One of the many concerns in the developing world is corruption. Although many developing nations have attempted to be rid of it; 38 ratifying the Anti-Bribery Convention since 199, it is still on the rise (Olken, B.A. and Pande, R., 2012).

Blockchain can be used to prevent corruption through its layers of security. When income is distributed by the state, Blockchain can be used to ensure it is assigned to the correct individuals. This would mean, states would only have to ensure the distributors of this wealth are corruption free. Instead of having anti-corruption spread throughout all government bodies and sub-structures. This has been is prone to failure (Doig, A. and Riley, S., 1998).

**Design philosophy**

When wealth is distributed (block creation), Blockchain is used to validate the receiver’s address. Take 2 individuals who are meant to be allocated blocks (welfare) by the state. Figure 1 demonstrates how the receivers address is preventing the tamperer’ from accessing the block. This would ensure the wealth is accurately allocated to the beneficiary and not just a fraction. It would also enable any cyber-security division to intervene and locate the receivers address of the tamperer and subsequently freeze the account and/or follow criminal proceedings.

**Figure 1.1**

Expected recipients hash addresses: {“F1GHLJ”, “M23HLJ”}



Individual 2 hash address = M23HLJ

Individual 1 hash address = F1GHLJ

Matches address, thus added

Matches address, thus added



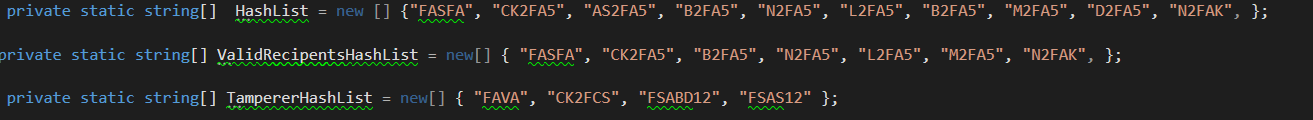
Address not valid thus not added



Tampere hash address = Z2FHFJ

**Coding plan**

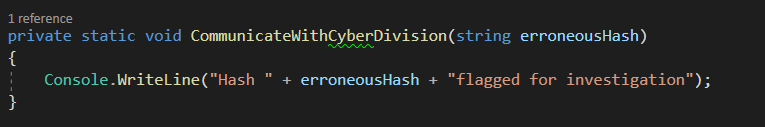
Three separate Hash lists are being used in the example scenario:



ExampleValidOrInvalidHashAddresses**: List of hash strings, containing both valid and potential invalid hash address. This is the base in which the program iterates. It represents a real scenario, as tampereres may join the** ValidRecipentsHashList **before the blocks are distributed.**

ValidRecipentsHashList**: Expected list of hash addresses in which the new blocks are allocated.**

TampererHashList**: Four hash addresses; example tamperers/saboteurs which may or may not attempt to join the existing hash list.**



**Local variables are also declared in the main loop:**

tempList**: Local variable; takes a copy of the hash list if an erroneous hash has been found. In C# values cannot be deleted from objects whilst iterating.**

**Two main classes exist, based on object orientation principles.**

1. **Blockchain: Base routine representing the process of authenticating the distribution of a new block (income is distributed)**
2. **SecurityDivision:** **Example cyber security state structure.**

**The base routine iterates through the ‘**ExampleValidOrInvalidHashAddresses’**. This routine (**CheckForTamperers**) demonstrates what may occur when blocks/wealth are distributed in a real-life scenario. The routine verifies the expected recipients of a new block (i.e. wealth distributed to a number of individuals).**

1. **Whether the previous element in the hash list matches the previous element in the ‘**previousList **variable. This is to check for any tamperers that may have joined the existing list. If the pointer does not match to the correct element, then an imposter has been found and is removed.**
2. **If the valid hash string is in the valid recipients hash list. This checks for any tamperers that may have joined the ‘ValidRecipentsLHashist’ before the blocks are distributed.**

**If either of these fail the variable within the iteration is removed and the routine is recursively executed until the iteration succeeds without either of these alternative flows occurring.**

**If the routine proceeds to succeed without any issue, the blocks are valid and can be distributed to each user.**

**References**

Olken, B.A. and Pande, R., 2012. Corruption in developing countries. *Annu. Rev. Econ.*, *4*(1), pp.479-509.

Doig, A. and Riley, S., 1998. Corruption and anti-corruption strategies: Issues and case studies from developing countries. *Corruption and integrity improvement initiatives in developing countries*, *45*, p.62.