HGAME 2021 WEEK WRITE UP

HGAME 2021 WEEK WRITE UP

MISC

A R K accuracy

MISC

ARK

导入 ssl 证书后,可以查看 http 协议的包了。在 wi reshark 中, 文件 - 导出对象 - HTTP ,将所有文件导出。

结合题中提到的"自律"功能,合理怀疑两个名为 getBattleReplay 的文件。打开后发现有 battleReplay 字段,包括数字、大小写字母,以 == 结尾,长度约14k字节。猜测是 base64 编码,且包括所有记录操作。解码后得到如下内容(开头部分)

```
PK\square??\square?·bKR¢Ûa\squaret(??\square¾\square?
???default_entryĝM«Ó@\square@ÿKÜF\square1w2\squareîT\\square"âB\square6j4M\muI\squareØ\betaæÖ/t ÀE\_¿Îkm^{\dagger}7ç0¼oÍë~ÿ\pm\squareß\Omegaæé ......
```

结合 PK 字段及 default_entry ,猜测为 zip 文件。将解码后的二进制代码用 WinHex 保存 . zip 文件。成功打开并解压,得到 default_entry 。用记事本打开,内容如下(开头部分)

```
{"campaignOnlyVersion":1,"timestamp":"1612849000","journal":{"metadata":
{"standardPlayTime":272.999725,"gameResult":1,"saveTime":"2021-02-
09T13:36:36.552186Z", "remainingCost":99, "remainingLifePoint":3, "killedEnemiesCnt":57, "mi
ssedEnemiesCnt":0,"levelId":"Activities/act16d5/level_act16d5_10","stageId":"act16d5_10","v
alidKilledEnemiesCnt":57}, "squad":
[{"charInstld":8,"skinId":"char 2015 dusk#1","tmplld":null,"skillId":"skchr dusk 1","skillIndex
":0,"skillLvl":1,"level":1,"phase":0,"potentialRank":0,"favorBattlePhase":38,"isAssistChar":true}
],"logs":[{"timestamp":0,"signiture":
{"uniqueld":2147483815,"charld":"char 2015 dusk"},"op":0,"direction":1,"pos":
{"row":12,"col":12}},{"timestamp":0,"signiture":
{"uniqueId":2147483815,"charId":"char_2015_dusk"},"op":0,"direction":1,"pos":
{"row":12,"col":13}},{"timestamp":0,"signiture":
{"uniqueId":2147483815,"charId":"char_2015_dusk"},"op":0,"direction":1,"pos":
{"row":12,"col":14}},{"timestamp":0,"signiture":
{"uniqueId":2147483815,"charId":"char_2015_dusk"},"op":0,"direction":1,"pos":
{"row":12,"col":15}},{"timestamp":0,"signiture":
```

后文内容高度相似, 仅有 row 和 col 字段后的数字不同。结合题中 出题人用可部署单位画了个东西 背景是 百色的 , 猜测是一个黑白图像, 两个字段对应的像素点为黑色、背景为白色。写一个c语言程序提取并绘制

```
#include<stdio.h>
int main(void)
   int pixel[100][100] = { 0 };
   int a, b;
   while (freopen("default_entry_mid.txt", "r", stdin) == NULL)
                                           //输入重定向, 命令行输入长度上限不够
   for (int i = 0; i < 10000; i++)
       scanf("{\"timestamp\":0,\"signiture\":
\ \"uniqueId\":2147483815,\"charId\":\"char_2015_dusk\"},\"op\":0,\"direction\":1
,\"pos\":{\"row\":%d,\"col\":%d}},", &a, &b);
       pixel[a][b] = 1;
                                         //default_entry开头的内容删除后作为输入
(偷个懒)
   }
   printf("\n\n\n");
   for (int row = 0; row < 100; row++) { //绘制100*100图像
       for (int col = 0; col < 100; col++)
           if (pixel[row][col])
              printf(" ");
                                            //cmd背景为黑色,打印空格即可
           else
              printf("龘");
                                            //复杂一点的字,白色面积更大一些
       printf("\n");
   }
   return 0;
}
```

保存、运行。得到一个二维码, 扫码后得到Flag。

accuracy

解压下载的压缩包,分析后得出:用 dataset.csv 的数据训练模型,将 chars.rar 中图片形式的手写数字转换为文本。一番操作后,放弃了 TensorFlow 和 百度AI开放平台 ,改为自行训练人工智能。Al速率约8bps;功耗三顿饭/十六小时。下为工作截图

■ *SHOUDA.txt - 记事本				- 🗆 X
文件(F) 編輯(E) 格式(O) 查看(V) 帮助(H) 24E00676F6B4C4E00904D4E09613F5982540C68814 E0A71D55C815C81957F76F889C121908FD953E5572				^
86700540F4E006BB576F463A567095F15752852309 F5365F6518D58FB5E386ECB547390FD5C16904D7EB 351705BB982E56D636EAA6C998C015FF5897F98CE7				
2EC81EA51C98D4C4E666D485F97FCFC			第 180 行,	第 32 列 100% Windows (CRLF) UTF-8
F 5 3 6 5	F 6 5	oping 7401:ping 7402:ping 7403:ping / 8 D S	B F B 5	E 3 8 6 E
7413.png 7414.png 7415.png 7416.png 7417.png		1.png 7422.png 7423.png 7424.png		29.png 7430.png 7431.png 7432.png 7433.png
7434.png 7435.png 7436.png 7437.png 7438.png	7439.png 7440.png 7441.png 7442	2.png 7443.png 7444.png 7445.png	7446.png 7447.png 7448.png 7449.png 74	50.png 7451.png 7452.png 7453.png 7454.png
		9 6 2 E 3.png 7464.png 7465.png 7466.png	7467.png 7468.png 7469.png 7470.png 747	71.png 7472.png 7473.png 7474.png 7475.png
6 C 9 Q S 7476.png 7477.png 7478.png 7479.png 7480.png		5 F F 5 4.png 7485.png 7486.png 7487.png		9 8 C E 7 92.png 7493.png 7494.png 7495.png 7496.png
9 E 6 8 1		1 C 9 8		E 6 6 6 D
7497.png 7498.png 7499.png 7500.png 7501.png		5.png 7506.png 7507.png 7508.png		13.png 7514.png 7515.png 7516.png 7517.png 9 9 5 F S
7518.png 7519.png 7520.png 7521.png 7522.png		6.png 7527.png 7528.png 7529.png		34.png 7535.png 7536.png 7537.png 7538.png
7539.png 7540.png 7541.png 7542.png 7543.png	7544.png 7545.png 7546.png 7547	7.png 7548.png 7549.png 7550.png	7551.png 7552.png 7553.png 7554.png 755	55.png 7556.png 7557.png 7558.png 7559.png
5 € 3 8 4 7560.png 7561.png 7562.png 7563.png 7564.png		(1) E (2) 7 8.png 7569.png 7571.png		4 E O D 5 76.png 7577.png 7578.png 7579.png 7580.png
3		C 2 3 6 9.png 7590.png 7591.png 7592.png		Y A 9 Y € 97.png 7598.png 7599.png 7600.png 7601.png
0 A 5 4 0	A 4 E C	0 0 7 6	7 - 5 &	7 4 4 6 0
7602.png 7603.png 7604.png 7605.png 7606.png		0.png 7611.png 7612.png 7613.png		18.png 7619.png 7620.png 7621.png 7622.png
7623.png 7624.png 7625.png 7626.png 7627.png	7628.png 7629.png 7630.png 7631	1.png 7632.png 7633.png 7634.png	7635.png 7636.png 7637.png 7638.png 763	39.png 7640.png 7641.png 7642.png 7643.png
7644.png 7645.png 7646.png 7647.png 7648.png	7649.png 7650.png 7651.png 7652	2.png 7653.png 7654.png 7655.png	7656.png 7657.png 7658.png 7659.png 766	60.png 7661.png 7662.png 7663.png 7664.png
3 C 8 5 9 7665.png 7666.png 7669.png 7669.png	7670.png 7671.png 7672.png 7673	5 5 5 3.png 7674.png 7675.png 7676.png		9 6 4 E 81.png 7682.png 7683.png 7684.png 7685.png
5 F 7 1 6		9 4 4 8		6 2 4 E 2
7686.png 7687.png 7688.png 7689.png 7690.png		4.png 7695.png 7696.png 7697.png		9 6 8 9 7
7707.png 7708.png 7709.png 7710.png 7711.png 阻 900 字节	7712.png 7713.png 7714.png 7715	5.png 7716.png 7717.png 7718.png	7719.png 7720.png 7721.png 7722.png 772	23.png 7724.png 7725.png 7726.png 7727.png I∷

全部打完后核对字节数,并上传,成功拿到Flag。