2      jmp     00430000
1e027945 1dc4871b1e      sbb     eax, 1e1b87c4
1e02794a 56              push    esi
1e02794b 8b742460        mov     esi,dword ptr [esp+60h]
1e02794f 57              push    edi
1e027950 33ff            xor     edi,edi
1e027952 83c8ff          or      eax,0FFFFFFFFh
1e027955 3bf7            cmp     esi,edi
0:000> a 430000
00430000 int 3
00430001 sub esp, 0x54
00430004 push ebx
00430005 mov ebx, [0x1e1b87c4]
0043000b jmp 0x1e02794a
```

Dynamic Recompilation

PyRun_* makes injection incredibly easy. Let's take a look at PyRun_String:

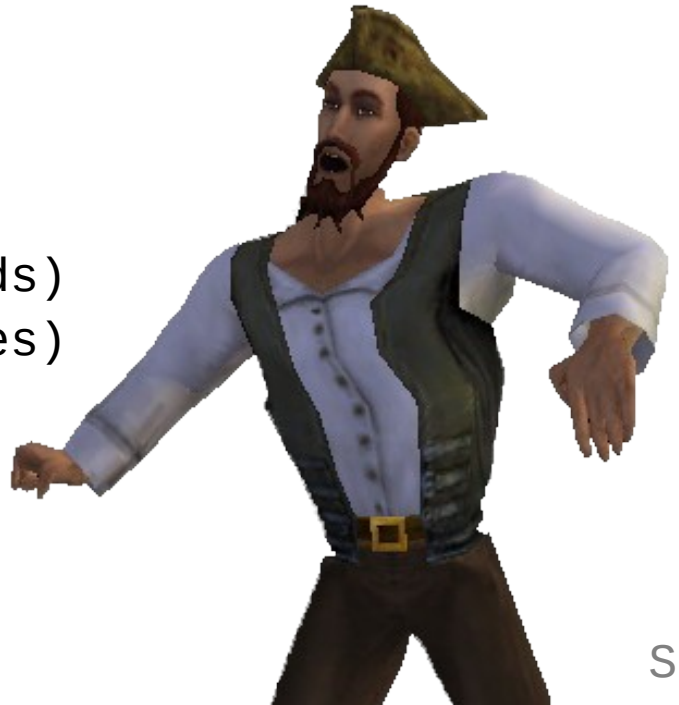
```
PyObject*
PyRun_String(const char* str, int start, PyObject*
   globals, PyObject* locals)
{
   return
   run_err_node(PyParser_SimpleParseString(str,
   start),
                "<string>", globals, locals, NULL);
}
```

Function Hooking in Python

Straightforward approach

Re-declare the function and then call the original:

```
def old(blah, heh, ok, im, over, it):
    print "hello globals()"

original_old = old
def new(*args, **kwds):
    print repr(args), repr(kwds)
    res = original_old(*args, **kwds)
    print "result was: %s"% repr(res)
    return res
old = new
```



Instance Method Hooking in Python

instancemethods are immutable and are bound to an instance

Just need to sneak it's type and then clone with your new function.

```
instancemethod = type(Exception.__str__)
instancemethod(function, instance, class)

class obj(object):
    def method(self):
        print "yay for methods"

def new(self):
    print "okay...."

x = obj()
old = x.method.im_func
x.method = instancemethod(new, x, type(x))
```

Python Supported Debugging Hooks

sys.settrace(fn)

http://docs.python.org/lib/debugger-hooks.html

```
def fn(*args):
    print repr(args)
sys.settrace(fn)
```

ihooks

http://effbot.org/librarybook/ihooks.htm

Enough Boring Stuff, Time for Demos

Static PYD Modifications for Pirates

Digging through the disassembly using AntiFreeze….

We notice *Globals generally contain interesting constants to modify

pirates.reputation.ReputationGlobals

Level/Experience cheats

pirates.economy.EconomyGlobals

Gold cheats

pirates.piratebase.PirateGlobals

Speed/Acceleration/Jump Height/… cheats

pirates.ship.ShipGlobals

Speed/Acceleration cheats


Notoriety
5
Health 373/373
Voodoo 62/62
523 / 1000
Port Royal
William Edgefoote LV5
Joan Bladefoote: ahoy will!
William Stormcutter: hi joan
Cadet: Halt!
Grunt: Mind your own business, pirate!
Sergeant: Surrender!
Sergeant: It's that pirate again!
Grunt: That coward ran away.
Sergeant: This area is off limits!
Sergeant: Over there!
Sergeant: A pirate!
Screenshot captured:
C:\Program Files\Disney\Disney Online\PiratesOnline\screenshot_2008-6-2_11-59-12.jpg
x1
T
365
F7
Tab

P


Shipyard
Light Frigate 800
Sloop 1000
Requires level 5
Galleon 3500
Requires level 5
Frigate 5000
Requires level 5
War Sloop 20000
Requires level 15
War Galleon 10
Requires level 15
War Frigate 1
War Frigate
The Frigate Class vessels pack the most firepower, sporting many cannons and strong below deck broadside capability.
The warship's strongest armor is near the front, with a weakness in the rear.
Ship Profile
Cost: 1
Hull
Sail 2400
Cannons 14
Broadsides 20
Cargo 12
Shipmates 12
Buy
Close


Notoriety
5
Health 391/391
Voodoo 62/62
861 / 1000
Star-chaser Stallion
Hull
Speed
Cargo 2/12
Shipmates 3/12
54:30
Scurvy Shallows
Sea Raider
Frigate
Navy Ferret
lv1 Light Sloop
Navy Ferret
lv2 Light Sloop
Navy Panther
lv9 Light Frigate
Navy Panther
lv8 Light Frigate
Victory Eagle
Light Galleon
Outcast Isle
EITC Sea Vipe
lv6 Light Sloop
Grace Sunspinner (C): what u need again t i forgot lol
Grace Sunspinner: Broadside port!!!
Grace Sunspinner: Broadside port!!!
Grace Sunspinner: Broadside port!!!
Grace Sunspinner: Broadside port!!!
Grace Sunspinner (C): what ships u need again theresa
Grace Sunspinner (C): william is taking a dip
Theresa (C): wondered who that was lol
Screenshot captured:
C:\Program Files\Disney\Disney Online\PiratesOnline\screenshot_2008-6-4_0-2-59.jpg
PISTOL
0 / 50
Level 1
x1
T
380
F7
Tab

P

Caught?

community@disneyonline.com to me show details Jun 5 (9 days ago) Reply

Dear thethunker,

We are writing to inform you that we have found personally identifiable information within a chat log attached to your account, shown below:

- 01:39:03 : two files you need to look at.. phase_1.mf and Phase1.pyd
- 01:39:04 : check http://www.recon.cx
- 01:40:03 : i did tell you... phase_1.mf and Phase1.pyd
- 01:42:00 : http://thunkers.net/~deft/Recon2008
- 01:45:03 : look at my antifreeze_new.jpg
- 02:14:01 : a p o r t n o y at g m a i l

Due to the nature of the information, we have placed your account on a 72 hour hold. To regain access to your account after the 72 hours, please review and agree to our Terms of Use and House Rules here: http://disney.go.com/corporate/legal/terms.html.

We are concerned about your safety online, so exchanging personal information through Disney's community of web sites is not permitted. This includes information like your name, age, e-mail address, school name, phone number, or city of residence. In addition, any discussion of Web addresses or social networking Web sites where personal information is or can be posted or exchanged are not allowed.

Please take this time to visit our netiquette page at: http://home.disney.go.com/guestservices/netiquette to learn more about online safety and manners. Understand that this 72-hour hold serves as a reminder to promote a safer, more pleasant online experience for all Guests. However, keep in mind that further violations of the Terms of Use may result in account suspension or termination.

Thank you,

The Disney Online Team

Screenshot Contest

Disney announced a screenshot contest that coincides with Recon
Top 10 get an iPod Touch

We'll submit our obviously cheating screenshots now…
http://apps.pirates.go.com/pirates/v3/#/community/contests.html

Questions?

Additionally, contact us via e-mail

aportnoy@tippingpoint.com
arizvisa@tippingpoint.com

Blog/Updates/etc at http://dvlabs.tippingpoint.com