Toward classified Instruction Set Architecture using Neural Network

Duc (cothan) Nguyen

List

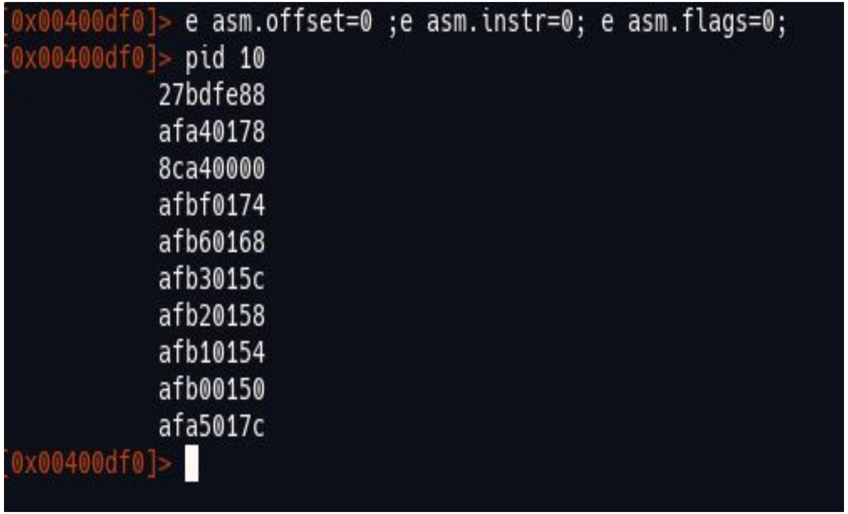
- What is Instruction Set Architecture (ISA)?
- Difficulty in Classifying ISA
- Threat model
- Manual approach in Classifying ISA
- Deep learning approach in Classifying ISA
- Result

Instruction Set Architecture (ISA)

- Assembly code run on a certain platforms. E.g: x86, nios, powerpc, arm …
- Popular ISA can be divided into 2 categories:
  - RISC: Reduced Instruction Set Computer
  - CISC: Complex Instruction Set Computer
- ISA contains:
  - Opcode
  - Register
  - Memory location
  - Operands

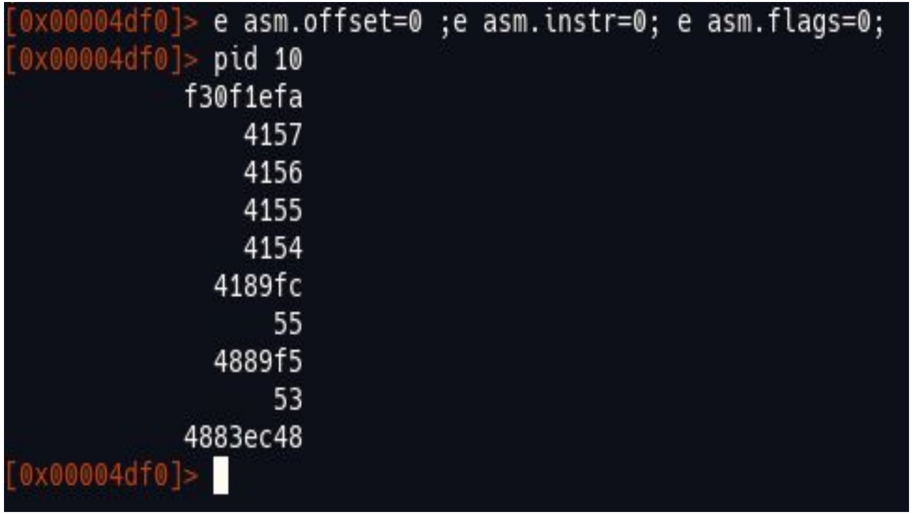
RISC vs. CISC

Fix instruction length



```
[0x00400df0]> e asm.offset=0 ;e asm.instr=0; e asm.flags=0;
[0x00400df0]> pid 10
          27bdfe88
          afa40178
          8ca40000
          afbf0174
          afb60168
          afb3015c
          afb20158
          afb10154
          afb00150
          afa5017c
[0x00400df0]> 
```

Dynamic instruction length



```
[0x00004df0]> e asm.offset=0 ;e asm.instr=0; e asm.flags=0;
[0x00004df0]> pid 10
            f30f1efa
                4157
                4156
                4155
                4154
              4189fc
                  55
              4889f5
                  53
            4883ec48
[0x00004df0]> 
```

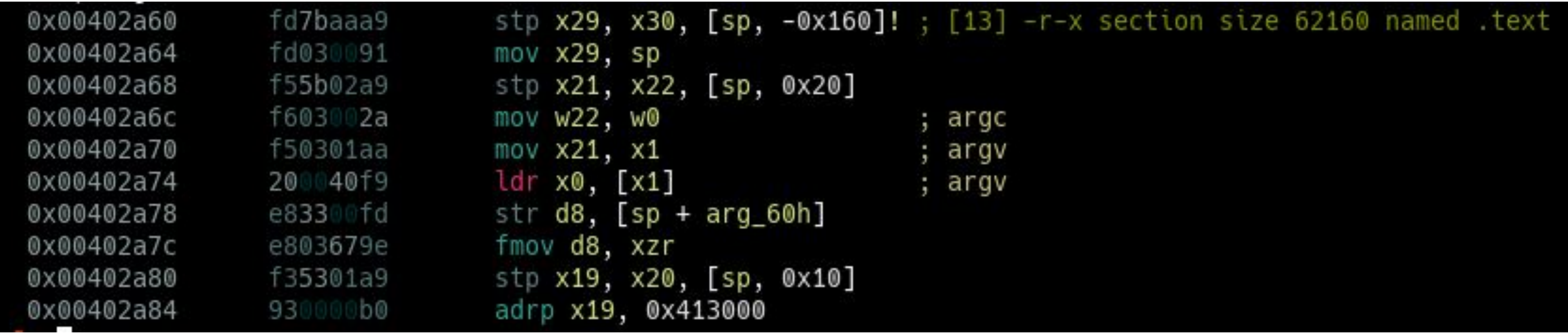
Classified ISA is never easy

Let’s play a game!

Score: 10

What ISA is this?

A. aarch64
B. avr
C. sh
D. mips



```
0x00402a60      fd7baaa9        stp x29, x30, [sp, -0x160]! ; [13] -r-x section size 62160 named .text
0x00402a64      fd030091        mov x29, sp
0x00402a68      f55b02a9        stp x21, x22, [sp, 0x20]
0x00402a6c      f603002a        mov w22, w0                 ; argc
0x00402a70      f50301aa        mov x21, x1                 ; argv
0x00402a74      200040f9        ldr x0, [x1]                ; argv
0x00402a78      e83300fd        str d8, [sp + arg_60h]
0x00402a7c      e803679e        fmov d8, xzr
0x00402a80      f35301a9        stp x19, x20, [sp, 0x10]
0x00402a84      930000b0        adrp x19, 0x413000
```

Score: 10

What ISA is this?

A. aarch64

B. avr

C. sh

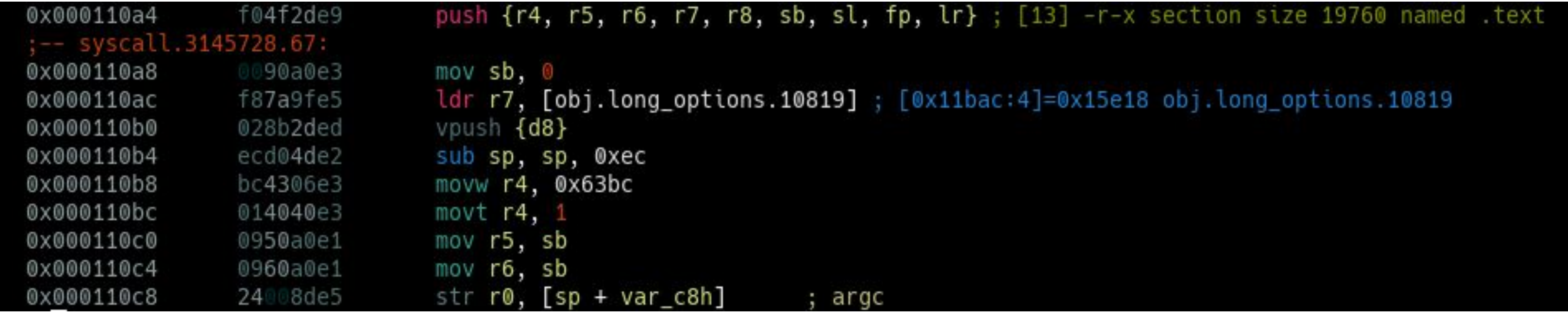
D. mips

E. powerpc64

F. [✓ armv8]

G. mipsel

H. riscv64

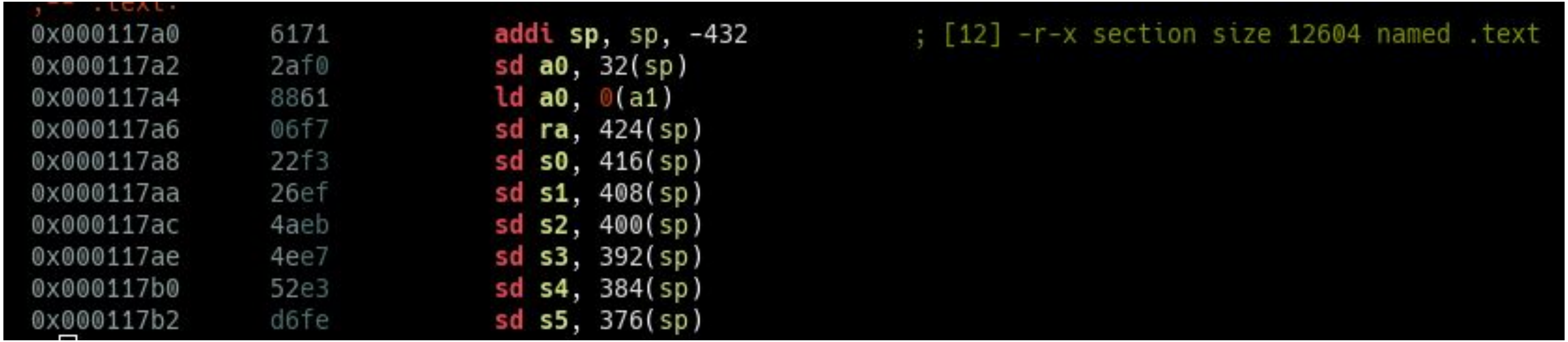


```
0x000110a4      f04f2de9        push {r4, r5, r6, r7, r8, sb, sl, fp, lr} ; [13] -r-x section size 19760 named .text
;-- syscall.3145728.67:
0x000110a8      0090a0e3        mov sb, 0
0x000110ac      f87a9fe5        ldr r7, [obj.long_options.10819] ; [0x11bac:4]=0x15e18 obj.long_options.10819
0x000110b0      028b2ded        vpush {d8}
0x000110b4      ecd04de2        sub sp, sp, 0xec
0x000110b8      bc4306e3        movw r4, 0x63bc
0x000110bc      014040e3        movt r4, 1
0x000110c0      0950a0e1        mov r5, sb
0x000110c4      0960a0e1        mov r6, sb
0x000110c8      24008de5        str r0, [sp + var_c8h]      ; argc
```

Score: 10

What ISA is this?

A. aarch64
B. avr
C. sh
D. mips
E. powerpc64
F. armv8
G. mipsel
[✓ H. riscv64]
I. nios2
J. s390
K. powerpc
L. sparc64



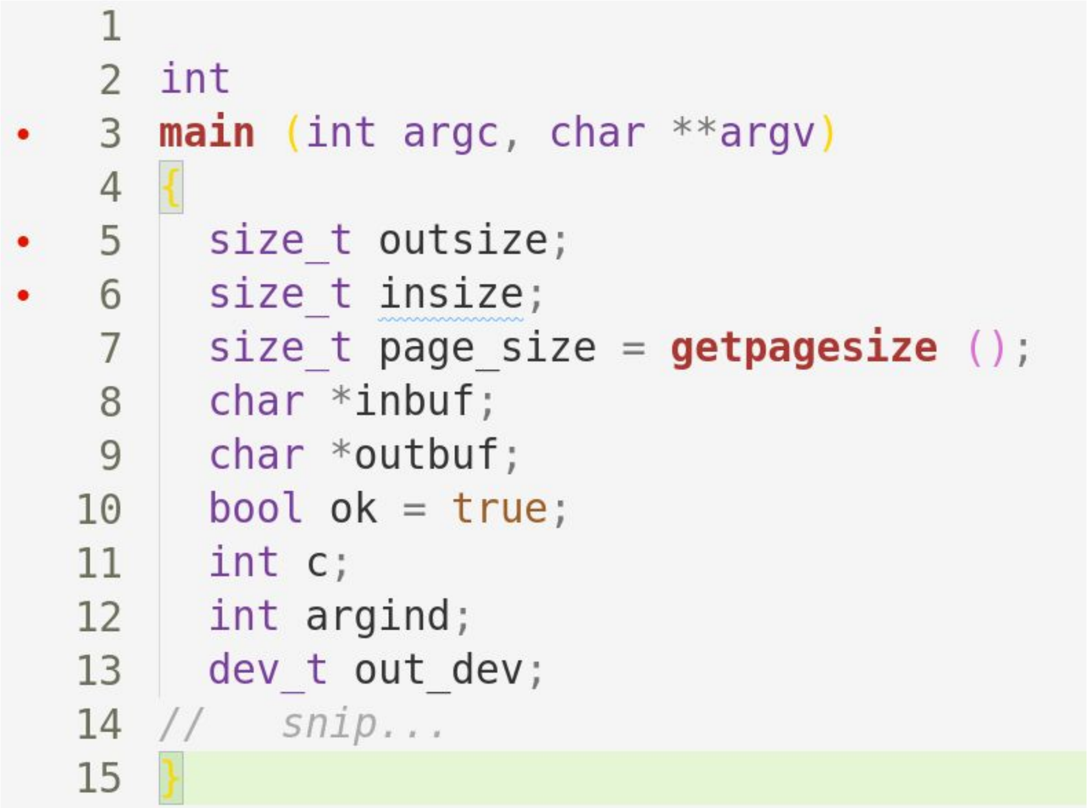
```
0x000117a0      6171          addi sp, sp, -432          ; [12] -r-x section size 12604 named .text
0x000117a2      2af0          sd a0, 32(sp)
0x000117a4      8861          ld a0, 0(a1)
0x000117a6      06f7          sd ra, 424(sp)
0x000117a8      22f3          sd s0, 416(sp)
0x000117aa      26ef          sd s1, 408(sp)
0x000117ac      4aeb          sd s2, 400(sp)
0x000117ae      4ee7          sd s3, 392(sp)
0x000117b0      52e3          sd s4, 384(sp)
0x000117b2      d6fe          sd s5, 376(sp)
```

All pictures are assembly of the same code

- The binary is coreutils/cat.c

- They are instruction code of line 3, 4, 5

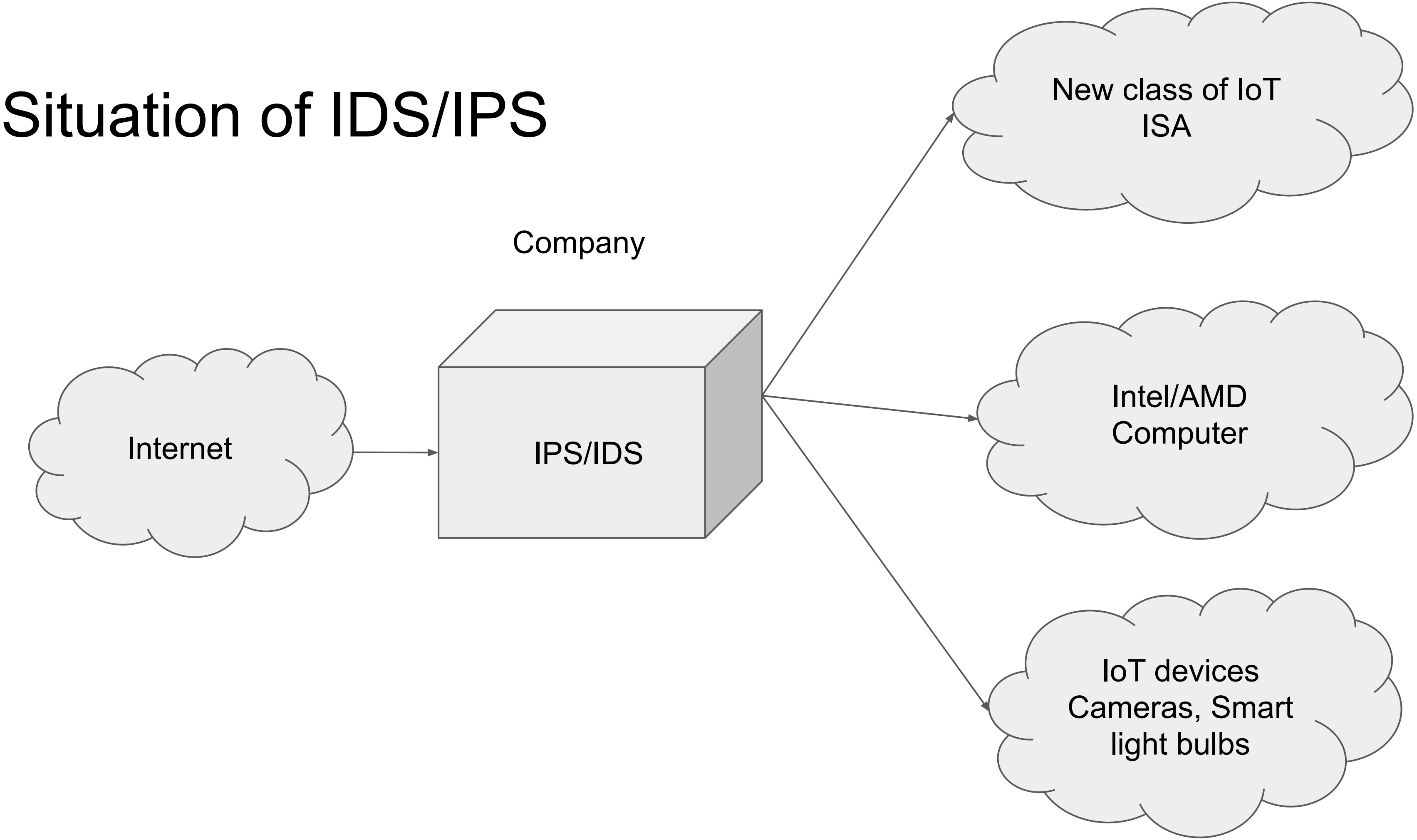
- What do you feel right now ?



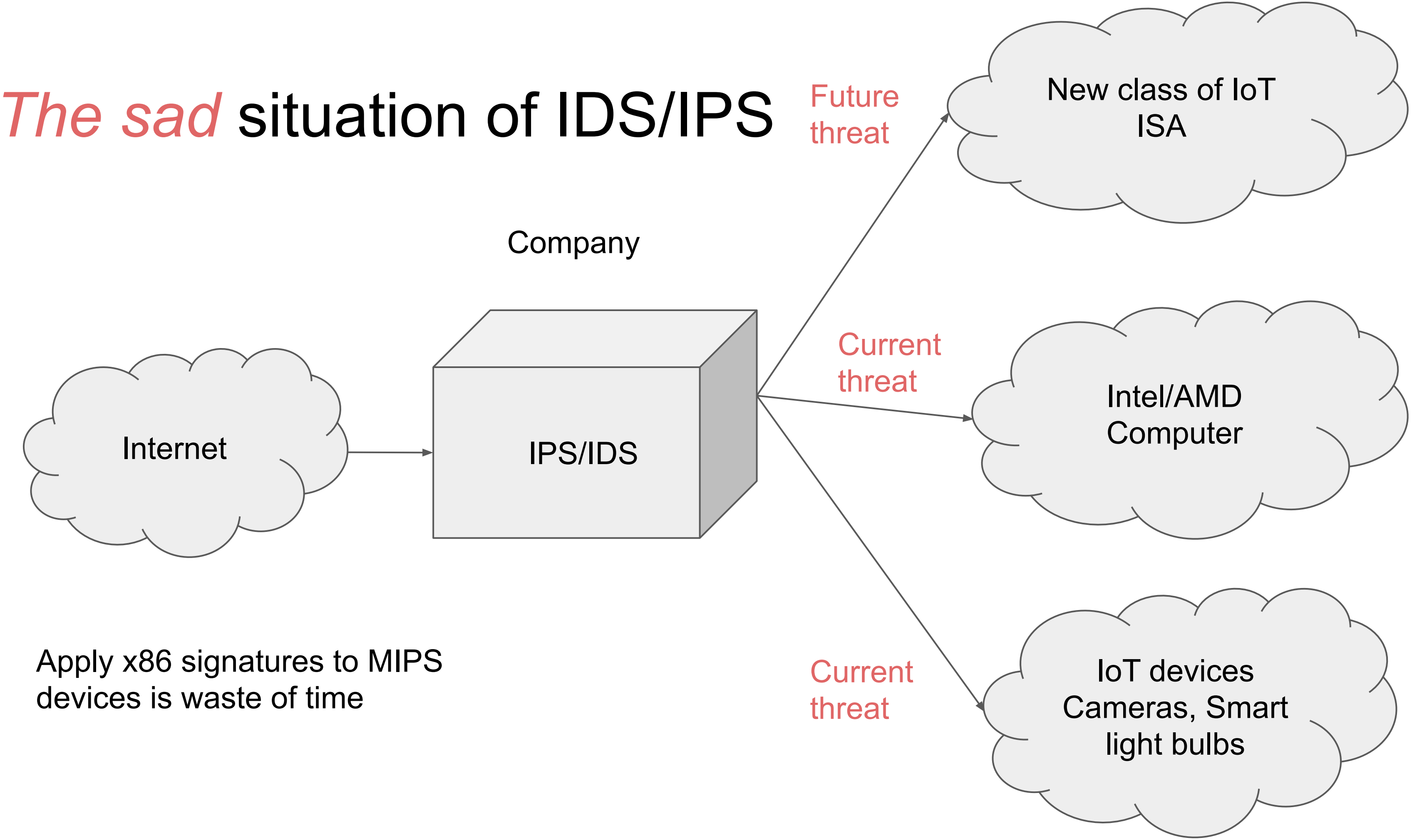
```
    1
    2  int
●   3  main (int argc, char **argv)
    4  {
●   5    size_t outsize;
●   6    size_t insize;
    7    size_t page_size = getpagesize ();
    8    char *inbuf;
    9    char *outbuf;
   10    bool ok = true;
   11    int c;
   12    int argind;
   13    dev_t out_dev;
   14  //   snip...
   15  }
```

Human
ISA
I am inevitable.

Situation of IDS/IPS



The sad situation of IDS/IPS



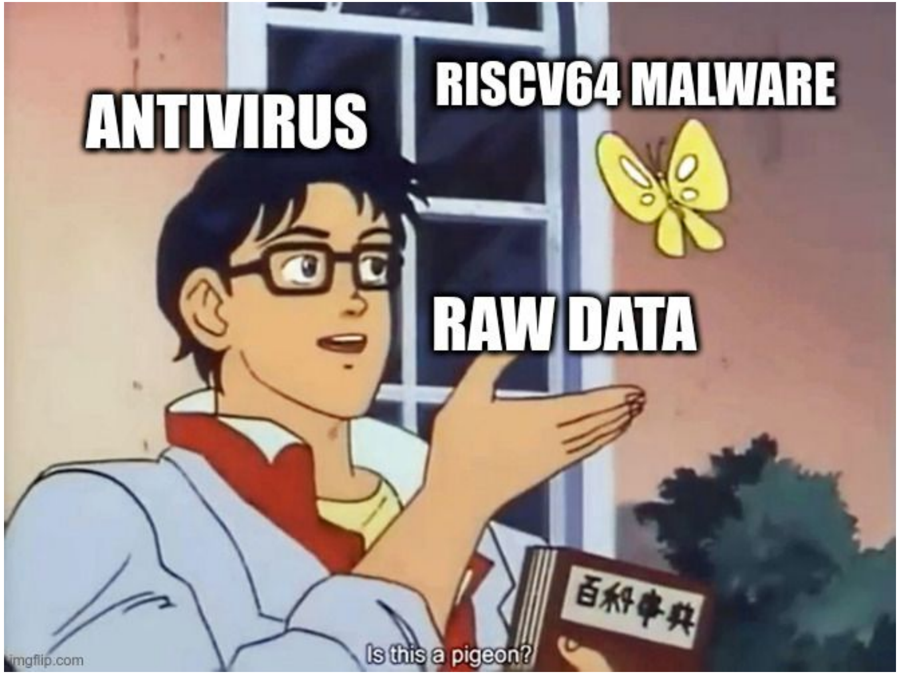
Apply x86 signatures to MIPS devices is waste of time

How IDS/IPS work?

- IDS/IPS works by detecting exploitation stage.
- How do they detect exploit?
  a. Signatures
  b. Heuristics
  c. More signatures: develop 0-day signatures from bug bounty submitters
  d. More heuristics: e.g stop sequence class of actions (behavioral)
- Like cat-mouse game, if there is no known signature, no detection.
- However, IDS/IPS work well for popular ISA, such as x86, x86_64

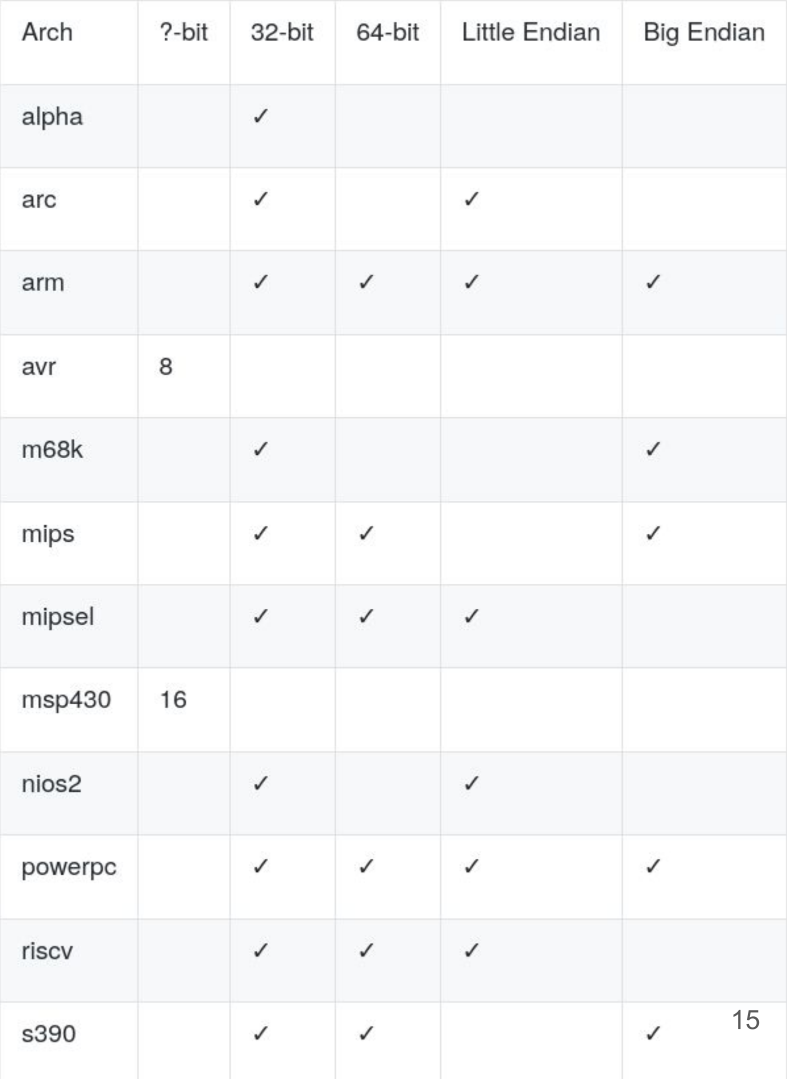
Why IDS/IPS fail to prevent future threat?

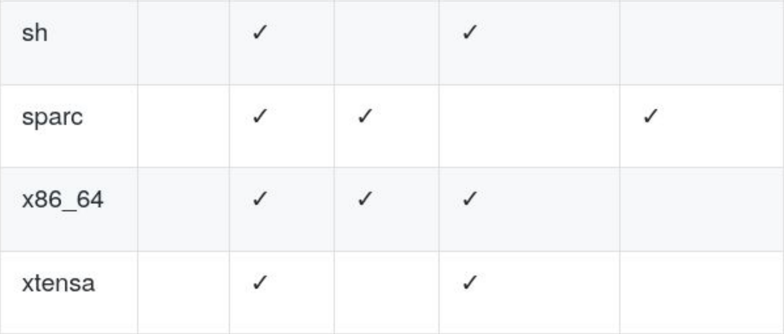
IDS/IPS cannot recognize the instruction inside malware



How many ISAs are there?

… a lot



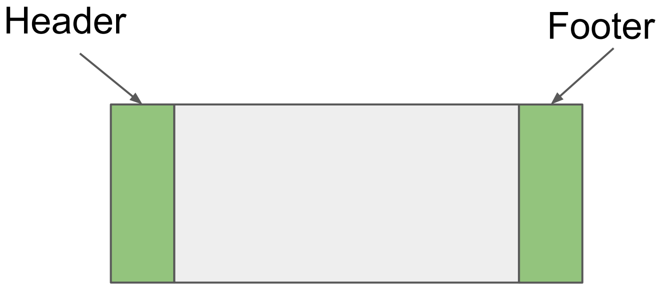


Arch	?-bit	32-bit	64-bit	Little Endian	Big Endian
alpha		✓			
arc		✓		✓	
arm		✓	✓	✓	✓
avr	8				
m68k		✓			✓
mips		✓	✓		✓
mipsel		✓	✓	✓	
msp430	16				
nios2		✓		✓	
powerpc		✓	✓	✓	✓
riscv		✓	✓	✓	
s390		✓	✓		✓
sh		✓		✓	
sparc		✓	✓		✓
x86_64		✓	✓	✓	
xtensa		✓		✓	

https://github.com/cothan/binary-samples

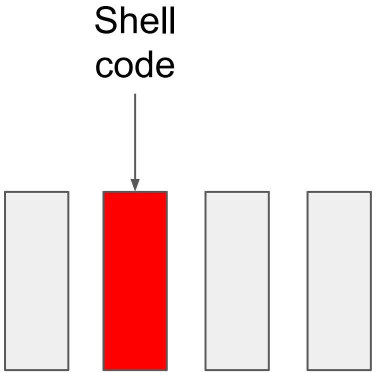
Threat model

At network level, can we detect ISA by looking at **large chunk** of data ?



Threat model

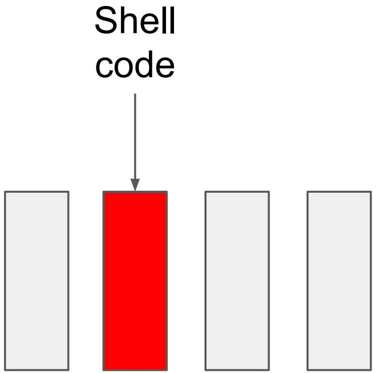
At network level, can we detect ISA and its shellcode by looking at **small chunk** of data ?



Checking Instruction Sequence

Can we detect ISA by looking at **instruction sequence**?

- Yes, by checking the syntax, registers, operands



x86: push r15; push rbp; mov rbp, rsi

mips: bgez v0, 0x401690;
addiu a2, a2, 0x64c0

powerpc: mr r4, r30;
lis r28, 2;
ori r22, r22, 0x49;

Checking syntax, operands, registers

- False positive in disassembler
- Easily fall into junk code trap
- Trial and error process

It’s hard, even for human.

“Junk code is a sequence of bytes that you have disassembled that are not actual instructions executed as part of a program. In addition to wasting time, I’ve seen people get alarmed and excited by the junk code they’ve found.”

Nick Harbour, FireEye

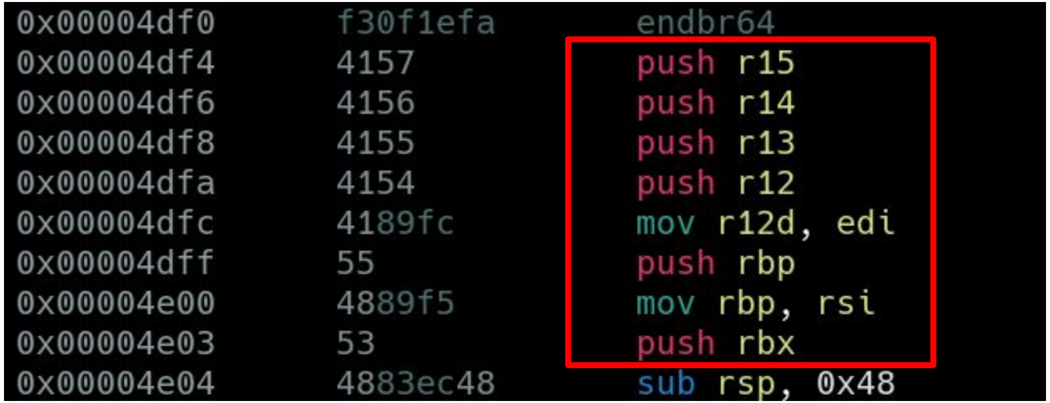
https://www.fireeye.com/blog/threat-research/2017/12/recognizing-and-avoiding-disassembled-junk.html

Classified ISA from raw byte

Can we detect ISA by looking at a sequence of raw byte ?

1. Heuristic

- Expensive analysis
- Easily bypassed by reordering instruction sequence



```
0x00004df0          f30f1efa          endbr64
0x00004df4          4157              push r15
0x00004df6          4156              push r14
0x00004df8          4155              push r13
0x00004dfa          4154              push r12
0x00004dfc          4189fc            mov r12d, edi
0x00004dff          55                push rbp
0x00004e00          4889f5            mov rbp, rsi
0x00004e03          53                push rbx
0x00004e04          4883ec48          sub rsp, 0x48
```

Classified ISA from raw byte

Can we detect ISA by looking at a sequence of raw byte ?

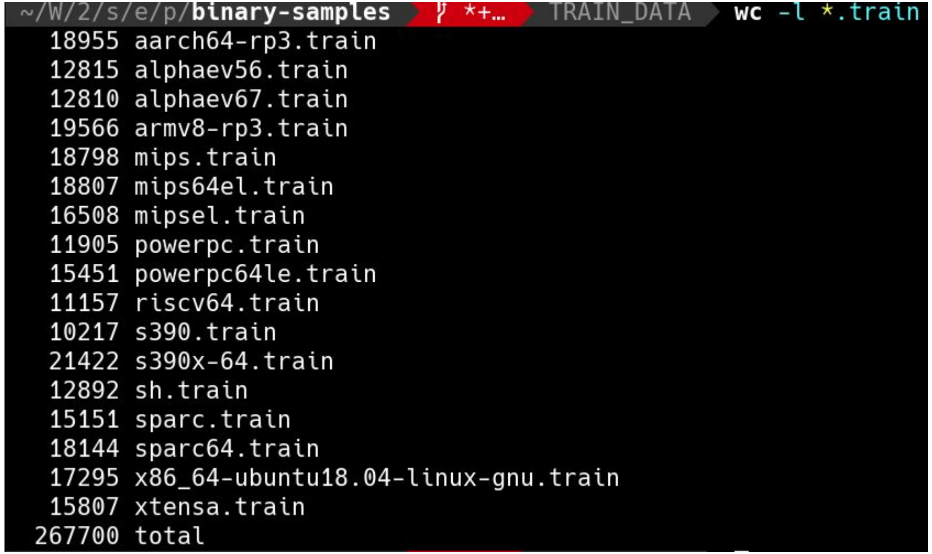
2. Neural Network

- Extensive training
- Good at classified/categorized ISA

Classified ISA using Deep Learning

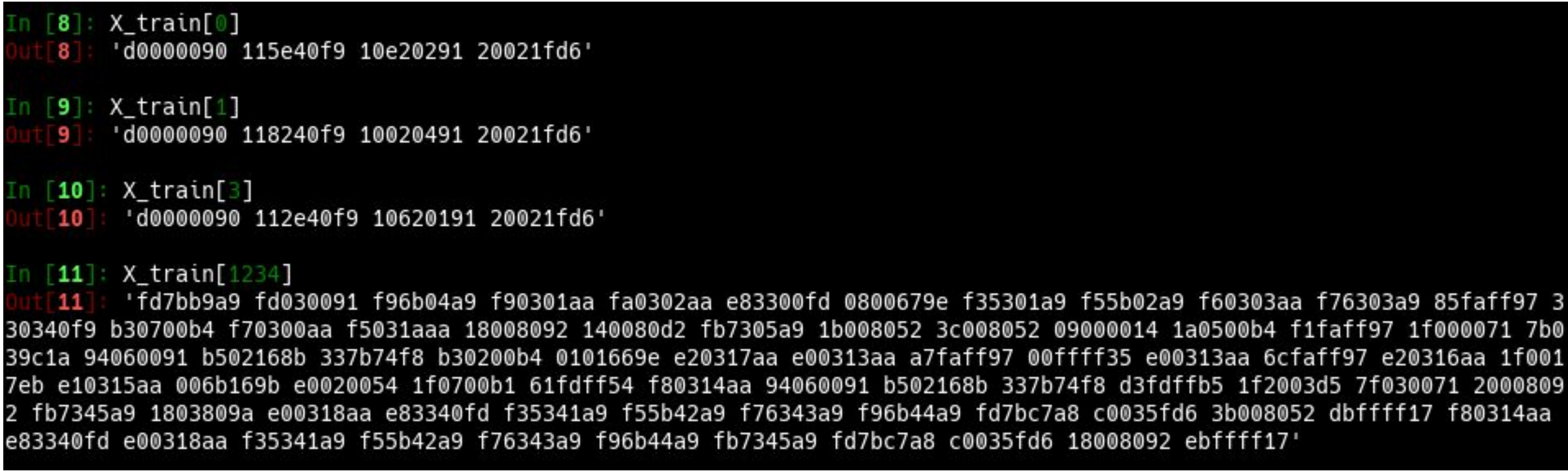
Prepare Training set:

- 17 architectures, built by crosstool-ng
- 817 Mb from 2040 binaries
- All binaries are dynamically linked
- Radare2 script to extract instructions
- Each sequence has different length, longest sequence is ~2000 bytes
- Split each sequence into smaller 64 bytes chunk



```
~/W/2/s/e/p/binary-samples  ⎇ *+…  TRAIN_DATA  wc -l *.train
  18955 aarch64-rp3.train
  12815 alphaev56.train
  12810 alphaev67.train
  19566 armv8-rp3.train
  18798 mips.train
  18807 mips64el.train
  16508 mipsel.train
  11905 powerpc.train
  15451 powerpc64le.train
  11157 riscv64.train
  10217 s390.train
  21422 s390x-64.train
  12892 sh.train
  15151 sparc.train
  18144 sparc64.train
  17295 x86_64-ubuntu18.04-linux-gnu.train
  15807 xtensa.train
 267700 total
```

What is the training data look like ?



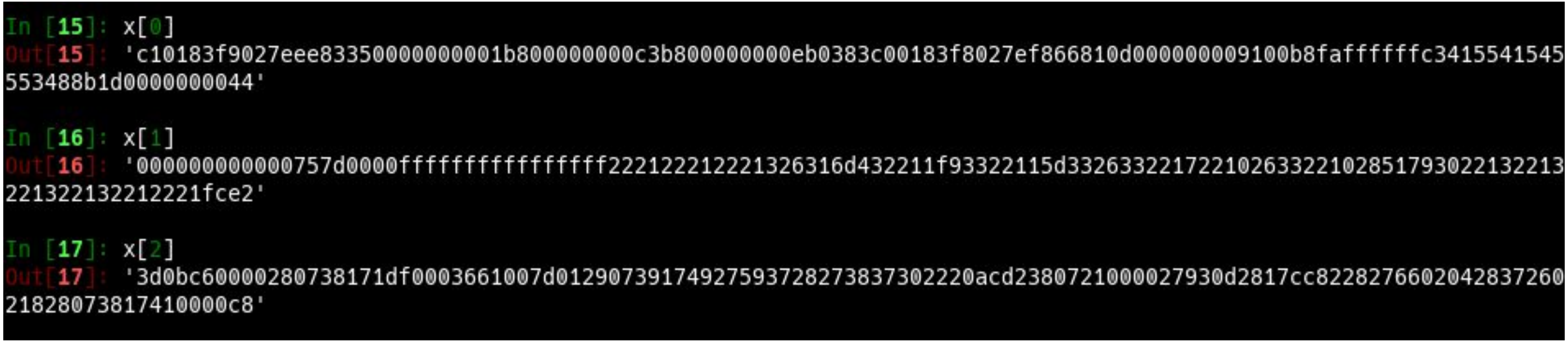
```
In [8]: X_train[0]
Out[8]: 'd0000090 115e40f9 10e20291 20021fd6'

In [9]: X_train[1]
Out[9]: 'd0000090 118240f9 10020491 20021fd6'

In [10]: X_train[3]
Out[10]: 'd0000090 112e40f9 10620191 20021fd6'

In [11]: X_train[1234]
Out[11]: 'fd7bb9a9 fd030091 f96b04a9 f90301aa fa0302aa e83300fd 0800679e f35301a9 f55b02a9 f60303aa f76303a9 85faff97 3
30340f9 b30700b4 f70300aa f5031aaa 18008092 140080d2 fb7305a9 1b008052 3c008052 09000014 1a0500b4 f1faff97 1f000071 7b0
39c1a 94060091 b502168b 337b74f8 b30200b4 0101669e e20317aa e00313aa a7faff97 00ffff35 e00313aa 6cfaff97 e20316aa 1f001
7eb e10315aa 006b169b e0020054 1f0700b1 61fdff54 f80314aa 94060091 b502168b 337b74f8 d3fdffb5 1f2003d5 7f030071 2000809
2 fb7345a9 1803809a e00318aa e83340fd f35341a9 f55b42a9 f76343a9 f96b44a9 fd7bc7a8 c0035fd6 3b008052 dbffff17 f80314aa
e83340fd e00318aa f35341a9 f55b42a9 f76343a9 f96b44a9 fb7345a9 fd7bc7a8 c0035fd6 18008092 ebffff17'
```

What is predict data look like?



```
In [15]: x[0]
Out[15]: 'c10183f9027eee83350000000001b800000000c3b800000000eb0383c00183f8027ef866810d000000009100b8faffffffc3415541545
553488b1d0000000044'

In [16]: x[1]
Out[16]: '000000000000757d0000ffffffffffffffff22212221222132631 6d432211f93322115d332633221722102633221028517930221322 13
221322132212221fce2'

In [17]: x[2]
Out[17]: '3d0bc60000280738171df0003661007d0129073917492759372827383730222 0acd23807210000279 30d2817cc822827660204283726 0
21828073817410000c8'
```

Classified ISA using Deep Learning

Data preprocessing:

- Encode each chunk: **Tokenizer**, **one-hot**, **word2vec**

Classifier Learning:

- CNN
- RNN
- LSTM

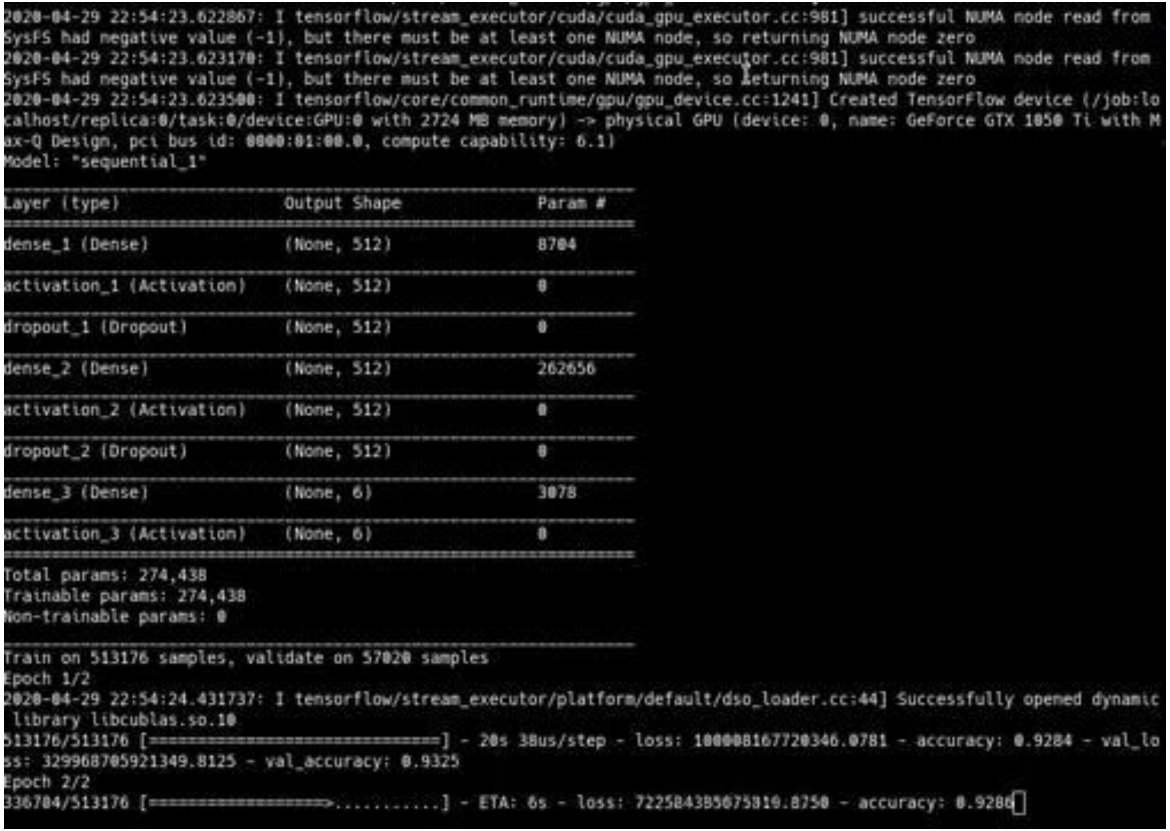
Future Work

Result

Type	Encoder	Layers	Accuracy
CNN 3rd	Tokenizer	8	92%
CNN 2nd	One Hot	4	15%
CNN 1st	Tokenizer	4	9%

Training In Action

Demo



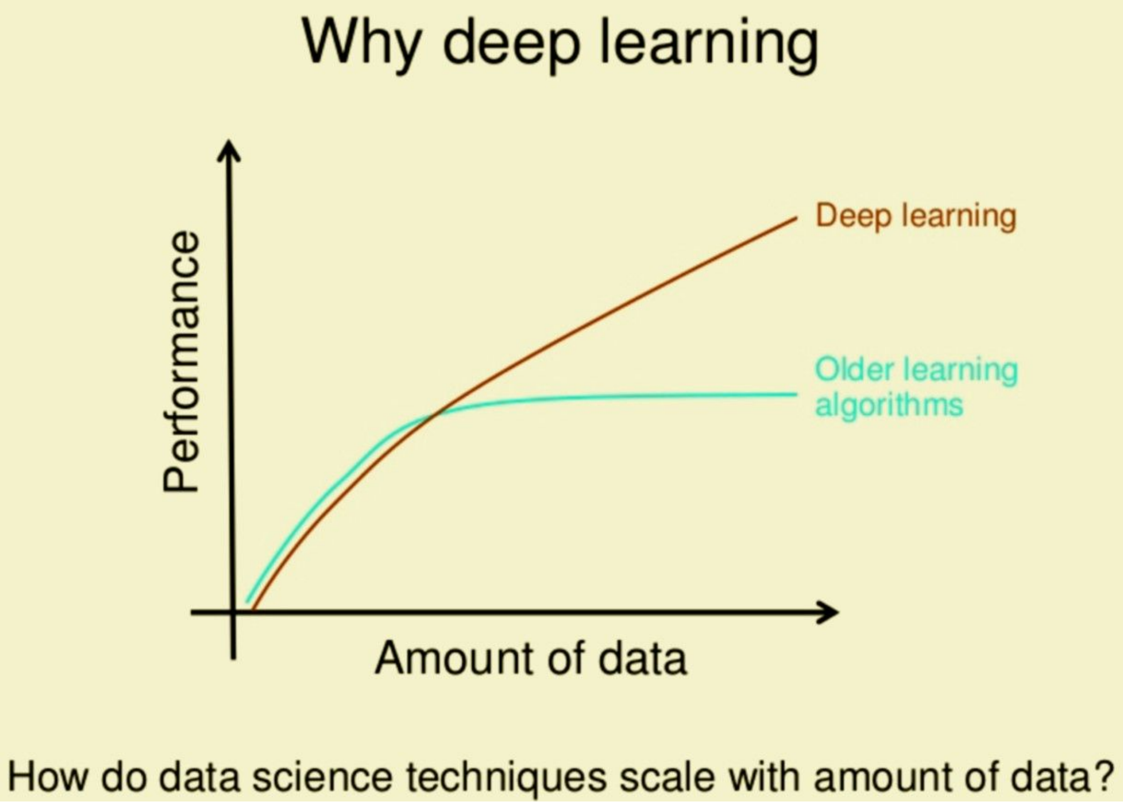
```
2020-04-29 22:54:23.622867: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:981] successful NUMA node read from
SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2020-04-29 22:54:23.623170: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:981] successful NUMA node read from
SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2020-04-29 22:54:23.623500: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1241] Created TensorFlow device (/job:lo
calhost/replica:0/task:0/device:GPU:0 with 2724 MB memory) -> physical GPU (device: 0, name: GeForce GTX 1050 Ti with M
ax-Q Design, pci bus id: 0000:01:00.0, compute capability: 6.1)
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 512)               8704
_________________________________________________________________
activation_1 (Activation)    (None, 512)               0
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               262656
_________________________________________________________________
activation_2 (Activation)    (None, 512)               0
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 3078
_________________________________________________________________
activation_3 (Activation)    (None, 6)                 0
=================================================================
Total params: 274,438
Trainable params: 274,438
Non-trainable params: 0
_________________________________________________________________
Train on 513176 samples, validate on 57020 samples
Epoch 1/2
2020-04-29 22:54:24.431737: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic
 library libcublas.so.10
513176/513176 [==============================] - 20s 38us/step - loss: 100008167720346.0781 - accuracy: 0.9284 - val_lo
ss: 329968705921349.8125 - val_accuracy: 0.9325
Epoch 2/2
336704/513176 [==================>...........] - ETA: 6s - loss: 722584385075810.8750 - accuracy: 0.9286
```

Why Deep Learning instead of Machine Learning

- Better accuracy

- Can solve problem with unknown solutions

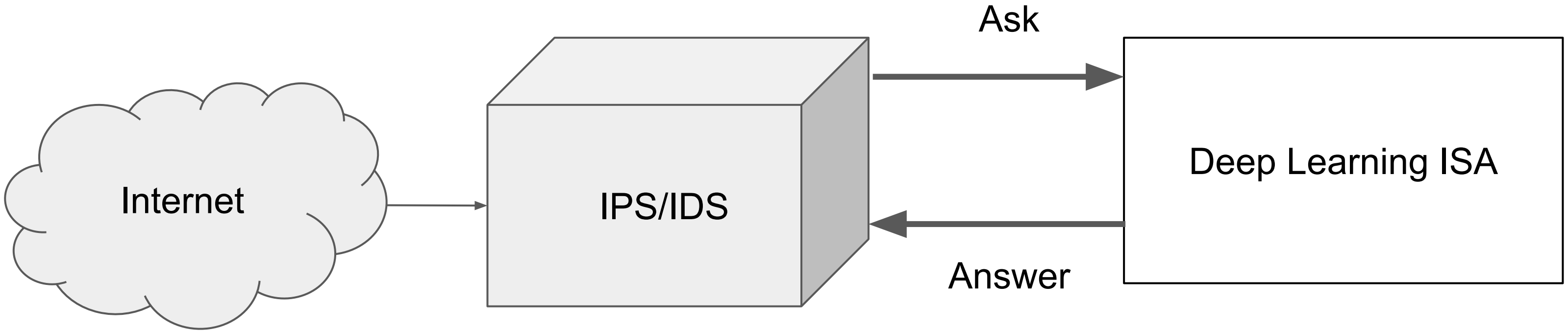
https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063



Why deep learning

How do data science techniques scale with amount of data?

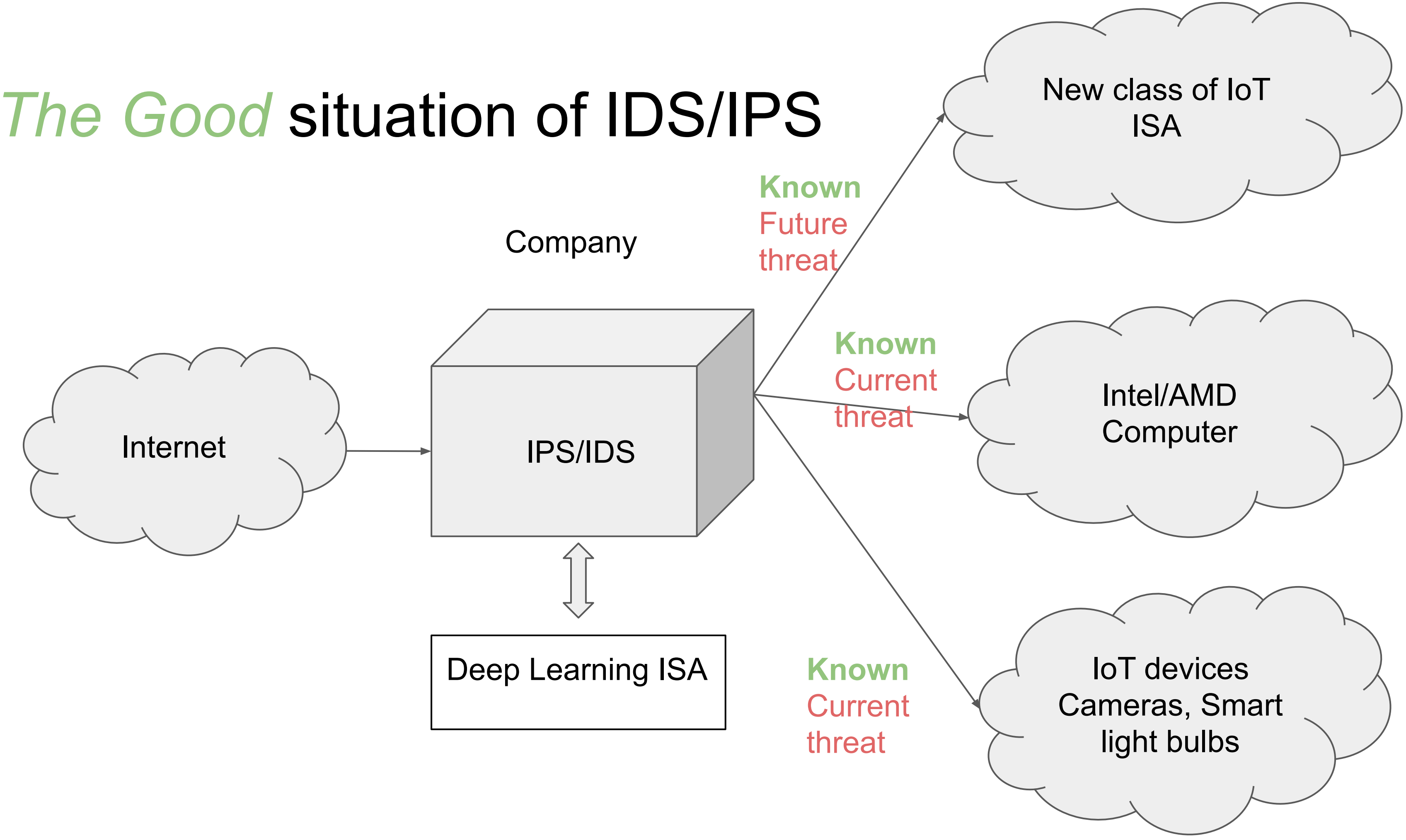
Let’s get back to our threat model



If Answer is ISA instruction:
- Activate signature/heuristic filter

Else:
- Pass

The Good situation of IDS/IPS



ISA
Deep Learning
I am inevitable.

Future work

The current accuracy can’t get more than 93%. Need to improve.

Future work:

- Need more sample for embedded system like avr, sh, …
- Apply different neural network
- Apply different encoder
- Apply deep learning to detect malware/shellcode in different architectures

Question ?

THANK YOU