B) Algo: Perform a k-ary search where k is the number of hackers

if $k=1$ launch fake sites to all users

$k==2$ launch a fake site to $n/2$ users and if the hackers are not detected in that half launch to the other half

any k launch to $\frac{n}{k}$ users and do a similar process to when $k==2$

when $k=n/2$ we go back down in k-searches

Runtime: we launch to n/k users each time, cutting our input by $1/k$ each time $O(\log_k n)$

~~we iterate over each subset~~

~~k-ary search has a runtime of~~

~~$O(k \log_k n)$~~

we have to launch k sites each time

$O(k)$

Since the launching process is nested in the dividing process:

$$O(k) * O(\log_k n) = O(k \log_k n)$$

Proof: In a k-ary search, we have k-subsets, so like in a binary search, the hackers are bound to be in one of them. If we detect activity in one of the subsets then we know that a hacker is in that subset. So we then split that subset and squeeze out the hacker. By going backdown (ie when $k > n/2$ we start to perform the same amount of searches for $k+2$ as $k-2$ and $k+1$ as $k-1$) we prevent from using an excessive amount of [redacted] Fake websites