Breaking open the Bazaar: identifying and exploiting key weaknesses in the OpenBazaar network

Richard Dennis
University of Portsmouth
Buckingham Building
Portsmouth
023 9284 6423
richard.dennis@port.ac.uk

Gareth Owen
University of Portsmouth
Buckingham Building
Portsmouth
023 9284 6423
gareth.owen@port.ac.uk

ABSTRACT

This paper provides the first analysis of the OpenBazaar network; it identifies vulnerabilities within the current network by developing and implementing multiple innovative new attacks to further exploit OpenBazaar's unique vulnerabilities. We conclude by suggesting and testing countermeasures to these attacks on a live network which not only make the network secure against a low-medium resourced adversary but also enhance the speed and storage capacity of the existing network.

Categories and Subject Descriptors

C.2.1 [**COMPUTER-COMMUNICATION NETWORKS**]: *Distributed networks, Network communications, Network topology.*

General Terms

Performance, Design, Reliability, Experimentation, Security.

Keywords

OpenBazaar, peer-to-peer, distributed networks, Kademlia, privacy enhancing technologies, cryptographic protocols

1. INTRODUCTION

In a post-Snowden world where trust in centralized services is low and anti-censorship sentiment is at an all-time high, an e-commerce network that is free, open sourced, decentralized and resistant to censorship is a highly desirable prospect; hence the launch of OpenBazaar in April 2014 [11]. OpenBazaar is a decentralized distributed network that launched in April 2014. It is a peer-to-peer online commerce network which uses Bitcoin as its method of payment. Unlike current e-commerce networks, it has no central point of failure and cannot be controlled by a single entity. The internet has provided decentralized communication, Bitcoin has provided decentralized currency, but currently there is no decentralized trade.

OpenBazaar is free software which connects users to marketplaces hosted on a free volunteer network and, despite being relatively new, it has quickly gained a large following. The rapidly growing network currently has 726 active developers and participants in its developer chat room, and in the past 12 months 2,827 markets have been established and begun actively trading [1][8].

OpenBazaar uses Kademlia distributed hash table (DHT) implementation for peer discovery and queries, and communication takes place directly between the client and marketplace instead of being forced to go through a centralized server.

The main advantages of OpenBazaar over more traditional centralized marketplaces such as eBay or even Silk Road is that there is no censorship, due to the decentralized design. OpenBazaar enables goods to be purchased using Bitcoin, and offers protection for both markets and customers by being the first network to implement Ricardian-style contracts.

The revolutionary introduction of notaries also sets OpenBazaar apart from previous e-commerce marketplaces. A notary is a human impartial third party who can look at evidence, such as tracking details, if issues arise in a transaction they are notarizing and who can then decide which party should get the money.

2. RELATED WORK

This section examines the technologies used in OpenBazaar and compares OpenBazaar with current alternatives. It also explains its network topology and protocols as well as examining previous attacks that have been conducted against the OpenBazaar network architecture.

2.1 Tor Hidden Services

Tor is an open-source project that is a decentralized low latency mix network of specially configured nodes, commonly called relays or bridges, which transmit only TCP traffic through virtual tunnels from a client to a destination.

Services such as websites can also be hosted within the Tor network. These are known as "Hidden Services" and can only be accessed through Tor. Tor Hidden Services were added in 2004, when the second generation of onion routing was developed [3]. In recent months, there have been several high profile "exit scams" conducted by two of the largest marketplaces on Tor, Agora and Evolution [11].

An exit scam is when a marketplace that has traded legitimately for months suddenly ceases to do so and, while still taking orders and money from customers, starts refusing to send out items. This can be achieved due to the use of an escrow account system, where the customer pays the marketplace and the marketplace holds the funds until the buyer has received the item. The escrow account system relies on the marketplace being trustworthy as, unlike OpenBazaar, there are no security implementations to prevent the marketplace from withdrawing the cash.

In OpenBazaar the nodes are currently arranged in a structured Kademlia topology and the network uses the Kademlia distributed hash table (DHT) to allow these nodes to find each other. Once a node has received the IP address and port of the node they wish to communicate with, they can set up a direct channel for communication using UDP, without having to send all communications through the P2P network.

2.2 Kademlia

Kademlia is a P2P topology that was first described by Maymounkov and Mazières in 2002 [9]. It has had widespread success, and is currently used in multiple P2P networks, arguably the most well-known of which is its use in BitTorrent. Kademlia is a distributed hashtable overlay network that assigns each node a global unique 160-bit identifier called the GUID, although the process that assigns the GUID is down to individual implementations [7]. This means that in some implementations attackers are able to generate GUIDs, and as such it can be considered a weakness [2].

Jenkov [6] compares Kademlia and Chord and identifies one of the most important features of Kademlia as being the use of XOR to calculate the distance between two points. XOR was implemented instead of standard subtraction methods, as seen in other DHT's such as Chord, because XOR is symmetrical. Consequently in Kademlia when a node decides to leave the network, it just has announce this to all the nodes in its routing table, as all the nodes that have the leaving node in their routing table also feature in the leaving node's routing table [9]. This makes leaving much easier, and more effective, than in Chord, as in Chord a node has to find all nodes with the leaving node in their routing table [10], thus taking significantly longer to process. In addition, it is also possible to miss out and fail to notify some nodes. XOR is also easier to calculate and when XOR is implemented the result can fit into a single data type instance, which is not always possible with other methods, such as the one used in Chord. Kademlia also has a flexible routing algorithm which can select routes based on latency or, as in OpenBazaar, send three parallel asynchronous queries.

Keong Lua et al. further expand on this, explaining how each peer in the network stores triples containing the IP address, UDP port and GUID of other peers; these are stored in K-Buckets, which usually contain 20 items [7]. Maymounkov and Mazières warn us that there is no guarantee that peers can be found, and explain how Kademlia only locates nodes which are closer than the current node to the target GUID [9].

As nodes are encountered on the network, i.e. when searching for a key for another peer, they are added to the K-buckets. These K-buckets are ordered with the most recently accessed peer at the tail [9]. K-Buckets provide a defense against some DOS attacks on the network by not flooding the K-Buckets with new - and potentially malicious - nodes; instead a node is only added to the K-Bucket once another has left the network.

Jenkov feels that the advantages of Kademlia over other P2P topologies is that XOR is simple to calculate and the routing tables (K-Buckets) make routing easier to manage and, arguably, more efficient [6].

Cholez et al. [2] describe how the organization of the routing tables enables Kademlia to route a query in O(log(n)), thus confirming Jenkov's statement about the organization of the routing tables making the network more efficient. However, they dispute the statement made by Maymounkov and Mazières that the K-Buckets prevent a Sybil attack, and demonstrate how a malicious node could rewrite a legal K-Bucket entry.

For a node to join the network, it must first know of a node on the network; this can be a node pre-coded into the implementation or found out-of-band. The node generates a GUID. As it is the client and not the network that does this, it is possible that the node could generate a GUID identical to that of another node. Wang et al. [12] acknowledge this as a potential problem and propose a solution of linking the IP address and port of the node to its GUID to prevent a GUID clash; whilst this is a good idea, it has not been implemented into the Kademlia protocol.

Once the node has its GUID and knows another node in the network, it can start to find other nodes. To do this, it queries its GUID to the other node it knows; this returns the three possible closest nodes which the new node will add into its K-Bucket; this process is repeated until the nodes' K-Buckets are full.

3. Methodology

Experiments using multiple attacks were conducted in order to determine the security of the OpenBazaar network.

In OpenBazaar, anyone can run a node by simply downloading the source code and running the application. The following newly discovered attacks can be carried out by an adversary with very limited resources, requiring only access to a computer and a stable internet connection.

4. New attacks on OpenBazaar

4.1 Double-agent attack

One of the major innovations of OpenBazaar is that it is the first to have successfully implemented a multi-signature contract on a P2P network. By requiring a majority of 2 out of the 3 signatures, the multi-signature Ricardian-style contracts ensure no single entity is able to withdraw the Bitcoins. This aims to prevent a situation where a client buys goods from a market and the market does not deliver the goods or service that was paid for.

Currently, the client selects the notary. Once the client and marketplace agree on the terms of a contract and the notary agrees to fulfill the role, a multi-signature Bitcoin wallet is created using the public keys from all three participants. This wallet is where the customer sends the payment to be held until two of the three participants use their private keys to release the funds.

OpenBazaar assumes that this system is secure as both parties can see the selected notary, however this system only provides security if none of the actors in the transaction are able to collude with each other. In the double agent attack, it is possible for the notary to collude with a client to allow the client to receive both the item and the Bitcoins that were used to pay for the item.

The attack requires an attacker to create two nodes, one to act as a buyer and one to be a notary. As this can be conducted from the same machine, the potential cost of this attack is quite low. Nodes do not require a certain uptime before becoming a notary in OpenBazaar, so this attack can be set up and conducted within minutes. To become a notary in OpenBazaar, a node announces to the network that it offers the notary service.

To conduct the attack, an attacker selects a target marketplace which sells an item they would like. The client selects the notary it controls and completes the transaction normally; the marketplace thinks this is a genuine transaction and ships the item to the customer. Once the client receives the items, the malicious client and notary can then use their two private keys for the multi-signature wallet to return the funds to the client. The marketplace is unable to stop this as it only controls one of the three keys, and therefore lacks the majority required to transfer funds. In this scenario, the attacker now has both the item and the Bitcoins.

This attack has been shown to have a 100% success rate, because the client has the ability to select the notary and the marketplace does not realize it is under attack until the client and notary collude and withdraw the funds, by which point it is unable to do anything about it.

After this attack, a marketplace could block the attacker by refusing to trade with the client and notary GUID, however as GUIDs are not linked to data such as IP addresses, users can change the GUID in the configuration file, which effectively renders this defense useless.

This attack could also feature a malicious marketplace and a malicious notary. This is a slightly cheaper attack to conduct as the attacker does not need the Bitcoins to buy the items first. However, this attack has a higher risk of failure as the marketplace would have to refuse any contract that does not contain a notary they control.

There are currently no restrictions on how many nodes an IP address can possess. There is also no central authority, unlike the Tor directory servers, which can monitor and approve nodes on the network. These features of the OpenBazaar network mean it is possible to Sybil attack the network. We conducted a Sybil attack on the live network and on a simulated version of the network and found it was extremely cheap, fast and effective to add thousands of malicious nodes to the network from a single (or cluster) machines with the only limitation being the amount of RAM each machine had (1 instance of OB required 0.1GB of RAM)

This variation of the attack could be combined with one of the aforementioned Sybil attacks to greatly increase the chance of a user selecting a notary controlled by the marketplace. This attack would, however, be harder for the market to conduct long-term, as word would soon get around that this marketplace was untrustworthy, and clients would stop using them.

This attack is very similar to the "exit scam" conducted by Evolution and Agora on Tor. These marketplaces traded legitimately for a long time before starting to take payments for goods and not sending them (although these marketplaces used the traditional send and receive payment and not multi-signature addresses). OpenBazaar have claimed that this attack can no longer be conducted due to the protection of the triple signed wallet, as well as both parties having the ability to see the notary node, however we have proven this not to be the case.

This attack only has to control two nodes - the client and notary - even on a 10,000,000 node network. This prediction was tested on a simulator so as not to impact the real network and held true, with a 100% success rate. This proves that even on a network containing millions of nodes this attack is possible for an adversary with highly limited resources.

This attack works on the assumption that the marketplace accepts the chosen client's notary; in the tests on the live network this assumption held true, but as the network grows and nodes develop a reputation, a marketplace may choose to refuse to deal with notary nodes with a less than 'X' reputation.

The Double Agent attack has proven to be an effective attack which only requires control of two nodes and which, when simulated multiple times on a variety of network sizes (all of which used identical communication to the real network), proved to be successful 100% of the time. As only two nodes need to be controlled in order for the attack to succeed, the success of the attack is not dependent on the size of the network.

4.2 Impersonation attack

In the implementation of OpenBazaar at the time of writing (Beta v4), the GUID of an OpenBazaar node is randomly created on installation, however this can be changed by manually editing the database. This attack demonstrates how it is possible to give two marketplaces, notaries or clients the same GUID. By being able to change the GUID of a node, it is possible to target and impersonate a specified node, however there are some issues here. While an attacker is able to impersonate the GUID, they can also impersonate the marketplace; they would simply need to copy the items they are selling and prices. Since each node hosts its own products and there is no data redundancy, once an attacker impersonates a marketplace's GUID, it can impersonate the whole market. However, an attacker cannot replicate the PGP address, as the attacker doesn't know the private key of the marketplaces PGP.

The ability to replicate a marketplace is not a useful attack on its own; OpenBazaar protects users from malicious marketplaces by using triple signature contracts. If, however, this attack were used in conjunction with the double agent attack or the Sybil attack, it would enable an attacker to use the reputation of a genuine marketplace and be able to conduct an attack to gain Bitcoins.

We created a 20,000 node OpenBazzar network using a simulator; this replicates all functionality of the network, apart from network latency. The impersonation attack was tested on a range of GUIDs belonging to clients, notaries and marketplaces and had an 80% success rate. There are several reasons why 20% of the tests failed; including the message failing to be delivered to the rest of the network as a result of the UDP protocol and the target node restarting while the attack was underway. This meant that although the attack node impersonated the target for a brief amount of time, when the target node successfully rebooted it would be the last node to publish the ID and as such the legitimate node controlled the GUID. It was calculated that on the current network a node is able to update all other nodes on the network which contain a reference to a certain GUID in under 5 seconds.

Another issue found during the experiment was that the node who joined the network last is the node who controls the GUID; this could start a restart competition between the malicious node and the legitimate node. The long-term effects of such a competition on nodes already storing routing data for the GUID is currently unknown.

The amount of time taken for the attacker's GUID to replace the legitimate one is normally very fast, averaging a few seconds. This rapid updating through the network will make the attack harder to spot and also more effective, as an attacker can quickly target the whole network. As the network grows, however, the propagation through the network will be slower.

When used in conjunction with the Sybil attack, it has been demonstrated that this attack can prevent new nodes from joining the network by surrounding it with malicious nodes and preventing the new node from discovering honest nodes.

When we alerted the developers of OpenBazaar to this major bug in their software, they quickly released a patched version which generates its GUID from the PGP key [5], this however does not prevent a clustering attack, where a node can generate thousands of GUIDs and PGP keys in order to be able to "swarm" around a target node and isolate them from the network. Later in this paper, we discuss a stronger solution that is not vulnerable to this weakness.

4.3 Double sniper impersonation attack

Expanding on the aforementioned impersonation attack, it is possible for an attacker to target the entire transaction chain. An attacker would first impersonate the marketplace, and then skillfully impersonate the notary. With the ability to run two nodes from a single machine, this attack has the same cost as the single impersonation attack, and the attack cost will not increase as the network size increases.

For the attack to work, the attacker selects a marketplace to attack and impersonates it using the same method previously described. When a user buys an item from this cloned marketplace and sends the contract containing his signature and chosen notary, the marketplace can see the chosen notary, which is shown by only its GUID. PGP keys, etc. for the notary are not included in the contract; these would allow the attacker to see what notary the target wanted to use and launch their impersonation attack, making the attacker a notary that has the same GUID as the one requested by the client, although as always with a different PGP key. The client would then receive a triple signed contract from the attacker's cloned marketplace and notary.

The attacker would control of two out of the three signatures in the transaction, thus enabling them to withdraw the funds from the multi-signature wallet. There are many ways in which an attacker could ensure their attack is less likely to be detected, such as only impersonating the market for a very brief amount of time, although one issue with this is the amount of time it takes for all nodes to see the attacker's version of the marketplace. In the lab, this was <2 seconds, but it is possible that in the real world with a larger network this time would be increased.

The time taken to replicate the notary node, however, should not serve as a warning to the client that it is under attack. Our findings show that in practice most notaries do not sign contracts straight away; it is a job that is conducted manually by humans and we have witnessed signatures being provided from within a matter of minutes to over a day.

The only chance of the attacker's marketplace/notary being identified as an imposter would be if a client already knew the PGP keys, and compared them to those of the marketplace. For the average client this would not be possible, as the PGP keys for the marketplaces/notaries are not stored anywhere in the network for comparison.

If the attacker were able to impersonate a marketplace such as the Silk Road on Tor, which at its peak had a turnover of USD 1.9 million per month, the attacker would be set to make $60,000 a day. However, no markets of Silk Road's scale currently exist on OpenBazaar - at present the majority sell a small number of items, with the total number of transactions per day still in single figures.

4.4 Takedown attack

Since OpenBazaar marketplaces do not have any redundancy, if an attacker is able to take the marketplace offline, it will be unavailable to the whole network until the marketplace is able to get back online.

It is therefore possible to target a specific marketplace and attack it using a DOS attack. This is trivial as a user requests a marketplace from the DHT and the node's IP address and UDP port are returned. This simple ability to discover the IP address and port of a node makes it very vulnerable to a DOS (denial-of-service) attack. A DOS attack aims to make a server inaccessible to clients by sending large amounts of malicious traffic or by sending malformed packets to use up the servers resources so it is unable to fulfill the requests of genuine users [4]. A successful attack results in no other users being able to connect with the targeted marketplace during the attack.

5. Effective countermeasures

We will now suggest a series of improvements that can be implemented to the network that aim to prevent or significantly increase the cost of attacks so as to make them unfeasible for an adversary with only limited or moderate resources.

Our solutions aim to maintain a high level of decentralization and continue to allow anyone who wants to contribute and use the network to join and use it.

5.1 Countermeasures to the double agent attack

Currently the double agent attack is successful because one party - the client - selects a notary. This makes it is incredibly easy for a client to ensure they also control the notary. Even expensive entry requirements to the network may not deter an attacker from becoming a malicious notary and it would still possible for a low-resourced adversary.

We propose that a user should not be able to be a client, marketplace and notary simultaneously from the same machine. This is because it is impossible to determine the trustworthiness of a pseudonymous individual in different roles; for example, a person who is a trustworthy marketplace may not be a trustworthy notary.

One possible solution is to pool several nodes together and require a majority to agree before the funds are released, although this is a good solution, there are many issues with it, such as increased resources being required per transaction and the question of whether this would be justifiable on a $10 contract. This solution does, however, provide security against an attack or situation where if a single notary goes offline, money can still be awarded correctly.

We recommend using a pool of seven notaries, as during our calculations this gave the best balance between the total cost of conducting a transaction and effective countermeasure to a Sybil attack. It would also be possible and more efficient if the notary pool size were dependent on the level of trust placed into a marketplace, the cost of the item, and the cost to the notaries for the transaction, should anything go wrong. As an example, for a $50 item, a pool of three notaries would be sufficient, while for a $1,000 item a pool of seven notaries would be preferred. This could be automated or left to the user/marketplace to select the level of security they require for a transaction. As trust between the user and marketplace increases, e.g. due to previous successful transactions, the number of notaries in a pool could then be lowered.

We conducted several experiments to evaluate the effectiveness of three possible solutions to prevent collusion between the client and notary in the double agent attack. The first method is the current notary selection where the client selects the notary; the second is the notary pool selection, where seven notaries are randomly put into a pool, which would require an attacker to control the client as well as four out of seven notaries in the pool to conduct the double agent attack. Finally, we evaluate how random selection of a notary would deter the attacker. The adversary in our threat model is a low-medium resourced adversary with a budget of $1,000 a month to use to attack the network.

We calculated the probability of each pool being controlled by a malicious majority, using the equation below:

$$\rho(m) = \binom{k}{m} p^m (1-p)^{k-m}$$

K is the size of the pool, and m is the number of malicious nodes. This equation calculates the probability for a single combination of malicious nodes and non-malicious nodes in a pool. For completeness, we then calculated the probability based on the sum of all possible combinations where the malicious nodes control a majority of the pool.

We created a simulator in order to verify the results from the equation of the probability that a notary, or notary pool, would be able to collude with the malicious client. To achieve this, the simulator would create varying number of honest nodes with a fixed amount of malicious nodes, with this figure being based on the number of nodes that an attacker could compromise. We then calculated the probability that each time a user selected a pool of nodes, a majority of the pool would be malicious. This was repeated 10,000,000 times to improve the accuracy of the results.

Figure 1 shows the chances of selecting a malicious notary or notary pool if the new notary selection was implemented on today's network without implementing our improvements in the network. With the set budget, an adversary would be able to control 3,330 malicious notary nodes and, as shown, it is still highly probable that they could control both the notary and client, even on a network of 50,000 nodes. A surprising find during the experiment was that, for this network, random selection provided a better degree of security to prevent a notary and client colluding in a network of up to 50,000 nodes, after this point the notary pool selection provides greater security.

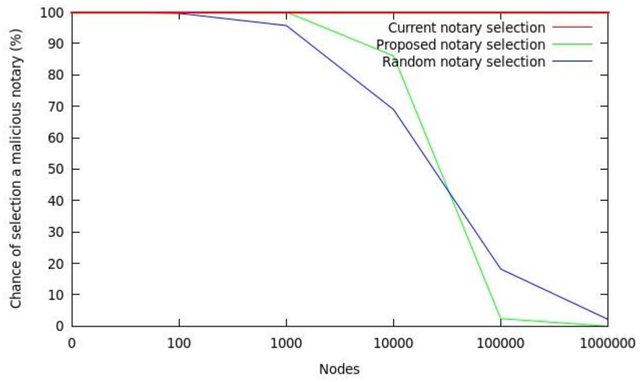

Figure 1. Probability of selecting a malicious notary (pool) using three different notary selection algorithms on the current network.

Figure 2 shows the chance of selecting a malicious notary pool, where at least four out of the seven notary nodes in the pool are malicious. In this simulation, the adversary has the same $1,000 a month to attack the network, but this simulation implements several suggested improvements to the network in order to prevent the Sybil attack. One of these improvements was to generate the GUID of a node based on its IP address, in order to prevent multiple nodes from being hosted on a single machine.

$$GUID = hash\ (IP\_Address \oplus UDP\_Port)$$

In addition to preventing a single node from performing more than one role, such as simultaneously being both a client and a notary, our improvements also require each node to donate 20GB of storage to the network to be used for marketplace mirrors.

These countermeasures aim to dramatically increase the cost of entry to the network in order to make it prohibitively expensive to all but the most well-funded adversary.

If implemented in the OpenBazaar network, the solutions proposed here would solve three possible attacks: the Sybil attack, the impersonation attack and the sniper impersonation attack

Now, the adversary would only be able to control 143 notary nodes in the network compared to the 3,330 notary nodes they were able to control without these improvements.

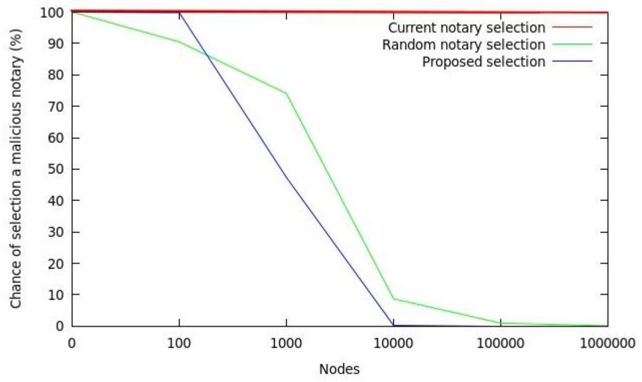

Figure 2. Probability of selecting a malicious notary (pool) using three different notary selection algorithms on a network with our solutions implemented.

We can see from Figure 2 above that randomly selecting a single notary results in a lower chance of a malicious node being selected by around 10% while the network size is less than 200 nodes. As the network size increases, however, the effectiveness of the pool notary solution increases whilst the chances of selecting a malicious notary pool dramatically decreases. Once the network exceeds 100,000 nodes, the chances of selecting a malicious node using both the random selection and notary pool is less than 0.5%.

With such a small difference in selecting a malicious notary using the random selection and pool in a large network, it could be better for performance to use the random selection over a pool of nodes. This only requires one notary to make a decision, rather than waiting for seven notaries to respond. For low-medium value transactions, this would make random selection a better solution in a large network.

The only reason the chances of selecting a malicious notary pool and client is so high while the network is small is that with a budget of $1,000 a month, attackers are able to Sybil attack the network; an attacker with a lower budget would have less success with networks of all sizes. Based on current predictions of network growth, in practice an attacker with a budget of this level would be unable to conduct a Sybil attack on the network.

Overall, we have shown the current method of notary selection to be open to abuse. An attacker controlling just two nodes is able to launch the double agent attack and undermine the security offered by a multi-signature wallet. We have proposed two solutions to reduce the chances of a notary and client colluding, a random selection algorithm, where a single notary is chosen at random, and a notary pool, where notaries are randomly chosen to form a pool and which requires a majority decision to release any funds.

We calculated the probability of selecting a malicious notary or notary pool in the current network and a network which implemented our previous recommendations. The results show that both solutions provide greater security against the double agent attack, and both provide less chance of collusion in a smaller network size when the network has implemented our suggestions compared to the current network. The difference in selecting a malicious client and a randomly selected notary and a notary pool is surprisingly small, especially as the network grows in size.

We therefore recommend that while the network is small (<10,000 nodes) the notary pool method should be used to select the notaries. However, as the network grows it should be down to the individual marketplace or client to select which method they wish to use, with such a slim likelihood of collusion, both methods would be suitable. Random selection would be faster as only one notary needs to sign the contract and, if necessary, make a decision, but, although slower, the pooled notary method would offer greater security and thus be more appropriate for a higher-value contract. We tested these assumptions on a simulated network which was identical in function to the real OpenBazaar network and found our results to be in line with these predictions.

5.2 Countermeasures to takedown attack

The takedown attack was so effective and quick because on the current network, marketplaces are hosted on a single node; there is no data duplication across the network.

To prevent this attack, it would be possible to cache the marketplace when it was requested by the nodes that serve the request; the more frequently requested, the longer the node would keep it. This solution would be beneficial in two ways, firstly it would speed up the network, making popular marketplaces available faster, and it would also prevent an attacker from performing the takedown attack as they would have to take multiple nodes offline; this would be more difficult as an attacker would not know the location of all the nodes containing a cached copy of the marketplace.

To prevent nodes altering the data being stored, it would be important to encrypt the marketplace's data. This would be done by encrypting the data with the marketplace's private key. This means only the marketplace could have created the content; it also increases trust in the network as you can be assured that the marketplace data is from the correct marketplace and not a copy from an attacker conducting the impersonation attack.

This method is very similar to I2P disk cache, although it would now be used for websites and not files, which results in a new issue of how to keep all cached copies of the website up-to-date.

This would be solved by a marketplace sending an announcement out of the network when it would like to update its website. While inefficient, it is the only method where the nodes caching the web page would get the message, as the marketplace would not know which nodes are caching a copy of their page. The nodes caching the web page can then either delete their copy and retrieve a new copy, or simply remove the copy from their cache and wait for the web page to be requested before caching it again.

The proposed solution takes advantage of the previously suggested implementation of requiring each node to donate 20GB of hard drive space to the network; this was to increase the cost of becoming a node, but it is this space that would be used to provide the cache, thus not requiring any more resources from a node to implement this feature.

6. Future Work

This paper has formed a basis upon which further research into OpenBazaar can be conducted. One area of research would be to evaluate the effect a trust value per node would have when conducting these attacks. Could the double agent attack be conducted with two low-trust nodes? Could trust values prevent the impersonation attack?

Another potential area of research for the future is research into a DOS attack unique to OpenBazaar's implementation which would require less resources than conventional DOS attacks and which could be launched from a single node, while being undetectable as the source of the attack.

It would be appropriate to re-run the attacks in this paper on a larger live network, as it would be interesting to compare some of the conclusions drawn in this paper to the realities of a larger network and compare how the change in size of the network affects these attacks.

Finally, perhaps the most interesting research area is the inclusion of anonymity into the network, how would this be achieved? How would users be able to rate other anonymous users? Would it facilitate or prevent the attacks mentioned in this paper and would the addition of anonymity bring a new range of attacks? This particular area of study is the focus of our ongoing research, and we hope to publish more results in the near future.

7. Conclusion

In this paper we have conducted the first analysis of OpenBazaar. We have shown that the current network is small in size and vulnerable to many attacks.

We demonstrated a new and unique attack - the double agent attack. This attack undermines the defenses offered by OpenBazaar in the form of multi-signature, three party contracts. By controlling just two nodes, an attacker was able to conduct the attack with a 100% success rate, regardless of the size of the network.

The final attack we demonstrated was an impersonation attack. We revealed a weakness in the generation of the GUID, which allowed a malicious node to impersonate a node with a high success rate; we showed how this could be done to marketplaces in order to impersonate an established marketplace.

Finally, we looked at solutions and countermeasures to all the attacks we conducted on the OpenBazaar network. We were able to show how the implementation of simple changes could not only increase the cost of conducting the attacks beyond the budget of a low-medium resourced adversary, but the changes were beneficial to the network, providing the network with greater bandwidth as well as storage space.

8. REFERENCES

[1] BazaarBay Statistics. Retrieved June 12 2015, from BazaarBay: http://bazaarbay.org/stats

[2] Cholez, T., Chrisment, I., and Festor, O. Evaluation of Sybil Attacks Protection Schemes in KAD. In *AIMS 2009 – 3rd International Conference on Autonomous Infrastructure, Management and Security,* (Enschede, Netherlands, 2009), Springer, 70-82.

[3] Dingeldine, R, Mathewson, N., and Syverson, P. Tor: The Second-Generation Onion Router. In *SSYM'04 Proceedings of the 13th conference on USENIX Security Symposium – Volume 13,* (San Diego, California, 2004), USENIX Association Berkeley, 21-21.

[4] Garber, L. Denial-of-service attacks rip the Internet. In *IEEE Computer*, 33 (4), 12-17.

[5] Generate GUID Use Signed Pubkey. Retrieved June 10, 2015, from GitHub: https://github.com/OpenBazaar/OpenBazaar/pull/1300

[6] Jenkov, J. Peer Routing Table. Retrieved June 14, 2015, from Jenkov Aps: http://tutorials.jenkov.com/p2p/peer-routing-table.html

[7] Keong Lua, E., Crowcroft, J., Pias, M., Sharma, R., and Lim, S. A Survey and Comparison of Peer-to-Peer Overlay Network Schemes, *IEEE Communications and Tutorial*, March 2004. 1-22.

[8] Maymounkov, P., and Mazières, D. Kademlia: A Peer-to-Peer Information System Based on XOR Metric. In *IPTPS '01 Revised Papers from the First International Workshop on Peer-to-Peer Systems*, (Berkeley, California, 2002), Springer-Verlag London, 53-65.

[9] Slack. Retrieved June 6, 2015, from OpenBazaar: https://openbazaar.slack.com/

[10] Stoica, I., Morris, R., Karger, D., Frans Kaashoek, M., and Balakrishnan, H. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of ACM SIGCOMM '01* (San Diego, California, 2001), ACM, n.p.

[11] Stone, J. Evolution Downfall: Insider 'Exit Scam' Blamed For Massive Drug Bazaar's Sudden Disappearance. Retrieved June 10, 2015, from the International Business Times: http://www.ibtimes.com/evolution-downfall-insider-exit-scam-blamed-massive-drug-bazaars-sudden-disappearance-1856190

[12] The OB1 Team. OpenBazaar is Entering a New Phase with Funding. Retrieved June 12, 2015, from the OpenBazaar Blog: https://blog.openbazaar.org/openbazaar-is-entering-a-new-phase-with-funding/

[13] Wang, P., Tyra, J., Chan-Tin, E., Malchow, T., Foo Kune, D., Hopper, N., and Kim, Y. Attacking the Kad Network. In *Proceedings of the 4th international conference on Security and privacy in communication networks,* (Istanbul, Turkey, 2008), ACM, New York, n. pag.