# Decentralised and Autonomous Reward Ecosystem Whitepaper

*-> System Properties:*

- *Metasploit exploit and payload format*
- *Synchronise with Exploit-db + cxsecurity*
- *At least RFI, BFO and other bugs that might launch the code*
- *Sandbox windows/mac/linux nodes,*
  - *always online, ready to test code*
  - *newest version, AV + CVE list watch*
- *New exploit can be sent to the system, then system will be able to launch code, using appropriate sandboxed virtual machine, and then after necessary checks, %Tokens will be sent out as reward*

*-> Blackhats, have few choices:*

1. *Choice to publish a bug for free*
2. *Choice to get paid tiny amount of money for even most critical bugs by large companies and get themselves under the radar.*
3. *Choice to sell it on blackmarket for up to couple of hundreds of thousands dollars absolutely anonymously*
4. *Choice to use it themselves for even larger gain (potentially millions of dollars of damage from each zero day exploit) and less likely to get under the radar*

- *Most of the hackings are fuelled by greed or lack of money...*
- *DAER gives a much better option and solution for cyberwar!*

- *Traditional anti-malware and anti-virus, security solutions, do not have a solution for most of the unknown exploits, cannot recognise more than 40% of threats and most certainly, cannot avert zero-day exploits until its too late!*

- *With NEO based, blockchain reward system "DAER"*
- *One can now fully anonymously, and fully autonomously, post exploit!*
- *One can actually help people and secure the internet even more!*
- *and get paid a fair bit amount of tokens, which might always grow in value, and the amount could be determined to be optimal, based on community decision of importance and significance of exploit.*
- *Thanks to incredible power of blockchains,*
- *we can actually start to get more and more hackers on the good side,*
- *keep-up fully autonomous database of exploits as engine for anti-viruses*
- *to use in future.*

# *API Documentation, developer instructions*

1) Firstly contract gets initialised and admin adds greyhats, sandbox nodes and curators

`bool registerAuthority(byte[] userAddress, string authType)`

this method can only be executed by admin, it takes neo address in bytes array and authentication level as string
it outputs true if check verified and action succeded.

2) Then, Greyhat applies working payload (0-day etc.) to smart-contrat

`bool ApplySploit(string cve, byte[] content)`

this method takes cve code as string and array of bytes of content,
which can either be a payload part or full metasploit-style script

3) Thirdly, Sandbox node PoC tests metasploit or other payload against given target

`cveObject CutLastPending()`

looks for last object submitted by greyhat, returns it and removes it from the array in the contract
then the payload is being fully tested using separate open-source software.
It will emulate windows or macos using virtual machine, and run test against appropriate target software
if at the end of the test, permissions has been elevated, the system will then call next function from contract
using following method with the same variables, and again will return true or false.

`bool SendToCurator(string cve, byte[] content)`

4) Once Curator receives the payload, he will manually check/confirm the exploit

report it to global Antivirus databases,
and release reward for the Greyhat from escrow
or deny if sandbox passed a false positive

`static bool ApproveSploit(string cve, byte[] content, byte[] owner)`

`static bool DenySploit(string cve)`