
SECCON 2017 × CEDEC CHALLENGE

ゲームクラッキング & チートチャレンジ

調査結果

— Harekaze —

目次

- Exercise1: HelloWorld
- Exercise2: Climb up
- CHUNI MUSIC

Exercise1: HelloWorld

Exercise1: HelloWorld - 概要

- Android, macOS 向けの Cocos2d-x 製ゲーム
- 目的: 2 億回タップすると表示される文字列を見つける

Exercise1: HelloWorld - 方針

- 自動化して 2 億回タップする
 - (連打ツール等では時間が掛かり非現実的)
- 実行中にメモリに変更を加える
 - タップ数を増やす
- **実行ファイルに変更を加える**
 - **必要なタップの回数を減らす**
 - タップ数の初期値を増やす
- 実行ファイルを解析する
 - 2 億回タップした時の文字列の表示処理を探す
 - strings のようなコマンドで文字列を探す

Exercise1: HelloWorld - 攻略

- macOS 向けの実行ファイルを objdump で逆アセンブル
- 2 億回タップしたかのチェック処理を探す
 - 現在のタップ数と 2 億 (0xbebc200) が比較されているはず？

Exercise1: HelloWorld - 攻略

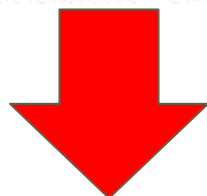
```
$ objdump -d -M intel -C HelloWorld-desktop
...
00000000100002864 <HelloWorld::onTouchBegan(cocos2d::Touch*, cocos2d::Event*)>:
...
    1000028b7:    81 3d 5f 2f 41 00 00    cmp     DWORD PTR [rip+0x412f5f],0xbcbc200
    1000028be:    c2 eb 0b               jle     1000028c1
    1000028c1:    7c 4a                   jle     10000290d
...
    10000290d:    b0 01                 mov     al,0x1
    10000290f:    48 83 c4 18           add     rsp,0x18
    100002913:    5b                     pop     rbx
    100002914:    41 5e                 pop     r14
    100002916:    41 5f                 pop     r15
    100002918:    5d                     pop     rbp
    100002919:    c3                     ret
```

タップ数が 2 億回未満なら
1 を戻り値として return

Exercise1: HelloWorld - 攻略

- バイナリエディタで、タップ数と比較されている値を変えてしまう

000028B0	7D	D0	E8	BF	77	2C	00	81	3D	5F	2F	41	00	00	C2	EB
000028C0	0E	7C	4A	48	8B	1D	66	2F	41	00	48	8B	03	4C	8B	B8



000028B0	7D	D0	E8	BF	77	2C	00	81	3D	5F	2F	41	00	01	00	00
000028C0	00	7C	4A	48	8B	1D	66	2F	41	00	48	8B	03	4C	8B	B8

Exercise1: HelloWorld - 攻略

```
$ objdump -d -M intel -C HelloWorld-desktop
...
00000000100002864 <HelloWorld::onTouchBegan(cocos2d::Touch*, cocos2d::Event*)>:
...
1000028b7: 81 3d 5f 2f 41 00 01      cmp     DWORD PTR [rip+0x412f5f],0x1
1000028be: 00 00 00                  jle     1000028c1
1000028c1: 7c 4a                     jle     10000290d
...
10000290d: b0 01                     mov     al,0x1
10000290f: 48 83 c4 18               add     rsp,0x18
100002913: 5b                         pop     rbx
100002914: 41 5e                     pop     r14
100002916: 41 5f                     pop     r15
100002918: 5d                         pop     rbp
100002919: c3                         ret
```

タップ数が 1 回未満なら
1 を戻り値として return

Exercise1: HelloWorld - 攻略

- 変更を加えた実行ファイルを実行すると...



Thank you for counting up!

Exercise1: HelloWorld - デモ



<https://youtu.be/YJYGRo67XB8>

Exercise2: Climb up

Exercise2: Climb up - 概要

- Android, Windows 向けの Unreal Engine 4 製ゲーム
- 目的: 通常のプレイでは登れないブロックに登る



Exercise2: Climb up - 方針

- 実行中にメモリに変更を加える
 - プレイヤーの座標をブロックの上に移動させる
- パッケージ化されたプロジェクトに変更を加える
 - ジャンプ時の速度を変える
 - ブロックの位置を変える
 - プレイヤーの初期位置を変える

Exercise2: Climb up - 攻略

- Windows 向けの実行ファイルを調査する
- どのようなファイルやフォルダがあるか調べる

Exercise2: Climb up - 攻略

- ClimbUp.exe
 - 本体、実行するとゲームが起動する
- Engine\
 - Binaries\ThirdParty\
 - PhysX、OpenVR などサードパーティのライブラリ
 - Binaries\Win32\UE4Game.exe
 - Saved\
- **ClimbUp**
 - **Content\Paks\ClimbUp-WindowsNoEditor.pak**
 - Saved\

Exercise2: Climb up - 攻略

- ClimbUp-WindowsNoEditor.pak
 - パッケージ化された UE4 のプロジェクト
 - panzi/u4pak [\[1\]](#) を使って展開ができる
 - 現在はメンテナンスされておらず、そのままでは使えない
 - 以下のように変更を加えると展開できる

```
@@ -655,7 +655,7 @@
...
-     elif version == 3:
+     elif version == 3 or version == 4:
         read_record = read_record_v3
```

[1] <https://github.com/panzi/u4pak>

Exercise2: Climb up - 攻略

- 展開されたファイルやフォルダを調べる
- Engine\
 - **ClimbUp\Content\
 - 2DSideScroller\
 - **2DSideScrollerBP\
 - **Blueprints\
 - **2DSideScrollerCharacter.uasset**
 - **2DSideScrollerCharacter.uexp****
 - **Maps\
 - **2DSideScrollerCharacter.uasset**
 - **2DSideScrollerCharacter.uexp********

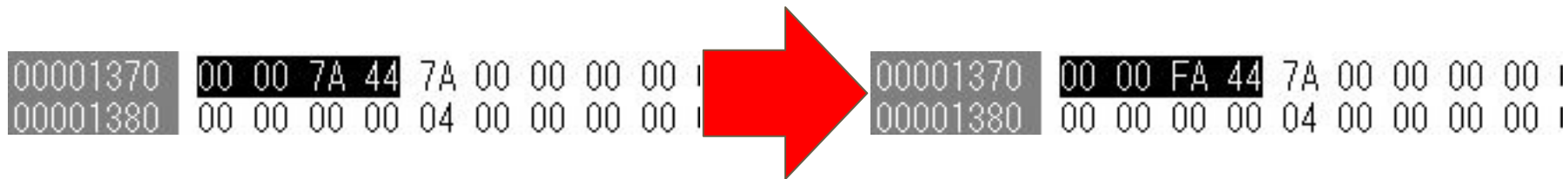
Exercise2: Climb up - 攻略

- 2DSideScrollerCharacter.{uasset,uexp} にブループリントの情報が格納されている?
- 2DSideScrollerCharacter.uasset に含まれる文字列を探す
 - **JumpZVelocity** (ジャンプ時の Z 方向の速度) の値を変えれば高くジャンプできるようになるのでは？

```
$ strings -a 2DSideScrollerCharacter.uasset
...
Jump
JumpZVelocity
K2_GetActorLocation
...
```

Exercise2: Climb up - 攻略

- JumpZVelocity の値が 1000 と当たりをつける
- バイナリエディタを使って 2DSideScrollerCharacter.uexp で 1000 (単精度浮動小数点数で表現すると **00 00 7a 44**) を探す
- これを 2000 (単精度浮動小数点数で表現すると **00 00 fa 44**) に変えてしまう



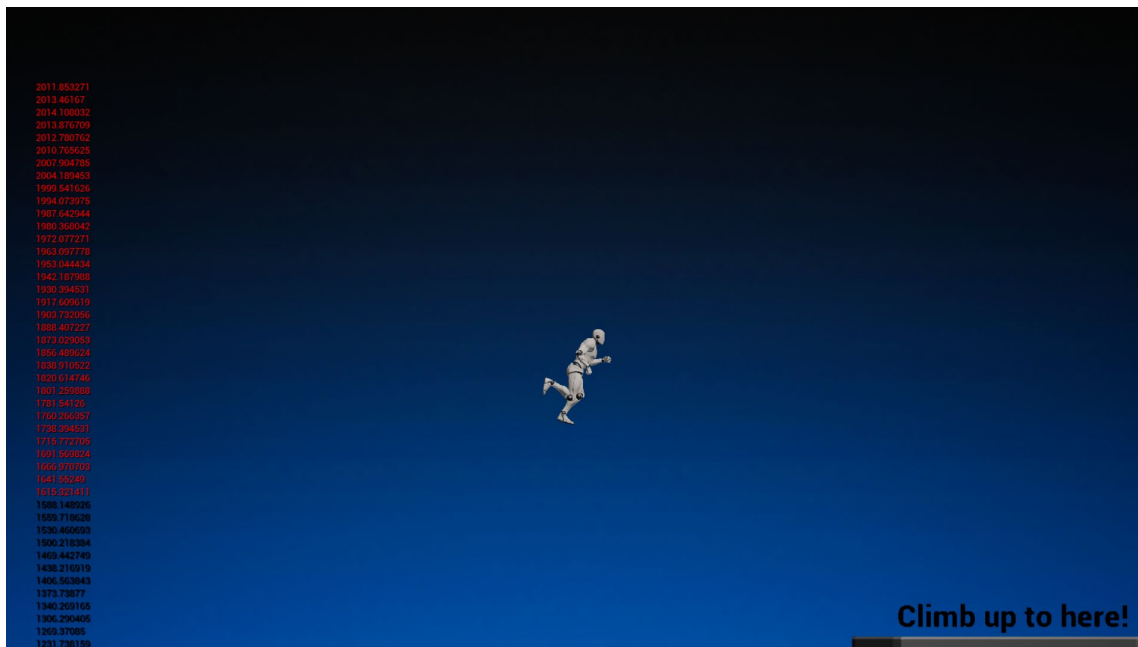
Exercise2: Climb up - 攻略

- JumpZVelocity が 2000 に変更された状態の ClimbUp-WindowsNoEditor.pak を作る必要がある
- 展開したフォルダで再度パッケージ化する

```
$ python2 u4pak.py pack ClimbUp-WindowsNoEditor.pak ClimbUp Engine
```

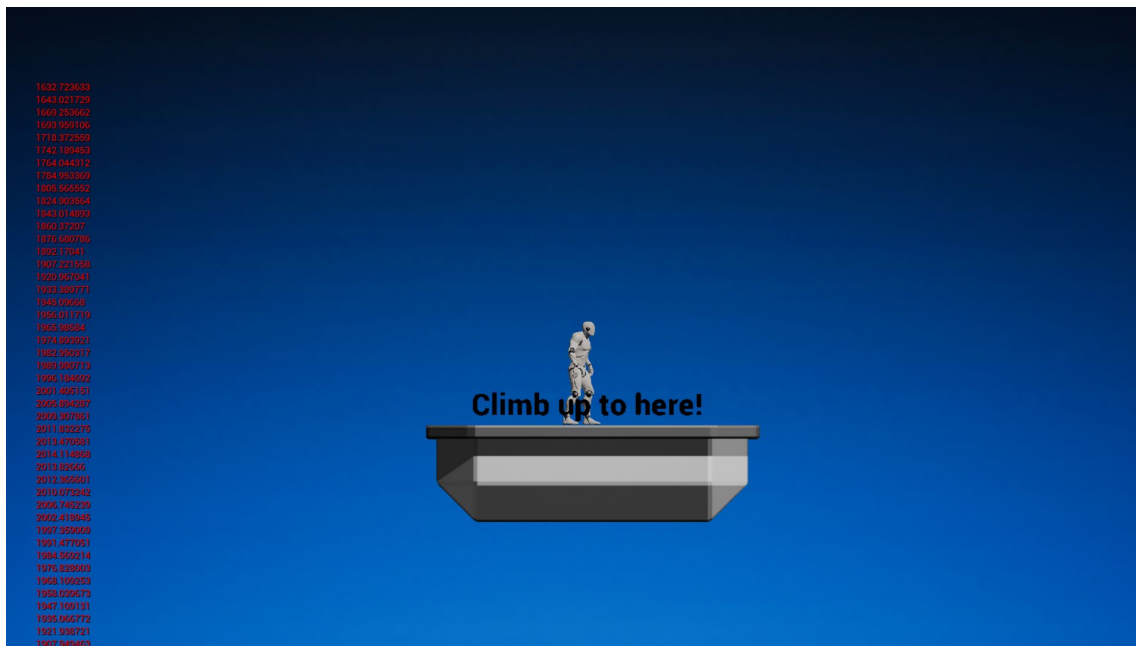
Exercise2: Climb up - 攻略

- ClimbUp.exe を実行すると...

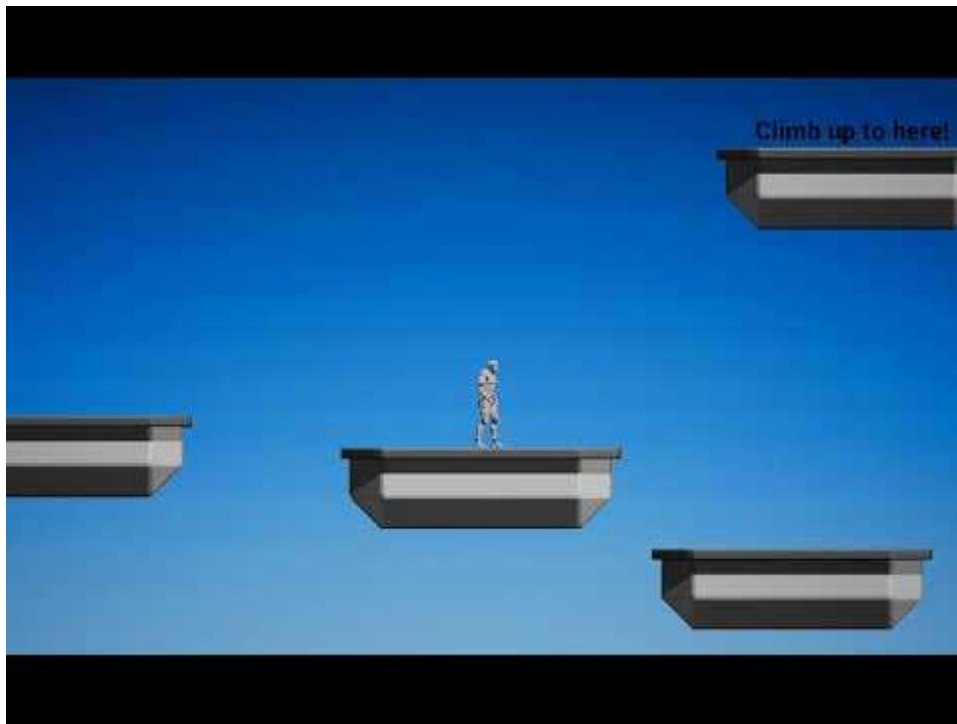


Exercise2: Climb up - 攻略

- ClimbUp.exe を実行すると...

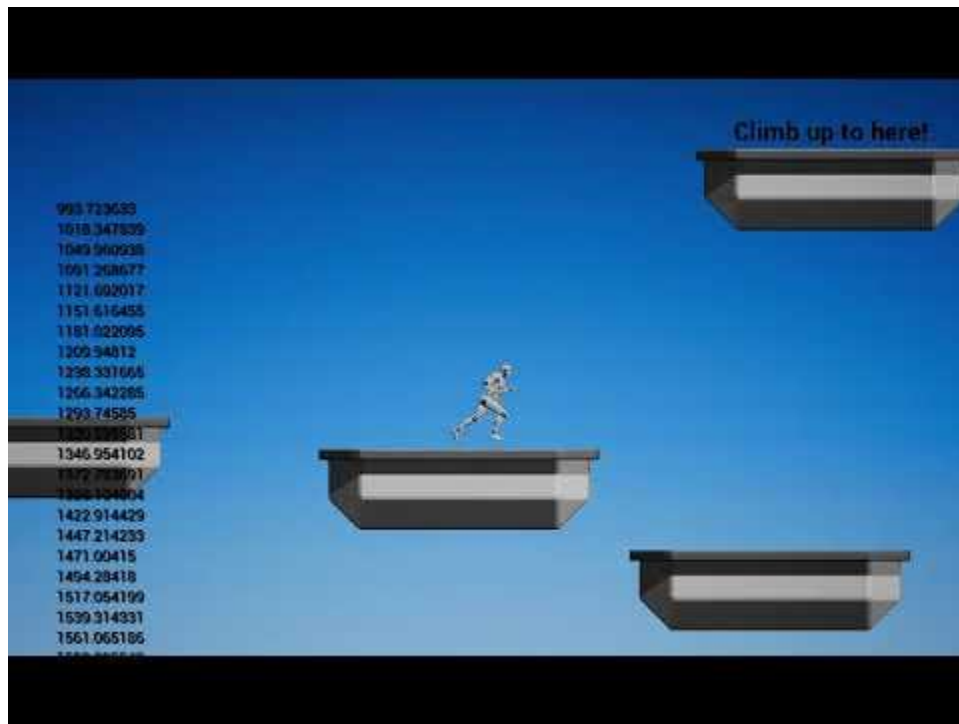


Exercise2: Climb up - デモ (変更前)



<https://youtu.be/aN0L2ZEozz4>

Exercise2: Climb up - デモ (変更後)



<https://youtu.be/Ee0cdNQfwQg>

Exercise2: Climb up - 攻略

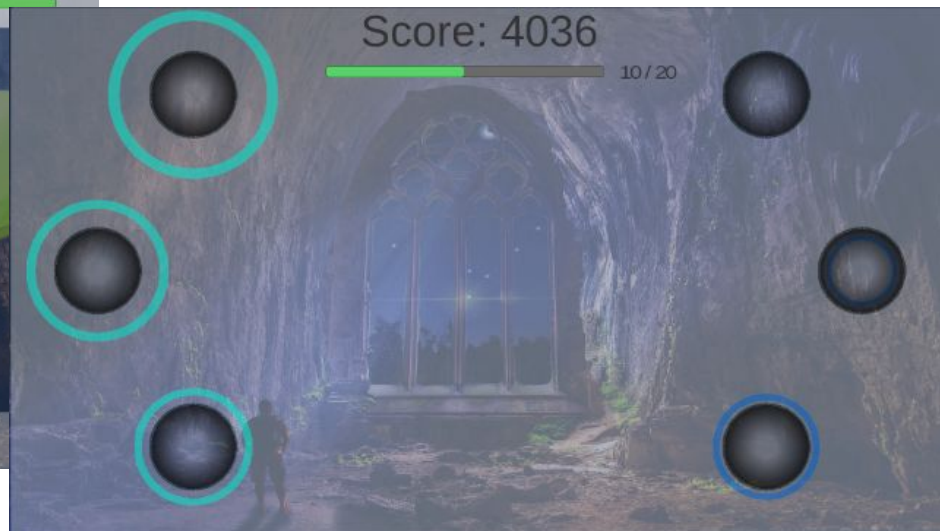
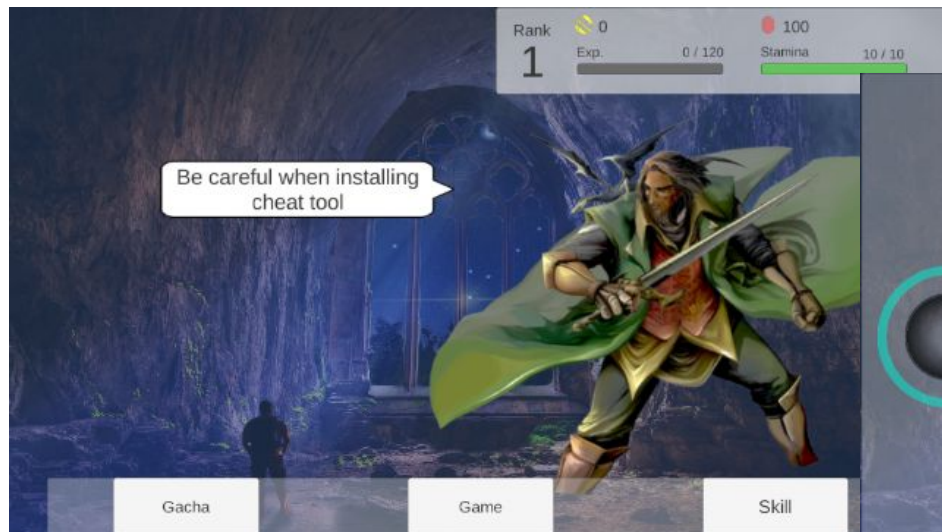
- ジャンプ時の速度を上げすぎるとバグる



CHUNI MUSIC

CHUNI MUSIC - 概要

- Android 向けの Unity 製ゲーム
- 目的: セキュリティ上の問題点を調査し、対策案を提案する



CHUNI MUSIC - 検証環境

- Genymotion 2.10.0 (Android 5.1.0)
 - Android エミュレータ
- Android Studio 2.3.3
- mitmproxy 0.18.2
 - プロキシ
- apktool 2.2.4
 - apk ファイルの展開・再パッケージ化ツール

CHUNI MUSIC - 各問題点の評価

- 他ユーザへの影響度、実行の容易性などを基準に
3段階 (Low < Medium < High) で危険度の評価を行う

問題点	内容	危険度
問題点1	中間者攻撃に対する脆弱性	Medium
問題点2	リクエスト改ざんによる任意回数のガチャの実行	Low
問題点3	ランキングへの不正なスコアの登録	High
問題点4	スタミナの任意タイミングでの回復	High

CHUNI MUSIC - 問題点 1 の概要

- 中間者攻撃に対して脆弱
- 端末に不正なルート証明書をインストールさせることで、このルート証明書が発行し、署名した証明書を信頼させることができる
 - この証明書の発行者が端末とサーバの間に入り込んで通信を盗聴
 - ⇒ SSL/TLS でサーバと通信していても解読できてしまう



攻撃対象のユーザ

攻撃者

サーバ

CHUNI MUSIC - 問題点 1 の手法

- mitmproxy ^[1] や Fiddler ^[2] のようなプロキシを使う
(今回は mitmproxy を使用する)
- 攻撃者がプロキシを起動する
- 攻撃対象の端末の設定を行う
 - 通信がプロキシを経由するように設定する
 - プロキシによって発行されたルート証明書をインストールする
- 攻撃対象の端末でゲームを起動すると...

[1] <https://mitmproxy.org/>

[2] <http://www.telerik.com/fiddler>

CHUNI MUSIC - 問題点 1 の手法

mitmproxy Start View Options Flow

▶ Accept ◀ Replay ⏏ Duplicate 🗑 Delete ↺ Revert ⬇ Download

Path	Method	Status	Size	Time
https://cedec.secon.jp/2017/key	POST	200	239b	339ms
http://connectivitycheck.android.com/generate_204	GET	204	0	114ms
https://cedec.secon.jp/2017/key	POST	200	241b	311ms
https://api.uca.cloud.unity3d.com/v1/events	POST	200	394b	280ms
https://api.uca.cloud.unity3d.com/v1/events	POST	200	347b	137ms
https://config.uca.cloud.unity3d.com/80588224-e2e...	GET	200	2b	340ms
https://cedec.secon.jp/2017/skill	GET	200	169b	76ms
https://cedec.secon.jp/2017/account	GET	200	343b	77ms
https://cedec.secon.jp/2017/key	POST	200	239b	72ms
https://cedec.secon.jp/2017/skill	GET	200	169b	67ms
https://cedec.secon.jp/2017/account	GET	200	342b	82ms

Request Response Details

HTTP/1.1
200
OK

Date: Thu, 10 Aug 2017 05:20:51 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Signature: 8b1bde4a20ed425afb7c6067c10530e6c5e3c3bc174efb7fbc
a034dd536e9cc6
Set-Cookie: token="!VBRkzfqqdVdGqtKrH9joEw==?gAJVBXRva2VucQFVI
HlxTgt0M1pLRU1NWecxOXYxTXJheFhtNGI4bTdIz1JxcQKGcQM
u"
Vary: Accept-Encoding
Content-Encoding: gzip
Keep-Alive: timeout=5, max=98
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

textkUM0D0LS4hqquad8nXc9+hb1lEdD9xnwsp8xgwcylu1YIB4bQbv7ab0yjQPhmlwVvKX5LPgfzBCa

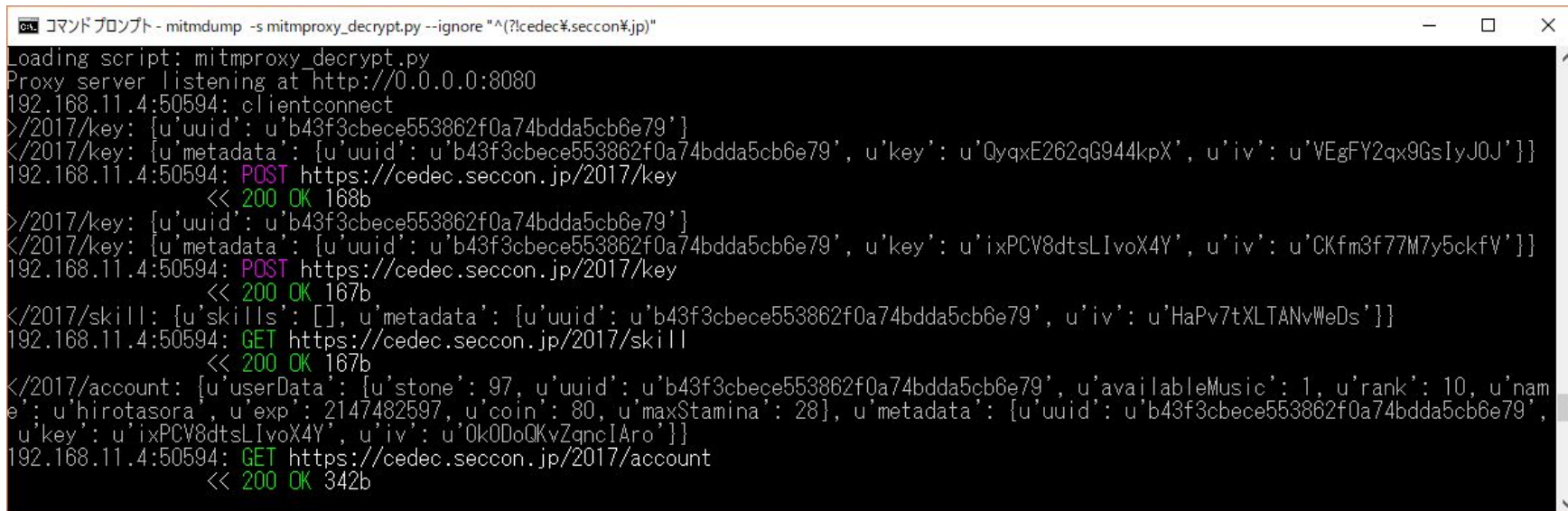
View: Auto [decoded gzip] Couldn't parse: falling back to Raw

CHUNI MUSIC - 問題点 1 の手法

- SSL/TLS の通信が復号できた
- しかし、アプリケーション側でも独自に暗号化が行われているため、このままではどのような通信が行われているか分からない
 - 通信を観察するとある程度暗号化の方式が分かる
 - ⇒ データサイズの変化が 16 バイト単位なので、ブロック暗号 (AES, DES など) と推測可能
 - apk ファイルを解析すると鍵と IV が分かる
 - ⇒ assets/bin/Data/Managed/Metadata/global-metadata.dat に平文で存在

CHUNI MUSIC - 問題点 1 の手法

- 自動的に通信の復号を行うスクリプト [\[1\]](#) を作成すると...



```
コマンドプロンプト - mitmdump -s mitmproxy_decrypt.py --ignore "^(?!cedec%.secon%.jp)"
Loading script: mitmproxy_decrypt.py
Proxy server listening at http://0.0.0.0:8080
192.168.11.4:50594: clientconnect
>/2017/key: {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79'}
</2017/key: {u'metadata': {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79', u'key': u'Qyqx262qG944kpX', u'iv': u'VEgFY2qx9GsIyJ0J'}}
192.168.11.4:50594: POST https://cedec.secon.jp/2017/key
<< 200 OK 168b
>/2017/key: {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79'}
</2017/key: {u'metadata': {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79', u'key': u'ixPCV8dtsLIvoX4Y', u'iv': u'CKfm3f77M7y5ckfV'}}
192.168.11.4:50594: POST https://cedec.secon.jp/2017/key
<< 200 OK 167b
</2017/skill: {u'skills': [], u'metadata': {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79', u'iv': u'HaPv7tXLTANvWeDs'}}
192.168.11.4:50594: GET https://cedec.secon.jp/2017/skill
<< 200 OK 167b
</2017/account: {u'userData': {u'stone': 97, u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79', u'availableMusic': 1, u'rank': 10, u'nam
e': u'hirotasora', u'exp': 2147482597, u'coin': 80, u'maxStamina': 28}, u'metadata': {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79',
u'key': u'ixPCV8dtsLIvoX4Y', u'iv': u'0k0DoQKvZqncIAro'}}
192.168.11.4:50594: GET https://cedec.secon.jp/2017/account
<< 200 OK 342b
```

[1] <https://gist.github.com/st98/e6f17c9fd574ff264a8173d4b651767a>

CHUNI MUSIC - 問題点 1 のデモ



<https://youtu.be/PGL6ImuB7DI>

CHUNI MUSIC - 問題点 1 の影響度

- 中間者攻撃によって通信の盗聴や改ざんが可能
- UUID を盗聴することによってなりすましができる
 - 勝手にハイスコアを登録したり、ガチャを引いたりできる
- 端末間でのアカウント共有や課金のような機能が増えると、例えばメールアドレスやパスワード、個人情報といった、ゲーム内で完結しない重要な情報の盗聴も考えられる

CHUNI MUSIC - 問題点 1 の影響度

- 攻撃のシナリオを考える
- 攻撃対象の端末が悪意のあるアクセスポイントに接続
 - ⇒ ログインページなどを使って
不正なルート証明書をインストールするように誘導
 - ⇒ ユーザがゲームを起動、通信の盗聴や改ざんが可能になる
- **必ず修正を行いたい**

CHUNI MUSIC - 問題点 1 の対策案

- 証明書の Pinning (ピン留め) を行う
 - 証明書チェーンの検証とは別に、アプリケーション内に証明書の情報 (拇印など) を埋め込んで検証を行う
- これによって、不正なルート証明書がインストールされて不正な証明書が発行された場合にもそれを検知、利用を防止できる
- 自発的に解析を行うような場合では完全に防ぐことはできないが、通信の解読を難しくすることができる (= 時間稼ぎができる)

CHUNI MUSIC - 問題点 1 の対策案

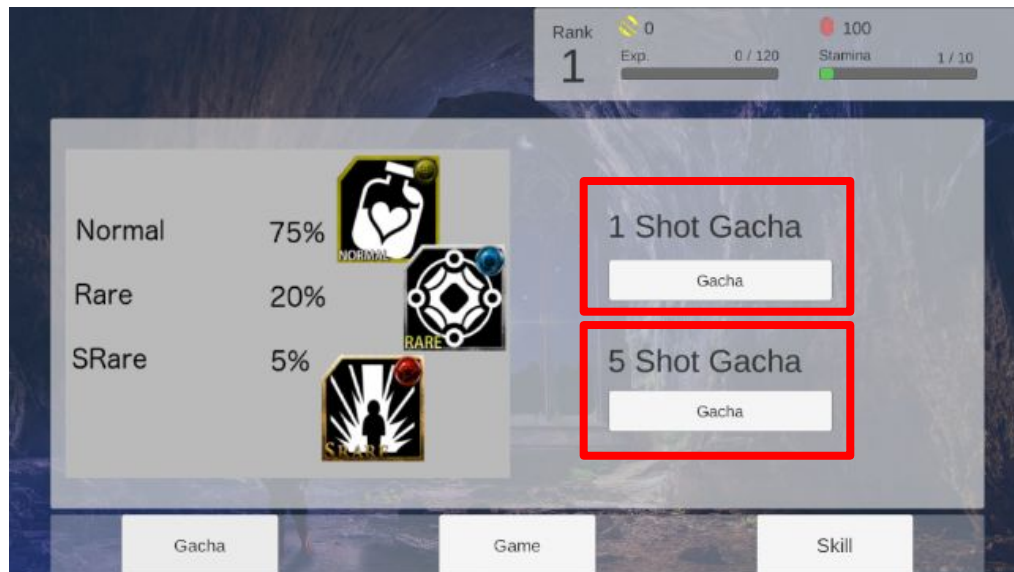
- 初期状態の鍵と IV を分かりにくいものにする
 - 鍵が def4ul7KeY1Z3456 と K33pK3y53cr3TYea の排他的論理和、IV が IVisNotSecret123 と分かりやすいものだった
 - これをランダムなバイト列にすると、
global-metadata.dat を見るだけではそれと分かりにくくなる
- 通信ごとに暗号化に使う鍵を変更する
 - apk ファイルの中に暗号化に使われる鍵と IV が保存されており、通信が容易に復号可能
 - ⇒ DH 鍵共有のようなアルゴリズムで鍵を共有することで、通信の解読を難しくすることができる

CHUNI MUSIC - 問題点 2 の概要

- 任意の回数ガチャが引けてしまう
 - 本来は 1 回か 5 回しか一度にガチャを引くことができないが、細工したリクエストを送ることで、指定した回数だけガチャを引ける

CHUNI MUSIC - 問題点 2 の手法

- 問題点 1 の手法を用いて通信の復号を行う
- ガチャを引いたときの通信を観察する



CHUNI MUSIC - 問題点 2 の手法

- 1 Shot Gacha を選択した場合

```
>/2017/gacha: {u'gacha': 1}
</2017/gacha: {"gacha":1} を POST
u'pe': 3, u'id': 7, u'param': 200,
u'name': u'Bas a': {u'uuid':
u'b43f3cbece553862f0a74bdda5cb6e79', u'ix': u'085BDVCErFbhYDa0'
192.168.11.4:51108: POST https://cedec.s
<< 200 OK 232b
```

ガチャを引いた結果が
JSON 形式で 1 つ返ってくる

CHUNI MUSIC - 問題点 2 の手法

- 5 Shot Gacha を選択した場合

```
>/2017/gacha: {u'gacha': 5}
</2017/gacha: {"gacha":5} を POST
u'StaminaUpS', u'skillType': 2, u'id': 4, u'param': 1, u'name':
u'ComboStaminaCureM', u'skillType': 4, u'id': 10,
u'param': 1, u'combo': 15}, {u'skillType': 2, u'id': 4, u'param': 1,
u'name': u'StaminaUpS'}, {u'name': u'ComboStaminaCureM', u'skillType': 4,
u'id': 11, u'param': 2, u'combo': 20}, {
u'skillType': 1, u'id': 1, u'param': 100
{u'uuid': u'b43f3cbece553862f0a74bdda5cb
192.168.11.4:51150: POST https://cedec.secon.jp/2017/gacha
<< 200 OK 534b
```

ガチャを引いた結果が
JSON 形式で 5 つ返ってくる

CHUNI MUSIC - 問題点 2 の手法

- {"gacha":3} を POST できないか考える
- データの暗号化は問題点 1 の手法を応用することで可能
- ただ、そのまま POST しても何も起こらない
 - HTTP ステータスコードを確認すると 500 となっており、サーバ側でエラーが発生していることが分かる

CHUNI MUSIC - 問題点 2 の手法

- 正規の HTTP リクエストを観察してみると、
POST 時には **X-Signature** ヘッダが付与されていることが分かる
(例: b88f4248c9fc5ac7dd14cf9f8532185877b39a13842c25de9f06d3933ae2aeaf)
- 改ざんのチェック用として、データに署名したものと考えられる
⇒ 長さは 32 バイトなので SHA-256?
- 署名には HMAC-SHA256 が使われていると考えて秘密鍵を探す
⇒ assets/bin/Data/Managed/Metadata/global-metadata.dat に
平文で存在
- 得られた秘密鍵を使って正規の HTTP リクエストのデータを署名すると、
X-Signature ヘッダの値と一致
- これで任意のデータについて X-Signature ヘッダの値を計算できる

CHUNI MUSIC - 問題点 2 の手法

- 正規の HTTP リクエスト/レスポンスを観察してみると、どれも token というクッキーが付与されていることが分かる
- これはデータによって変わることはないので、そのまま使える

CHUNI MUSIC - 問題点 2 の手法

- クッキー、X-Signature ヘッダを付与した上で
再度 {"gacha":3} の POST を試みるスクリプト [\[1\]](#) を作成する

```
$ python2 gacha.py (UUID)
[{"name": "ComboStaminaCureS", "skillType": 4, "id": 10, "param": 1,
"combo": 15}, {"skillType": 3, "id": 7, "param": 200, "name":
"BaseScoreUpS"}, {"skillType": 5, "id": 13, "param": 1, "name":
"JudgeImprove"}]
```

ガチャを引いた結果が
JSON 形式で 3 つ返ってきた!

[1] <https://gist.github.com/st98/436a4972a36a811164bbf75127efde49>

CHUNI MUSIC - 問題点 2 の影響度

- 本来メニューからは選択できない回数ガチャを引くことができる
- 時間はかかるが、消費されるダイヤ石の数は変わらないため
結局は 1 回ずつガチャを引いた場合と同じ結果になる
- また、負数や所有しているダイヤ石で引ける以上の回数を指定しても、
ガチャは引かれなかったため、不正にガチャを引くことはできない
- **ゲームに影響はないが、できれば修正したい**

CHUNI MUSIC - 問題点 2 の対策案

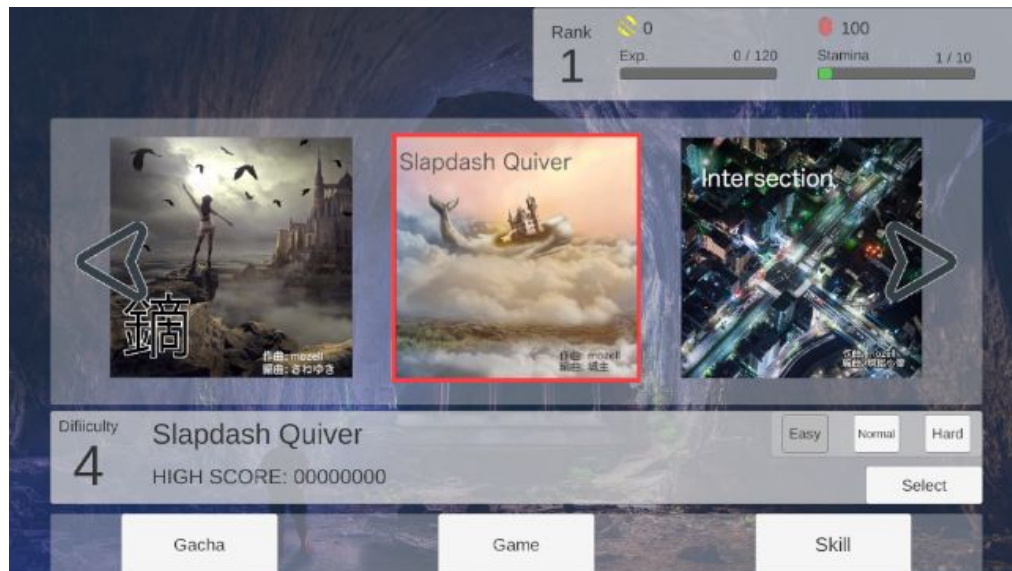
- サーバ側でガチャの回数のチェックを行う
 - メニューに存在しない回数であれば無効とする

CHUNI MUSIC - 問題点 3 の概要

- 不正なスコアをランキングに登録できてしまう
 - 通常のプレイでは不可能な高いスコアに登録ができる
 - difficulty (難易度) や musicId (楽曲の番号) に任意の値を指定できる
 - 現在は公開されていない楽曲のスコアに登録ができる

CHUNI MUSIC - 問題点 3 の手法

- 問題点 1, 2 の手法を応用する
- 楽曲をクリアした際の通信を観察する



CHUNI MUSIC - 問題点 3 の手法

- Slapdash Quiver の Easy をクリアした場合



CHUNI MUSIC - 問題点 3 の手法

- Slapdash Quiver の Easy をクリアした場合

```
>/2017/score: {u'myScore': {u'difficulty': 0, u'uuid':  
u'9f4321e6f7196018ef08c14ef3f9b715', u'score': 61726, u'musicId': 1,  
u'name': u''}}  
</2017/score: {u'myScore': {u'difficulty': 0, u'uuid': u'9f4321e6f7196018ef08c14ef3f9b715', u'score': 61726, u'musicId': 1, u'name': u'hirotasora'}},  
u'metadata': {u'uuid': u'9f4321e6f7196018ef08c14ef3f9b715', u'iv':  
u'qDU4xOa6WapHUiNm'}}  
192.168.11.4:54477: POST https://cedec.secon.jp/2017/score  
    << 200 OK 197b
```

スコア、UUID、楽曲の ID と難易度を
JSON 形式で POST

CHUNI MUSIC - 問題点 3 の手法

- Slapdash Quiver の Easy をクリアした場合

```
>/2017/score: {u'myScore': {u'difficulty': 0, u'uuid':  
u'9f4321e6f7196018ef08c14ef3f9b715', u'score': 61726, u'musicId': 1,  
u'name': u''}}  
</2017/score: {u'gameScores': [{u'score': 61726, u'name': u'hirotasora'}],  
u'metadata': {u'uuid': u'9f4321e6f7196018ef08c14ef3f9b715', u'iv':  
u'qDU4xOa6WapHUiNm'}}  
192.168.11.4:54477: POST [REDACTED]  
<< 200 OK 197b
```

更新されたランキングが返ってくる

CHUNI MUSIC - 問題点 3 の手法

- スコアを大きな値にして POST できないか考える
- 正規のリクエストで使われた JSON のスコアを **123456789** に変えたものを、問題点 2 で作成したスクリプトを改変 [\[1\]](#) して POST を試みる

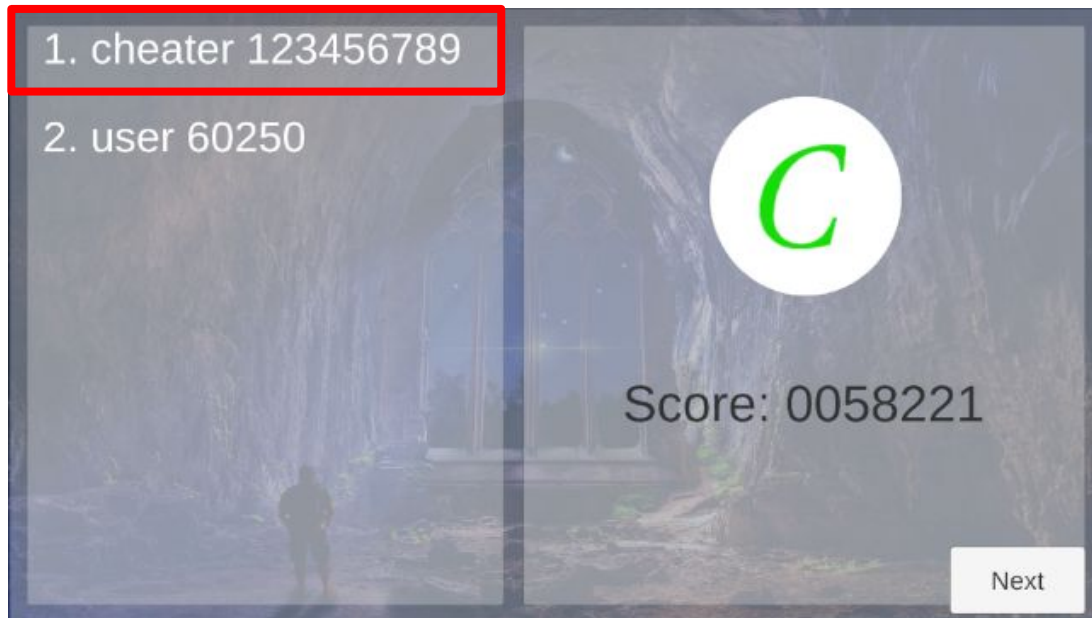
```
$ python2 cheat_score.py (UUID)
[{"score": 123456789, "name": "cheater"}]
```

スコアが **123456789** の記録が
ランキングに登録された!

[1] <https://gist.github.com/st98/b340b5bc84415597687d9cae42b15d1b>

CHUNI MUSIC - 問題点 3 の手法

- 別のユーザ (user) で同じ難易度の同じ楽曲をクリアしてみると...



CHUNI MUSIC - 問題点 3 の手法

- 楽曲の番号と難易度を 100 に変えたもの [1] を POST しても、スコアの登録は成功する

```
$ python2 cheat_score.py (UUID)
{"myScore": {"difficulty": 100, "uuid": "a50dd1c8f62e141754a01147c27362d2",
"score": 123456789, "musicId": 100, "name": ""}}
[{"score": 123456789, "name": "testdayo"}]
```

楽曲の番号と難易度が **100**、
スコアが **123456789** の記録が
ランキングに登録された!

[1] <https://gist.github.com/st98/1e6a3963e5122f715fbba75b746b6607>

CHUNI MUSIC - 問題点 3 の影響度

- 不正なスコアがランキングに登録されることで、
他のプレイヤーがゲームを継続するモチベーションが失われる
⇒ プレイヤーの離脱、収益の減少を招く
- 継続的に不正なスコアが登録されることで、
ランキングの信頼性、運営会社への信頼が失われる
- ゲームのバランスが著しく崩れる可能性があるため、必ず修正を行いたい

CHUNI MUSIC - 問題点 3 の対策案

- 楽曲のランキングを常に監視し、不正なスコアが登録されていればスコアやアカウントの削除を行う
 - 対症療法的、根本的な対策ではない
- もし通常のプレイでも可能な範囲で不正なスコアが登録されたら？
 - 現在の仕組みでは通常のプレイと見分けがつかない
 - タップの座標やタイミングなどを記録し、スコアをサーバ側で計算させる
 - 暗号化の鍵の難読化などを行い、リクエストの偽造のコストを大きくする
- サーバ側で楽曲の番号や難易度のチェックを行う
 - 現在は遊べない楽曲や難易度であれば無効とする

CHUNI MUSIC - 問題点 4 の概要

- SharedPreferences に変更を加えるだけで、
任意のタイミングでスタミナの回復が可能

CHUNI MUSIC - 問題点 4 の手法

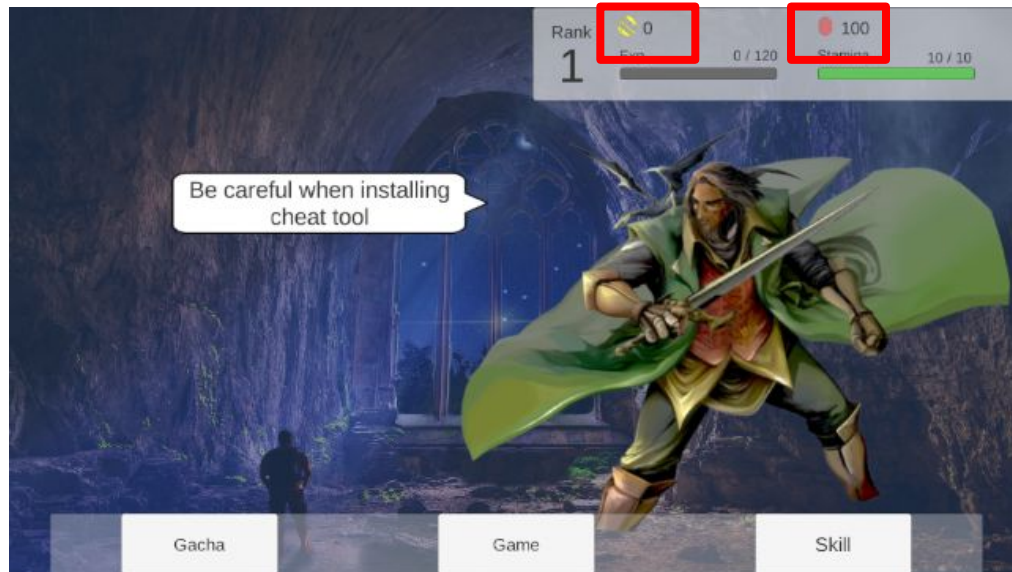
- スタミナがどのように管理されているか確認する
- 問題点 1 の手法を用いてゲーム起動時の通信を観察する

```
</2017/account: {u'userData': {u'stone': 26, u'uuid':  
u'b43f3cbece553862f0a74bdda5cb6e79', u'availableMusic': 1, u'rank': 12,  
u'nam GET /2017/account b': 2147482037, u'coin': 0, u'maxStamina': 32},  
u'metadadata': {u'uuid': u'b43f3cbece553862f0a74bdda5cb6e79', u'  
u'uZgQvaqsVi34KXYn', u'iv': u'b7OH1Q {..., "maxStamina":32}, "metadadata": {...}}  
192.168.11.4:54868: GET https://cedec << 200 OK 340b
```

スタミナの現在値のような値はなく、
最大値だけが与えられている

CHUNI MUSIC - 問題点 4 の手法

- コインかダイヤ石を消費してスタミナを回復できる
- 問題点 1 の手法を用いて回復時の通信を観察する



CHUNI MUSIC - 問題点 4 の手法

- コインを消費して回復した場合 (成功)

```
>/2017/useItem: {u'item': u'coin'}
</2017/useItem {u'item': u'coin'}
u'b43f3cbece55...': u'iQcJmR4kQtD0WKaF'}}
192.168.11.4:54779: POST https://.../2017/...
<< 200 OK 167 {"status":"ok","metadata":{"...}}
```

{“item”:”coin”} を POST

アイテムを使えるかどうか
JSON 形式で返ってくる

CHUNI MUSIC - 問題点 4 の手法

- コインを消費して回復した場合 (コインが足りず失敗)

(サーバとの通信は発生しない)

CHUNI MUSIC - 問題点 4 の手法

- ダイヤ石を消費して回復した場合 (成功)

```
>/2017/useItem: {u'item': u'stone'}  
</2017/useItem: {"item": "stone"} を POST  
u'b43f3cbece55...adata': {u'uuid':  
u'LKgBEcZxVIaNJ34g'}}  
192.168.11.4:54779: POST https://...  
<< 200 OK 166 {"status": "ok", "metadata": {...}}
```

アイテムを使えるかどうか
JSON 形式で返ってくる

CHUNI MUSIC - 問題点 4 の手法

- コインかダイヤ石を消費してスタミナを回復する際には、サーバに使用するアイテムを知らせて使用可能か調べるだけ
- スタミナの現在値はクライアント側で管理しているようなので、今回はこれを利用 (悪用) する

CHUNI MUSIC - 問題点 4 の手法

- スタミナが減った状態でゲームを終了し、
SharedPreferences (端末内にキーと値のペアで設定などを保存できる仕組み) を調べる
- `/data/data/com.totem.chuni_music/shared_prefs/com.totem.chuni_music.v2.playerprefs.xml` にデータが XML 形式で保存されている
- **adb pull** でこの XML を取り出す

CHUNI MUSIC - 問題点 4 の手法



楽曲をプレイしスタミナが減った状態

CHUNI MUSIC - 問題点 4 の手法

```
$ adb pull /data/.../shared_prefs/com.totem.chuni_music.v2.playerprefs.xml .
```

```
com.totem.chuni_music.v2.playerprefs.xml
```

```
encoding="utf-8" standalone="yes" ?>
```

端末上の XML を取り出す

```
<string name="unity.player_sessionid">1581708901398830045</string>
<string name="unity.cloud_userid">018378072fb9049f5995095f4c67dad4</string>
<string name="unity.player_session_background_time">1502410149107</string>
<string
name="uuid">VcXcMPyvBVidIRrakqx6sSFXogLppB2nHpI3S2YEmCbsPsRUfLzR5quhEjOEJQWV</string>
<int name="Screenmanager%20Resolution%20Height" value="480" />
<int name="stamina" value="6" />
<int name="Screenmanager%20Resolution%20Width" value="854" />
<int name="Screenmanager%20Resolution%20DPI" value="0" />
<string name="Screenmanager%20Resolution%20DPI" value="0" />
<int name="Screenmanager%20Resolution%20DPI" value="0" />
<string name="Screenmanager%20Resolution%20DPI" value="0" />
<string name="lastUpdated">08%2F11%2F2017%2000%3A44%3A01</string>
<string name="unity.player_session_elapsed_time">0</string>
</map>
```

スタミナが平文で保存されている
(現在の値は 6)

CHUNI MUSIC - 問題点 4 の手法

- スタミナをの値を 6 から 123456789 に書き変えて、
adb push を使って端末内の XML を置き換える

CHUNI MUSIC - 問題点 4 の手法

```
$ cat com.totem.chuni_music.v2.playerprefs.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <string name="unity.player_sessionid">1581708901398830045</string>
  <string name="unity.cloud_userid">018378072fb9049f5995095f4c67dad4</string>
  <string name="unity.player_session_background_time">1502410149107</string>
  <string
name="uuid">VcXcMPyvBVidIRrakqx6sSFXogLppB2nHpI3S2YEmCbsPsRUfLzR5quhEjOEJQWV</string>
  <int name="Screenmanager%20Resolution%20Height" value="480" />
  <int name="stamina" value="123456789" />
  <int name="Screenmanager%20Resolution%20Width" value="854" />
  <int name="Screenmanager%20Volume" value="100" />
  <string name="5" value="5" />
  <int name="__UNITY_PLAYERPREFS_VERSION__" value="1" />
```

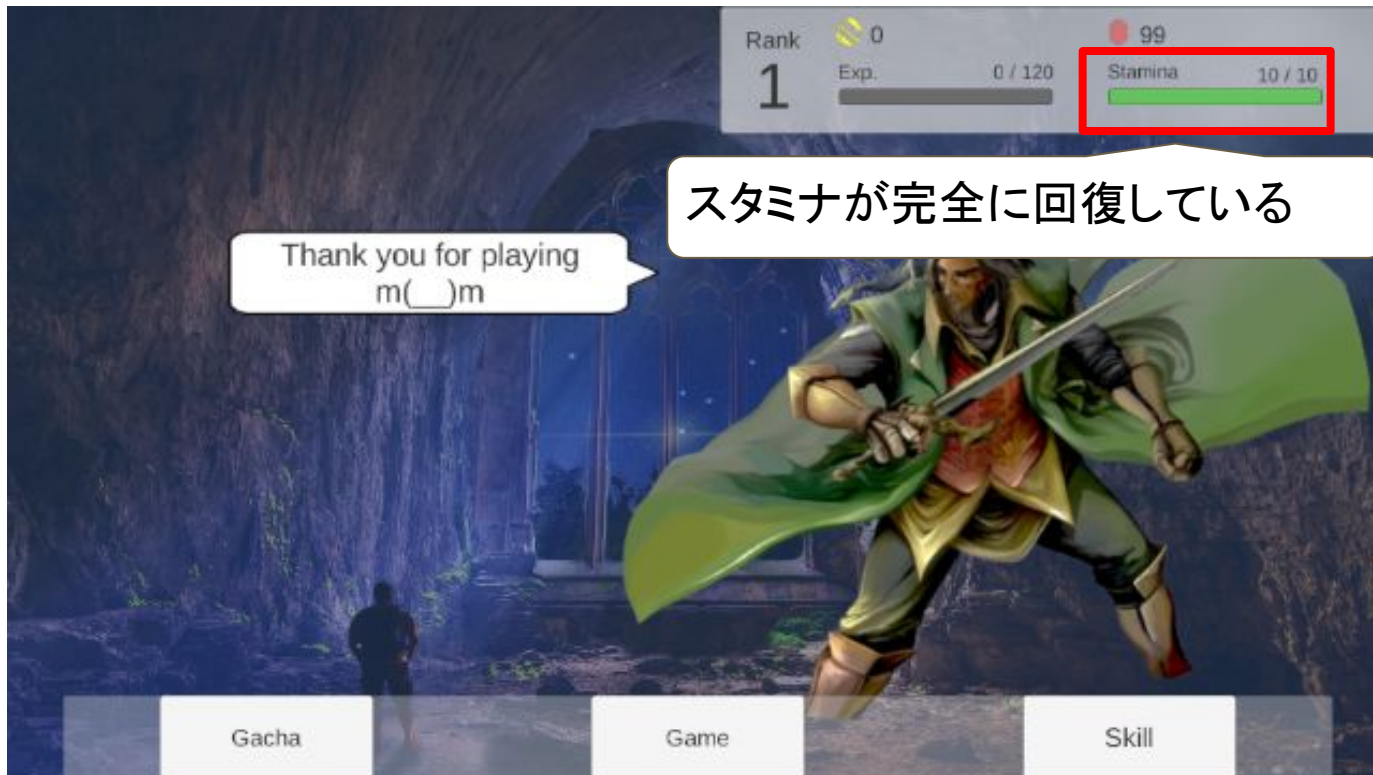
スタミナの値を **123456789** に書き換える

端末上の XML を置き換える

```
08%2F11%2F2017%2000%3A44%3A01</string>
session_elapsed_time">0</string>
```

```
$ adb push com.totem.chuni_music.v2.playerprefs.xml /data/.../shared_prefs/
```


CHUNI MUSIC - 問題点 4 の手法



CHUNI MUSIC - 問題点 4 の影響度

- 時間によって自然回復するか、
コインかダイヤ石を使用することでスタミナを回復できる
 - これらを見捨てて無制限に回復しゲームをプレイできる
 - ⇒ コインやダイヤ石への課金のインセンティブが失われる
- クライアント側でスタミナを管理しているため、問題点 3 の手法で、
現在のスタミナに関係なく無制限にスコアを登録できてしまう
- この手法でスタミナは元々の上限以上に増やすことはできない
 - そのため、スタミナを非常に大きな値に変えて
実質無制限にゲームのプレイをできるようにはならない
- **ゲームのバランスが崩れる可能性があるため、必ず修正を行いたい**

CHUNI MUSIC - 問題点 4 の対策案

- スタミナの現在値を暗号化して保存する
 - UUID は現在のバージョンでも暗号化されているが、スタミナは平文で保存されているため暗号化する
 - それでも解析を行えば改ざんができてしまうため、できるだけクライアント側にステータスを持たせない方が好ましい

CHUNI MUSIC - 問題点 4 の対策案

- スタミナの現在値の管理をサーバに行わせる
 - サーバからのアカウントのステータス取得時に、スタミナの最大値だけでなく現在値も与えるようにする
 - 楽曲のスタート時に通信を行わせて、サーバ側でスタミナを減らす処理を行うようにする
もしスタミナが足りなければ、楽曲のスタートは行わせない

CHUNI MUSIC - 問題点 4 の対策案

- 対策案の実施前



攻撃者 (本来のスタミナは 0)

67890 点のスコア登録を
不正にリクエスト



スコア登録完了!



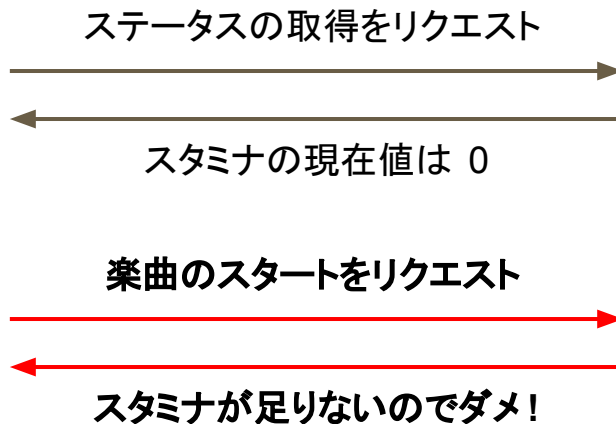
サーバ

CHUNI MUSIC - 問題点 4 の対策案

- 対策案の実施後 1



攻撃者 (本来のスタミナは 0)



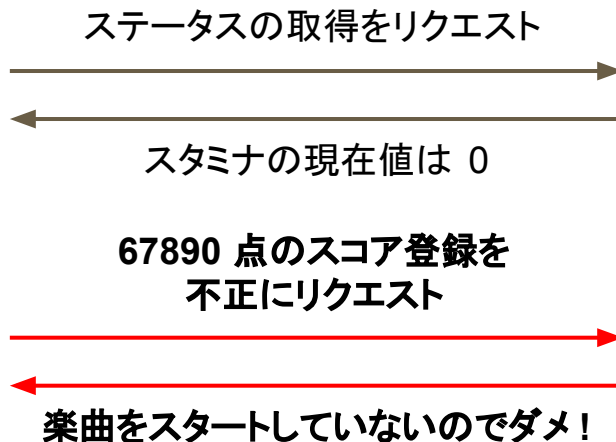
サーバ

CHUNI MUSIC - 問題点 4 の対策案

- 対策案の実施後 2



攻撃者 (本来のスタミナは 0)



サーバ

CHUNI MUSIC - その他の問題点

- root 化された端末でもゲームがプレイ可能
- 過剰なリセマラが簡単に可能
- スタミナが完全に回復されている状態でも、
コインかダイヤ石をタップすると消費されてしまう
- ユーザ名に空文字列や空白文字だけの文字列が使用可能

CHUNI MUSIC - その他の問題点

- root 化された端末でもゲームがプレイ可能
 - 影響度: メモリやファイルの書き換えなどによって、クライアント側で保持するデータが改変でき、容易にチートが可能のため、できれば修正したい
 - 対策案: ゲーム起動時などにユーザが root であるかチェックを行い、もし検知されればそこでゲームを終了する

CHUNI MUSIC - その他の問題点

- 過剰なリセマラが簡単に可能
 - 初回起動後すぐにガチャを 10 回引くことができるため、もし高レアリティのスキルが排出されなければゲームのデータを削除してもう一度ガチャを引く、いわゆるリセマラが簡単に可能
 - 影響度: サーバへの負荷の増大や、再インストールで行われた場合にはインストール数と実際のユーザ数との乖離が発生するため、できれば修正したい
 - 完全に規制すれば不公平感が生まれてしまうため、ある程度は許容したい

CHUNI MUSIC - その他の問題点

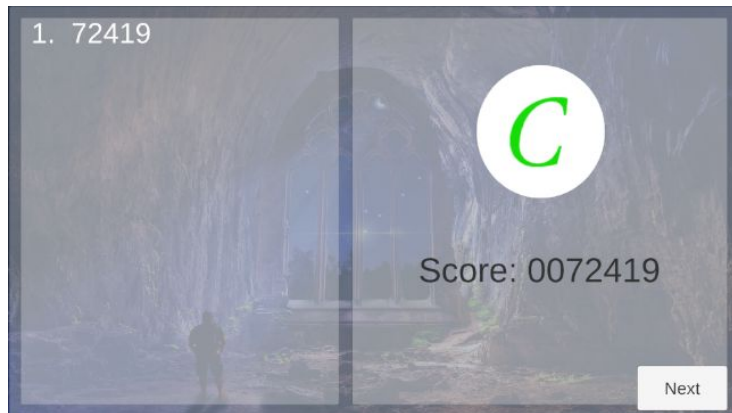
- 過剰なリセマラが簡単に可能
 - 対策案 1 (リセマラを禁止する)
 - メールアドレスや電話番号などを用いたアカウント登録機能を導入し、登録済みでないかチェックする
 - 対策案 2 (リセマラに時間がかかるようにする)
 - チュートリアルを用意して、終了までガチャが引けないようにする
 - 対策案 3 (リセマラを不要にしてしまう)
 - 初回のガチャは何度でもやり直せるようにする
 - 初回プレイ時に好きなスキルを選べるようにする

CHUNI MUSIC - その他の問題点

- スタミナが完全に回復されている状態でも、
コインかダイヤ石をタップすると消費されてしまう
 - 影響度: タップ後にワンクッション (ダイアログで確認) はあるが、
誤って消費してしまった際のショックが大きいいため、できれば修正したい
 - 対策案: タップ時にスタミナが完全に回復されていた場合には、
コインやダイヤ石は使用できないと表示して処理を中断する

CHUNI MUSIC - その他の問題点

- ユーザ名に空文字列や空白文字だけの文字列が使用可能
 - 影響度: ランキングに名前が載ると点数だけが表示されているように見え、ユーザの混乱を招く可能性があるため、できれば修正したい
 - 対策案: 空白文字以外の文字が含まれているか、ユーザ名のチェックをサーバ側とクライアント側の両方に入れる



おまけ

CHUNI MUSIC - 付録 A (静的解析)

- Unity では、C# など書かれたスクリプトが IL にコンパイルされ、Assembly-CSharp.dll や Assembly-UnityScript.dll として出力される
- 出力された dll は、ILSpy ^[1] や dnSpy ^[2] のようなツールを使うことで、容易に C# のコードとして復元することができる

[1] <http://ilspy.net/>

[2] <https://github.com/0xd4d/dnSpy>

CHUNI MUSIC - 付録 A (静的解析)

- ビルド時に IL2CPP を通すと、IL から C++ に変換され、最終的に libil2cpp.so として ARM や x86 のネイティブコードが出力される
- ネイティブコードへのコンパイルにより、パフォーマンスの向上や静的解析 (コード自体の解析) を困難にするといった効果が期待できる
- CHUNI MUSIC では IL2CPP が使用されており、lib/ 下に ARM と x86 向けにそれぞれ libil2cpp.so が存在している

CHUNI MUSIC - 付録 A (静的解析)

- libil2cpp.so にはクラス名やメソッド名のような情報が存在せず、このままでは解析は難しい
- IDA (高機能な逆アセンブラ・デバッガ) のプラグインの `nevermoe/unity_metadata_loader` ^[1] や `kenjiaiko/unity_metadata_loader` ^[2] を利用することで、`global-metadata.dat` から文字列やシンボルの情報が復元できる

[1] https://github.com/nevermoe/unity_metadata_loader

[2] https://github.com/kenjiaiko/unity_metadata_loader

CHUNI MUSIC - 付録 B (musicgame.db)

- assets/musicgame.db というデータベースが存在しているが、暗号化されており、このままでは読めない
- lib/armeabi-v7a/ 下に libsqlcipher.so が存在しているため、SQLCipher (暗号化機能が付いた SQLite) が使われていると推測できる
- データベースの暗号化に使われている鍵を探す
 - ⇒ assets/bin/Data/Managed/Metadata/global-metadata.dat に平文で存在

CHUNI MUSIC - 付録 B (musicgame.db)

- 復号してテーブルの構造を調べる

```
$ sqlcipher musicgame.db
SQLCipher version 3.15.2 2016-11-28 19:13:37
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> PRAGMA KEY = 'piyopoyo';
sqlite> .schema
CREATE TABLE skills (id integer primary key, type integer not null, param
integer, combo integer, name string);
CREATE TABLE scores (id integer, difficulty integer, score integer, primary
key(id, difficulty));
```

CHUNI MUSIC - 付録 B (musicgame.db)

- テーブルの内容を調べる

```
sqlite> SELECT * FROM skills;  
2295|1|10000|20|ScoreComboBuffS  
2296|4|1|15|ComboStaminaCureS  
sqlite> SELECT * FROM scores;  
1|0|74482
```

CHUNI MUSIC - 付録 B (musicgame.db)

- skills (所持しているスキルの一覧)

id	type	param	combo	name
2295	1	10000	20	ScoreComboBuffS
2296	4	1	15	ComboStaminaCureS

- scores (自分のハイスコアの一覧)

id	difficulty	score
1	0	74482

CHUNI MUSIC - 付録 B (musicgame.db)

- 端末上ではどこにデータベースがあるか調べる

```
$ adb shell
root@vbox86p:/ # find / -name musicgame.db
/mnt/shell/emulated/0/Android/data/com.totem.chuni_music/files/databases/musicgame.db
/data/media/0/Android/data/com.totem.chuni_music/files/databases/musicgame.db
root@vbox86p:/ #
```

CHUNI MUSIC - 付録 B (musicgame.db)

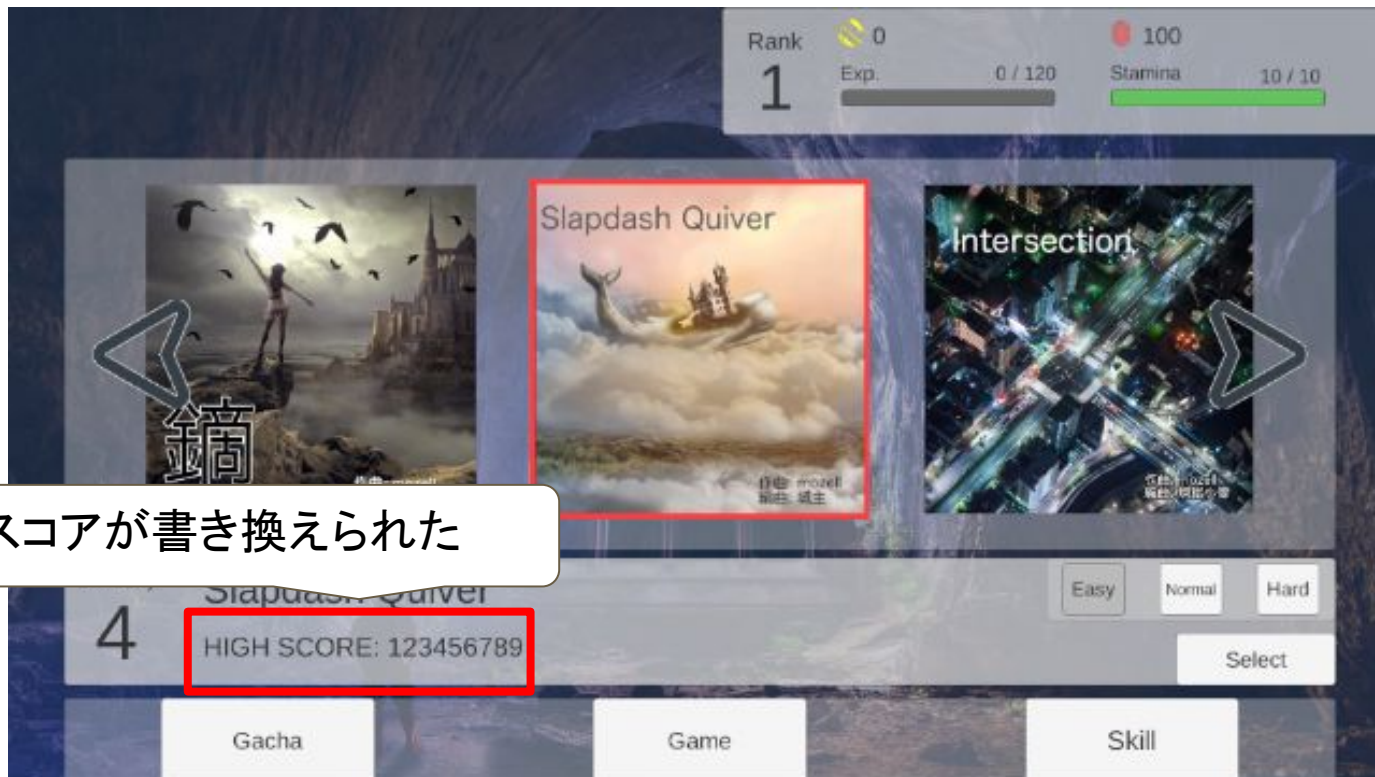
- このスコアを書き換えて保存する

```
sqlite> UPDATE scores SET score = 123456789;  
sqlite> .quit
```

- 端末上のデータベースを書き換える

```
$ adb push musicgame-modified.db /data/media/.../databases/musicgame.db  
musicgame-modified.db: 1 file pushed. 1.3 MB/s (4096 bytes in 0.003s)  
$ adb push musicgame-modified.db /mnt/shell/.../databases/musicgame.db  
musicgame-modified.db: 1 file pushed. 0.7 MB/s (4096 bytes in 0.006s)
```

CHUNI MUSIC - 付録 B (musicgame.db)



CHUNI MUSIC - 付録 B (musicgame.db)

- skills に変更を加えても何も起こらない
- scores に変更を加えてもサーバ上のランキングに影響はない
- 問題点とはいえない

CHUNI MUSIC - 付録 C (apk の改変)

- apk に変更を加えてインストールしたい
- 以下のような流れで行う
- apktool で apk を展開
 - ⇒ 好きなファイルに変更を加える
 - ⇒ apktool で apk の再構築
 - ⇒ jarsigner で apk に再署名
 - ⇒ adb install でインストール

CHUNI MUSIC - 付録 C (apk の改変)

- 実際に global-metadata.dat に含まれる文字列を書き換えて、ダイヤ石を使ったスタミナの回復時のテキストを変更してみる
- まず **apktool d app.apk** で apk を展開
 - ⇒ **META-INF/** を削除
- global-metadata.dat をバイナリエディタで編集する

CHUNI MUSIC - 付録 C (apk の改変)

- apk の再構築を行う

```
$ apktool b app-modified -o app-modified.apk
I: Using Apktool 2.2.4
I: Checking whether sources has changed...
I: Checking whether resources has changed...
I: Building apk file...
I: Copying unknown files/dir...
```

CHUNI MUSIC - 付録 C (apk の改変)

- apk の再署名を行う

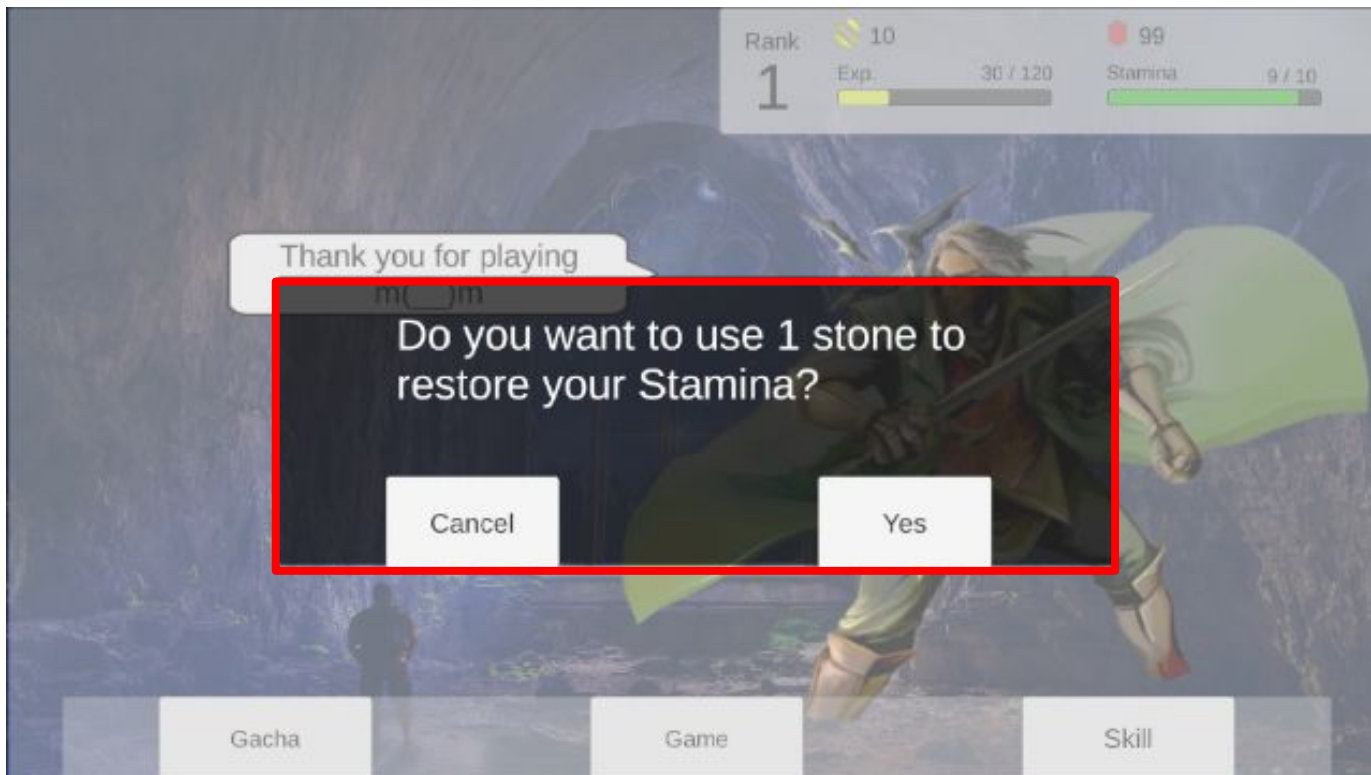
```
$ jarsigner -verbose -signedjar app-modified-signed.apk -keystore  
~/.android/debug.keystore -storepass android -keypass android  
app-modified.apk androiddebugkey  
 追加中: META-INF/MANIFEST.MF  
 追加中: META-INF/ANDROIDD.SF  
 追加中: META-INF/ANDROIDD.RSA  
署名中: AndroidManifest.xml  
...  
署名中: res/drawable-xxxhdpi-v4/app_icon.png  
署名中: resources.arsc  
jarは署名されました。
```

CHUNI MUSIC - 付録 C (apk の改変)

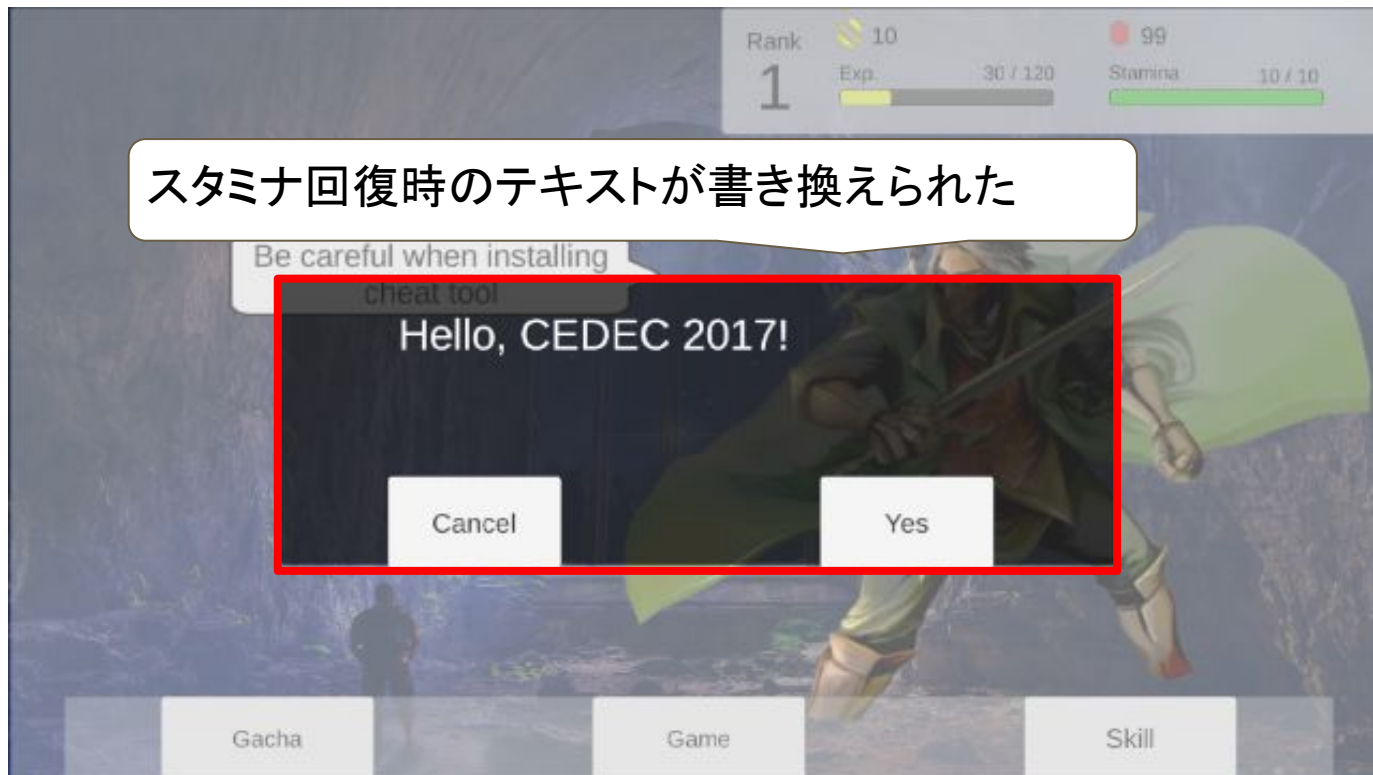
- apk の再インストールを行う

```
$ adb install -r app-modified-signed-aligned.apk
app-modified-signed-aligned.apk: 1 file pushed. 33.3 MB/s (57985924 bytes in
1.659s)
    pkg: /data/local/tmp/app-modified-signed-aligned.apk
Success
```

CHUNI MUSIC - 付録 C (apk の改変)



CHUNI MUSIC - 付録 C (apk の改変)



CHUNI MUSIC - 付録 C (apk の改変)

- libil2cpp.so に変更を加えることで、
ボタンのタップ判定やコンボ数などのチートができる?
- ARM の libil2cpp.so に変更を加えて、
コンボ数が大きく増加するようにする
- コンボ数についての処理を探すと、
MusicGameManager\$\$updateCombo というメソッドが見つかった

CHUNI MUSIC - 付録 C (apk の改変)

```
MusicGameManager$$updateCombo
```

```
...
```

```
loc_40E97C:
```

```
LDR            R0, [R0, #0x4C]
```

```
LDR            R1, [R0, #0x1C]
```

```
LDR            .
```

```
EOR
```

```
ADD            R2, R0, #1
```

```
SUB            R0, R11, #0x38
```

```
BL            ObfuscatedIntXor$$op_implicit_0
```

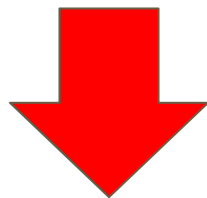
```
...
```

コンボ数に 1 加えている

CHUNI MUSIC - 付録 C (apk の改変)

- バイナリエディタで、コンボの増加数を変えてしまう

0040E980 1C 10 90 E5 20 00 90 E5 01 00 20 E0 FF 20 80 E2



0040E980 1C 10 90 E5 20 00 90 E5 01 00 20 E0 01 20 80 E2

CHUNI MUSIC - 付録 C (apk の改変)

- 再パッケージ化して実行すると...



CHUNI MUSIC - 付録 C (apk の改変)

- x86 の libil2cpp.so に変更を加えて、ミスをして HP が減らないようにする
- HP がどうやって初期化されているか調べると、**MusicGameManager\$\$InitParams** というメソッドが見つかった

CHUNI MUSIC - 付録 C (apk の改変)

```
MusicGameManager$$InitParams
```

```
...
```

```
loc_3B000D:
```

```
    call    PlayerDataManager$$CurrentRank
```

```
    lea     eax, [eax+eax+12h]
```

```
    mov
```

```
    mov
```

```
    test
```

```
    jz      loc_3B0118
```

```
...
```

現在のプレイヤーのランクを取得している
(ランクが上がると初期 HP も増える)

CHUNI MUSIC - 付録 C (apk の改変)

```
MusicGameManager$$InitParams
```

```
...
```

```
loc_3B00E2:
```

```
mov     eax, [ebp+arg_0]
```

```
mov     esi, eax
```

```
mov     eax, [esi+34h]
```

```
mov     [esp+81], eax
```

```
lea     ObfuscatedIntPlus$$op_implicit_0 を呼んで
```

```
mov     [esp], eax
```

```
call    ObfuscatedIntPlus$$op_implicit_0
```

```
sub     esp, 4
```

```
...
```

```
call    MusicGameManager$$updateHitPointUI
```

```
...
```

CHUNI MUSIC - 付録 C (apk の改変)

```
MusicGameManager$$InitParams
```

```
...
```

```
loc_3B00E2:
```

```
mov     eax, [ebp+arg_0]
```

```
mov     esi, eax
```

```
mov     eax, [esi+34h]
```

```
mov     [esp+8], eax
```

```
lea     eax, [esp+18h]
```

```
mov     [esp], eax
```

```
call    ObfuscatedIntPlus$$Implicit_0
```

```
sub     esp, 8
```

UI の HP 表示を更新している

```
...
```

```
call    MusicGameManager$$updateHitPointUI
```

```
...
```


CHUNI MUSIC - 付録 C (apk の改変)

- HP が参照されている部分を探すために
ObfuscatedIntPlus\$\$op_implicit_0 を呼んでいる箇所を調べる
- **MusicGameManager\$\$InitParams、**
MusicGameManager\$\$checkBadGrade、
MusicGameManager\$\$updateCombo から呼ばれていた
- 2 つ目のメソッドでミスをした際に HP を減らす処理をしている?

CHUNI MUSIC - 付録 C (apk の改変)

```
MusicGameManager$$checkBadGrade
```

```
...
```

```
mov     [esp], eax
```

```
mov     dword ptr [esp+8], 0
```

```
mov     dword ptr [esp+4], 0
```

```
call    GameObject$$SetActive
```

```
mov     eax, [eax]
```

```
mov     ecx, [ecx]
```

```
lea     eax, [eax+ecx-1]
```

```
mov     [esp+8], eax
```

```
lea     eax, [esp+20h]
```

```
mov     [esp], eax
```

```
call    ObfuscatedIntPlus$$op_implicit_0
```

```
...
```

HP から 1 を引いて ...

CHUNI MUSIC - 付録 C (apk の改変)

MusicGameManager\$\$checkBadGrade

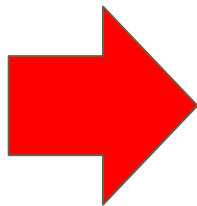
```
...  
mov     [esp], eax  
mov     dword ptr [esp+8], 0  
mov     dword ptr [esp+4], 0  
call    GameObject$$SetActive  
mov     eax, [edi+38h]  
mov     ecx, [edi+3Ch]  
lea     eax, [eax+ecx-1]  
mov     [esp+8], eax  
lea     eax, [ObfuscatedIntPlus$$$op_implicit_0] ObfuscatedIntPlus$$$op_implicit_0 を呼んでいる  
mov     [esp], eax  
call    ObfuscatedIntPlus$$$op_implicit_0  
...
```

CHUNI MUSIC - 付録 C (apk の改変)

- バイナリエディタで、HP から引かれる値を変えてしまう

003AFA10 8D 44 08 FF

`lea eax, [eax+ecx-1]`

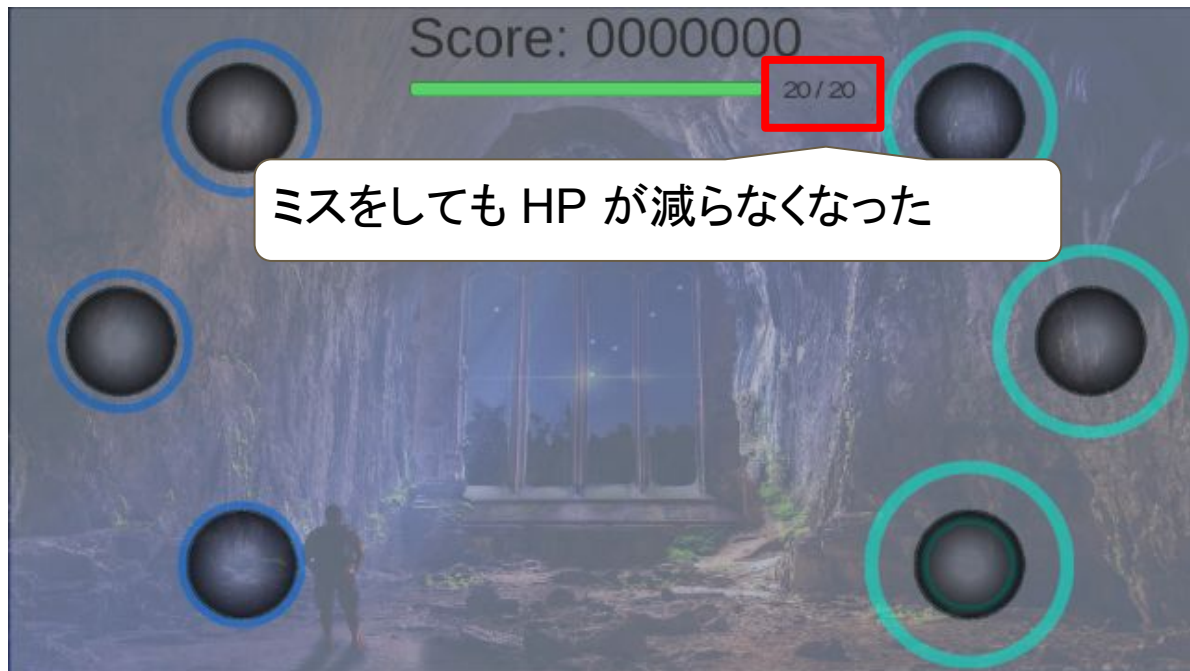


003AFA10 8D 44 08 00

`lea eax, [eax+ecx]`

CHUNI MUSIC - 付録 C (apk の改変)

- 再パッケージ化して実行すると...



ミスをしても HP が減らなくなった

CHUNI MUSIC - 付録 C (apk の改変)

- 再パッケージ化して実行すると...



<https://youtu.be/zxfDEeKmNPI>

ありがとうございました。