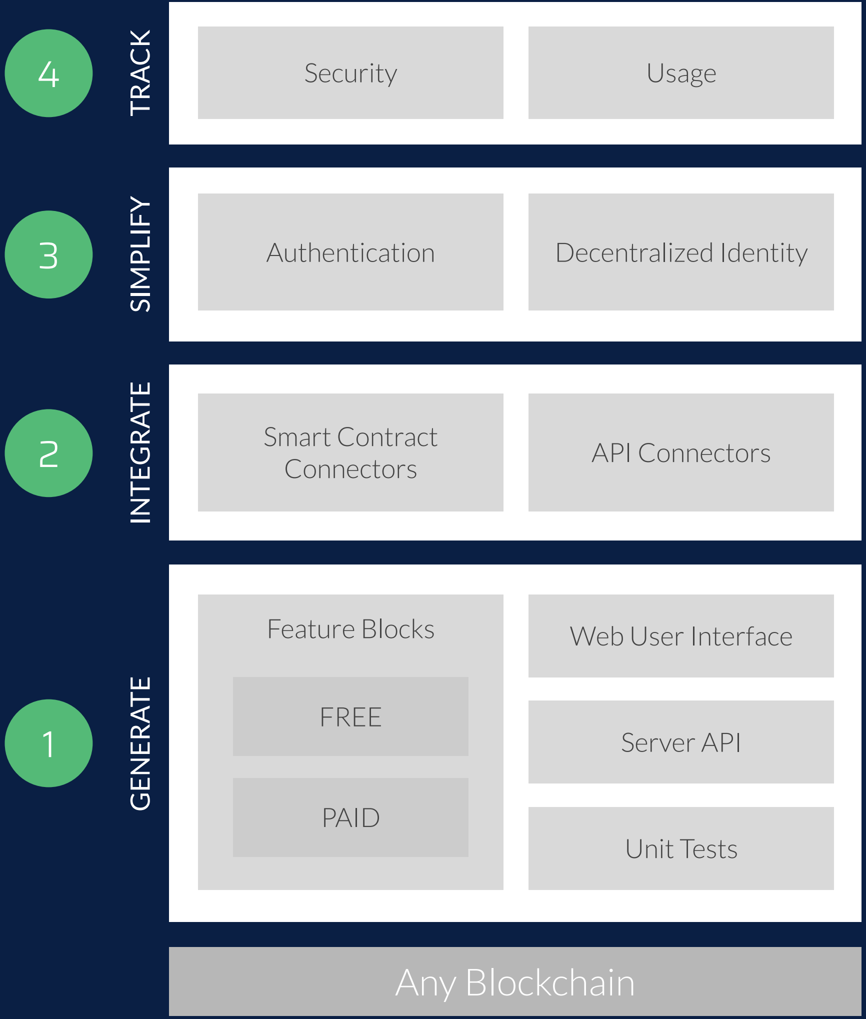
TRYCRYPTO

GIST
Stack



4 TRACK

Security

Usage

3 SIMPLIFY

Authentication

Decentralized Identity

2 INTEGRATE

Smart Contract Connectors

API Connectors

1 GENERATE

Feature Blocks

FREE

PAID

Web User Interface

Server API

Unit Tests

Any Blockchain

DappGrader **(Q4 2020)**
Blockchain app auditing and analytics service.

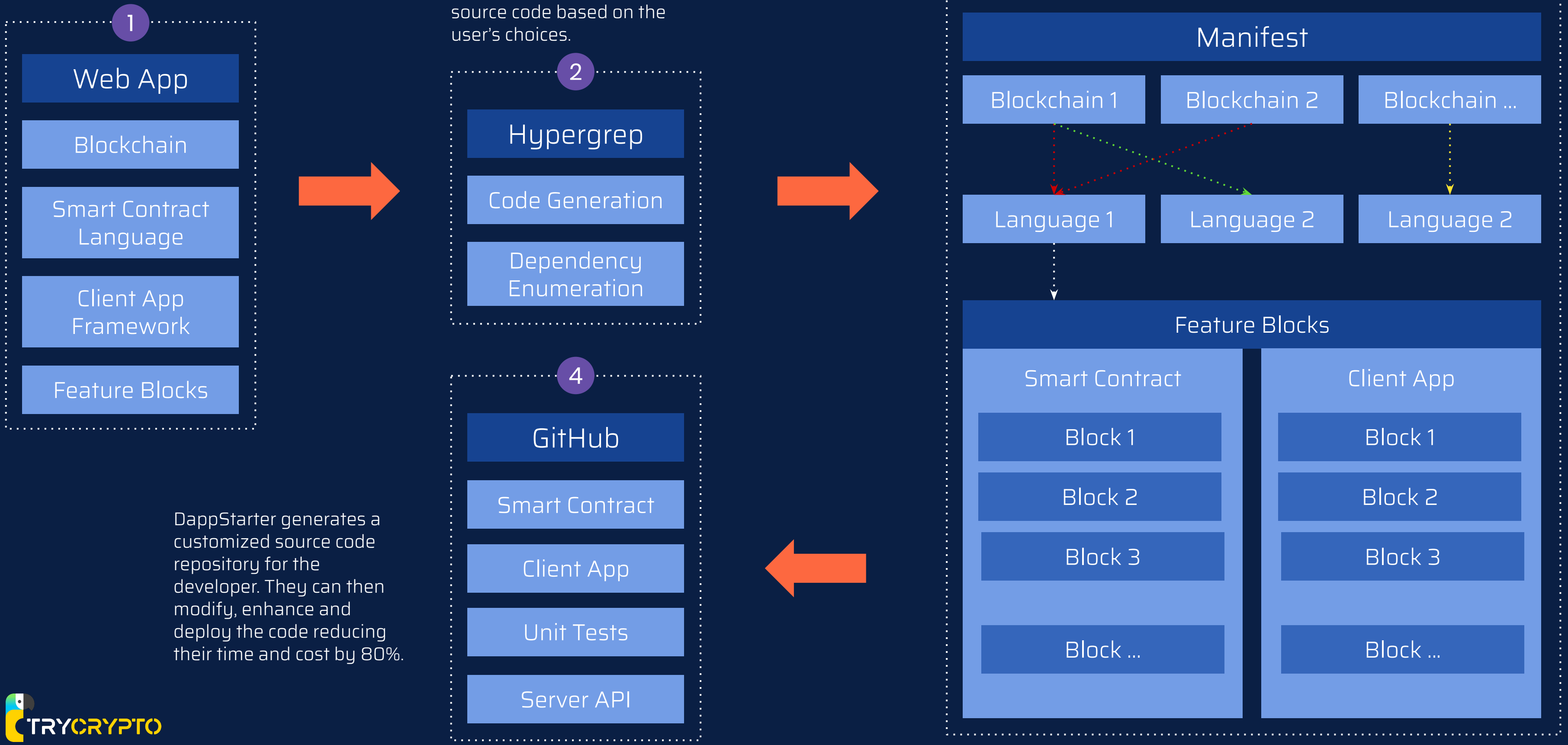
PhotoKey **(alpha launched)**
Open Source, blockchain-agnostic, authentication and decentralized identity application.

DappConnector **(in development)**
Blockchain oracle service for easy integration of existing data using a visual connection builder.

DappStarter **(beta launched)**
Full-stack, blockchain app generator that lets developers assemble their app using feature blocks

DappStarter Platform Architecture

Developer chooses their desired configuration from our SaaS website app.



1

- Web App
- Blockchain
- Smart Contract Language
- Client App Framework
- Feature Blocks

Our proprietary code generation engine uses an intelligent manifest system to produce full-stack project source code based on the user's choices.

2

- Hypergrep
- Code Generation
- Dependency Enumeration

Code generation uses a flexible, intelligent, manifest-based architecture that enables code re-use and complex dependencies.

3

Manifest

- Blockchain 1
- Blockchain 2
- Blockchain ...

- Language 1
- Language 2
- Language 2

Feature Blocks

Smart Contract

- Block 1
- Block 2
- Block 3
- Block ...

Client App

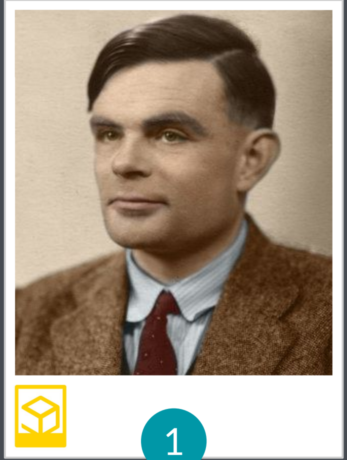
- Block 1
- Block 2
- Block 3
- Block ...

4

- GitHub
- Smart Contract
- Client App
- Unit Tests
- Server API

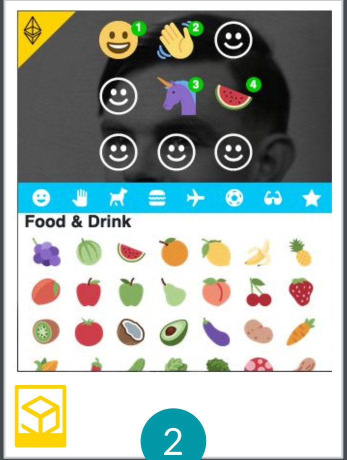
DappStarter generates a customized source code repository for the developer. They can then modify, enhance and deploy the code reducing their time and cost by 80%.

TRYCRYPTO



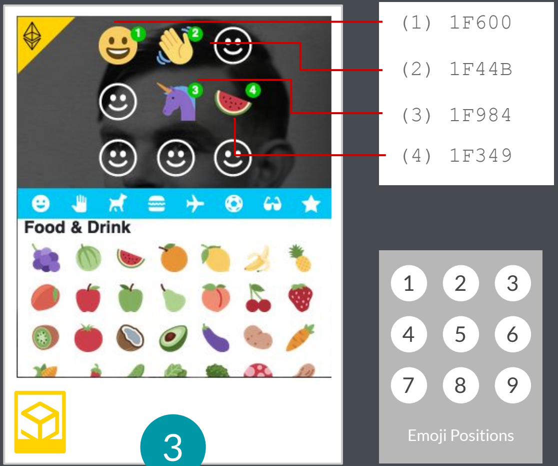
1

User uploads their photo. PhotoKey down-samples it to a smaller size and creates a PhotoKey file.



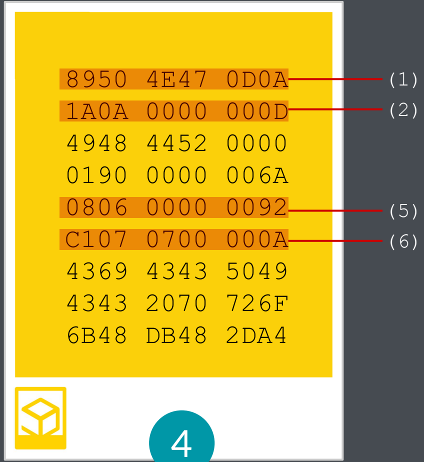
2

The user creates an EmojiKey by choosing emojis at each of nine positions in sequence.



3

PhotoKey gets the byte value for each emoji in sequence, ignoring empty positions.



```
8950 4E47 0D0A
1A0A 0000 000D
4948 4452 0000
0190 0000 006A
0806 0000 0092
C107 0700 000A
4369 4343 5049
4343 2070 726F
6B48 DB48 2DA4
```

4

PhotoKey divides photo bytes into nine segments and extracts bytes for each emoji position.

H1 = blake2s("1F600", "89504E470D0A")

H2 = blake2s("1F44B", "1A0A0000000D")

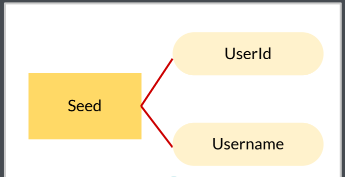
H3 = blake2s("1F984", "080600000092")

H4 = blake2s("1F349", "C1070700000A")

Seed = blake2s(H1, H2, H3, H4)

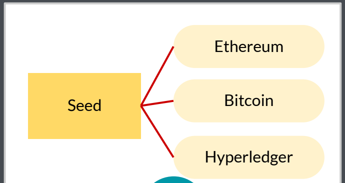
5

PhotoKey uses the Blake2s algorithm to hash emoji bytes with photo bytes to derive a high-entropy seed for keygen. Any change in photo or emoji bytes yields a different seed.



6

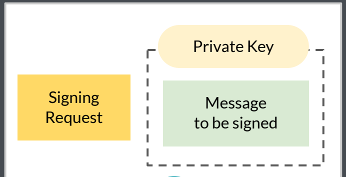
The seed is used to deterministically generate a UsedId and Username. Their hash is stored in the XMP (eXtensible Metadata Platform) section of the PhotoKey file.



7

When signing in, steps 2-6 are repeated except the hash in Step 6 is compared to the stored value in the photo. If they match, the seed is used to derive a public key and account for the blockchain in use.

No private key or any other security information is ever stored in PhotoKey!



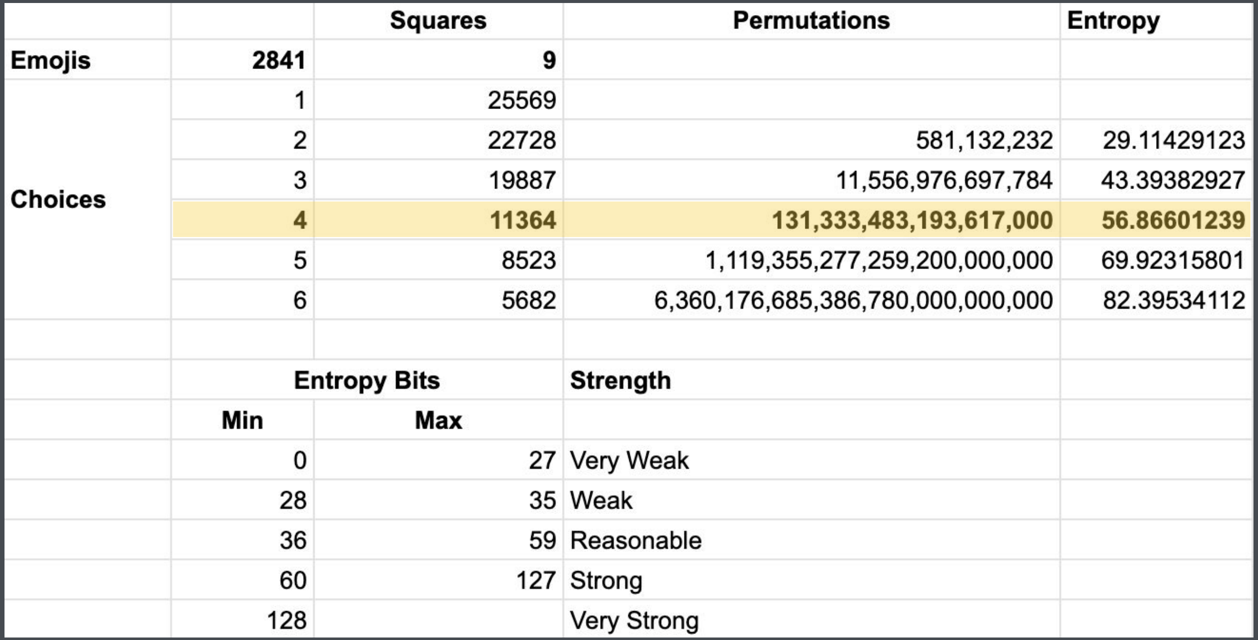
8

The public key and account address are reported to the calling application. The private key is only generated for signing requests and not available to the application.

TRYCRYPTO

PhotoKey Security

131 quadrillion permutations with 4 emojis (the minimum required).



		Squares	Permutations	Entropy
Emojis	2841	9		
Choices	1	25569		
	2	22728	581,132,232	29.11429123
	3	19887	11,556,976,697,784	43.39382927
	4	**11364**	**131,333,483,193,617,000**	**56.86601239**
	5	8523	1,119,355,277,259,200,000,000	69.92315801
	6	5682	6,360,176,685,386,780,000,000,000	82.39534112
	Entropy Bits		**Strength**	
	Min	**Max**		
	0	27	Very Weak	
	28	35	Weak	
	36	59	Reasonable	
	60	127	Strong	
	128		Very Strong	

No server - 100% browser-based

No private key storage

PDF recovery kit at creation

No steganography

Ref: https://math.stackexchange.com/questions/2961461/combinatorics-with-ordering-significance