Gladius Technical Whitepaper

Section 1 - Overview

1.1 - Background

The first known Denial of Service (DDoS) attack took place in 1974 and was executed by a 13 year old who took down 31 computers in one of University of Illinois' computer research labs (Radware, 2017). Over the course of the past 40+ years, these attacks have evolved to become highly coordinated and distristributed. Today, Distributed Denial of Service (DDoS) attacks can be launched by a disgruntled employee to take down a former company's website, coordinated by a group of activists to prevent access to content they find objectionable, or even leveraged by nation states to shut down an enemy's critical infrastructure. Believe it or not, it has actually become easier to launch these attacks over time and it will only get worse as more devices are connected to the Internet.

To tackle this issue, Gladius will focus on the 3 biggest problems associated with today's solutions.

1. **Technical: Centralized Solutions vs. Distributed Solutions**

   Hackers are becoming increasingly more sophisticated and may spend weeks, months, or even years researching their next victim. If they can approximate the amount of mitigation resources being used by their target in terms of bandwidth, processing power, memory, storage and application capability, they can construct a network of distributed computers to bring the target to its knees. It's simple math. If the distributed resources available to the hackers are greater than the centralized resources available to the website, the target is going down!

   To make matters worse, these attacks are growing in size! For example, the largest attacks on record in 2013, 2014 and 2015 reached 300 Gbps, 400 Gbps and 500 Gbps, respectively; while the first Tbps-level attack was recorded in 2016 (Deloitte, 2017). In 2018, we've seen attacks of over 1.7 Tbps (ZDNet, 2018). That's a roughly 6x increase from 2013 to 2018.

   To mitigate against these attacks, service providers and website owners spend ever more money on power, space, cooling, compute infrastructure and network infrastructure. For business executives, this is money that isn't being spent on driving new revenue or developing new products and services. For the IT professionals, they're

constantly requesting more budget to scale for corner cases and worst case scenarios to ensure business continuity.

It's really a technical arms race, where one opponent gets to gather all of the distributed resources they can muster against an opponent that has to continuously procure more and more services and or centralized infrastructure just to defend themselves! To be successful, Gladius has to turn the tables and fight a distributed problem with a distributed solution!

2. **Economic: Attack Costs vs. Mitigation Costs**

When you dig into the economics, the numbers show just how lopsided the battle really is. According to Kaspersky's "IT Security Risks Survey 2017", the financial implications of reacting to a DDoS attack for enterprises jumped from $1.6 million in 2016 to $2.3 million in 2017. For small to medium sized businesses the number jumped from $106 thousand in 2016 to $123 thousand in 2017. This represents annual increases of 43.7% and 16.0% respectively. In additions to these costs, 23% of the respondents said they lost revenue and business opportunities, and 22% expressed a loss of reputation among clients and partners as a direct consequence of a DDoS attack.

Breaking it down even further, Information Technology Intelligence Consulting's "2017 Reliability and Hourly Cost of Downtime Trends Survey" found that 33% of large enterprises with more than 1,000 employees claim that a single hour of downtime would cost their company over $1 million! 98% claimed that an hour of downtime would cost at least $100 thousand. That's approximately $1700 to $17,000 per minute of downtime!

On the other side of the equation we have the cost of launching a DDoS attack. According to Kaspersky Lab, any malicious actor can procure a DDoS service from the dark web. For approximately $20, you can buy a botnet capable of sustaining a 125 Gbps attack for an hour. The price even includes 24/7 dedicated support for the duration of you attack. So, while it costs an enterprise between $1,700 and $17,000 per minute in downtime, it only costs hackers $0.33 per minute to shut them down!

Think about it… On the lower end of the scale, with today's solutions, it costs an enterprise roughly 5000x more to mitigate against a DDoS attack than it takes for a hacker to launch one! To be successful, Gladius has to level the economic playing field!

3. **Business Model: Walled Gardens vs. Collaborative Ecosystems**

A walled garden is simply a closed platform or closed ecosystem where a single entity has complete control over a given solution. While this business model can be beneficial for companies when attacking certain problems, Gladius doesn't believe this approach works for DDoS mitigation.

> Malicious actors that launch DDoS attacks are not bound by walls. Their software typically installs on a variety of hardware configurations, runs on multiple operating systems, leverages the bandwidth of any service provider and can be coordinated to attack any website anywhere in the world. These advanced botnets are typically a collection of 100s of thousands of unrestrained and underutilized resources that can target companies that don't have the same luxuries.
>
> The victims of DDoS attacks typically work with vendors and service providers that work within walled gardens. No matter how great their solution is, it will always be at a disadvantage. Again, it's just a numbers game. If your mitigation software only runs on a single operating system, you're missing the opportunity to leverage existing resources that use a different operating system. If one of your service providers offers a managed service that doesn't interoperate with another one of your service providers, you'll have less resources available to mitigate a large scale attack. Using a constrained solution to fight an unconstrained attacker is a very difficult proposition!
>
> To effectively address this problem, we need to tear down walls and collaborate. The solution will look more like what the hackers use and less like today's offerings. We need an open source solution that can install on a variety of hardware components or operating systems, run over any service provider network and offer companies better automation, transparency, and security while leveling the economic playing field!

The good news is that Gladius will address all of these issues by leveraging an open source (GPLv3), blockchain-based (ERC-20 token), cross-platform (Linux, Windows, macOS), hardware agnostic (Raspberry Pi through High End Data Center Server) solution coupled with a pay as you go collaborative ecosystem.

1.2 - Platform overview

The goal of Gladius is to create a fully decentralized, peer to peer, serverless node network to connect bandwidth and storage pools to websites looking for DDoS protection and expedited content delivery. Anyone with a computer can download and run the Gladius peer client in the background to rent out their unused bandwidth and storage space and earn Gladius Tokens (GLA). Large pools with hundreds, if not thousands, of nodes will then be able to handle a continuous stream of requests to validate website connections and block malicious activity.

Client nodes can be started up on any Linux, Windows, or macOS machine and will automatically run in the background when a user chooses. Users will be part of localized pools where they contribute their bandwidth and small amounts of storage and processing power. Every website request that a user processes will earn them Gladius Tokens. In turn, users can sell these tokens back to websites to create an economic cycle that promotes the growth of the Gladius Network.

Websites looking for protection will simply be able to create an account on the Gladius website, acquire Gladius Tokens, and then request services in a matter of clicks. Once a site is part of the Gladius network, site owners will be able to view statistics about their website.

Users of the software will easily be able to create a local account and wallet, configure the network settings to open any necessary ports, join any pools that they are best suited for, and start earning GLA. Users will be able to toggle when they are renting their spare bandwidth and space in a matter of seconds. There will also be additional configuration settings to automatically toggle the service based upon time of day, other programs running, and even more in-depth user configurable parameters.

1.3 - Tokens

GLA will be a key component to the Gladius Network. The token will be used by websites to buy DDoS protection and CDN services. The majority of the fees charged for protection will go to the node owner (the individual renting out their spare bandwidth and storage space) and the pool manager, with a small percentage going back to protocol development and support. All fees will be denominated in GLA, and are subject to change based on supply and demand.

The previously mentioned node owners who are part of protection pools will, in essence, act as miners and be incentivized with GLA for their network support. Node operators will be paid for their individual work in these pools, and will be rewarded for sharing bandwidth and storage space.

Section 2 - Technical Details

2.1 - Gladius Architectural Description

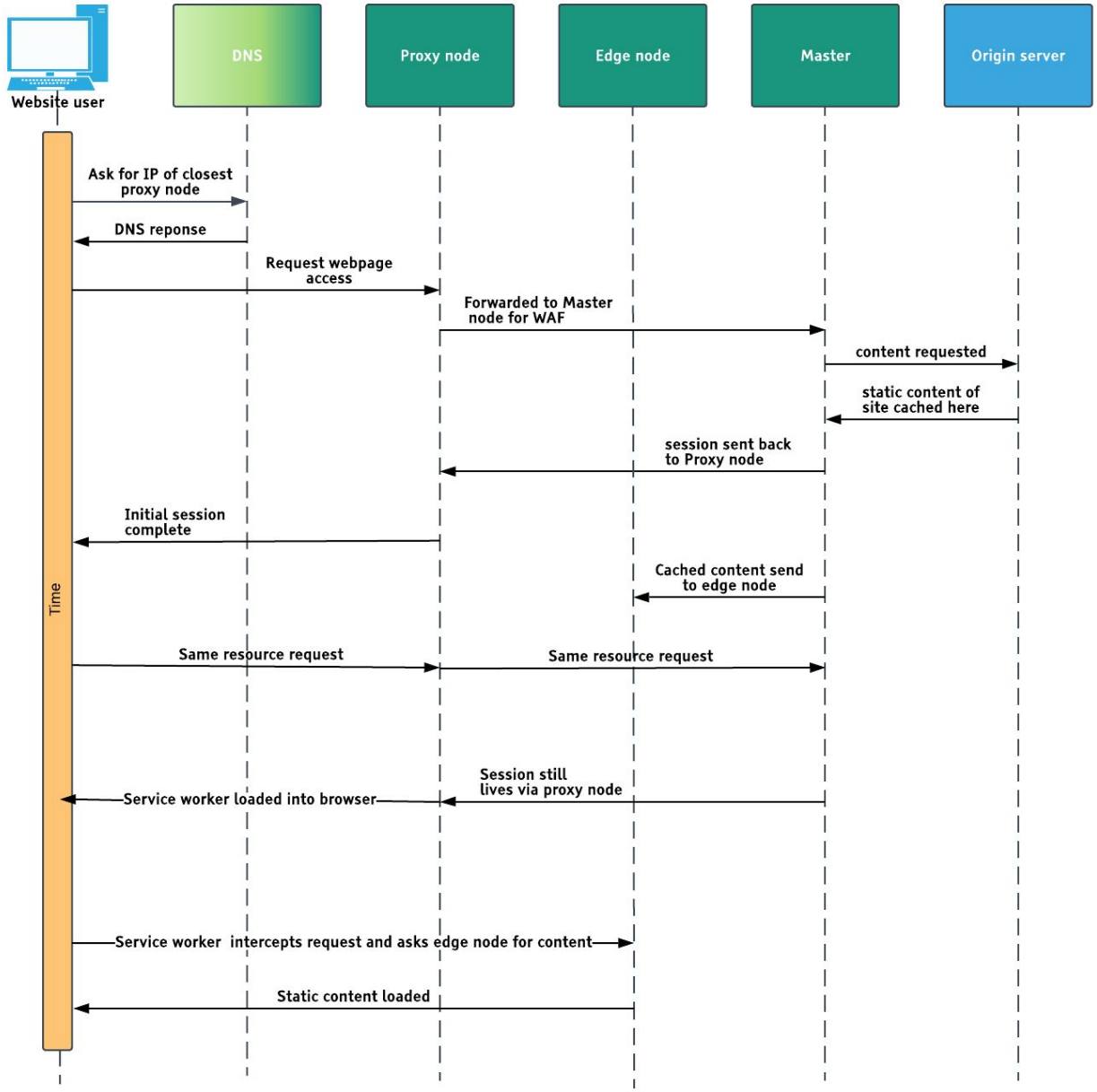
Gladius works similarly to traditional CDN and DDoS protection services by creating a custom proxy which sits between a website's server and the open internet. Unlike traditional networks, the layer that sits between the website and the internet is made up of small clients that split up the traffic verification, cached files, and content over multiple nodes that are able to communicate with each other via a peer to peer overlay network in fractions of seconds.

Pools exist to group nodes into logical demographics (geography, SLA, etc.) to provide a better and faster network experience for a customer. These pools can be seen and accessed through a marketplace where their information on location, pool size, and reputation can be viewed and compared. Pools can additionally consist of only one individual's or organization's resources,

allowing them to run their own instance of a Gladius pool by not approving outside nodes. Using our software, anyone can accelerate their site and combat DDoS attacks.

A critical component of any DDoS protection system is keeping the IP address of the origin server hidden. The Gladius network accomplishes this by having a final proxy (master node) mask the IP from the outside world. The network will also have a built in reputation system to prevent malicious pools. Pools also have the ability to approve individual nodes entering the pool, allowing for a more secure experience.

Below is a general architecture diagram of how requests flow through the network:



2.2 - Network Component Overview

Storage and Marketplace - Ethereum Blockchain
Acts as a decentralized database for storing the pools and their associated service providers, as well as a marketplace where these services are sold. To discourage spam, registrants are required to stake GLA to get added to the list in the marketplace. The staking price can be adjusted to account for changes in the token price.

Node - Proxy and Cache
There are two types of edge nodes in the Gladius network, proxy and content nodes. The proxy nodes serve as a higher availability node that filters incoming traffic without ever stripping the TLS layer. Proxy nodes are pure TCP proxies with a firewall. They have to be more reliable then edge nodes due to being directly connected to the pool DNS servers.

Content nodes are HTTP sources of content. We will touch on how those work later in the paper.

Master node - WAF
Acts as a source of content for local nodes via a content node it runs. Master nodes strip TLS for the Web Application Firewall and act as a barrier between untrusted nodes and the origin server's IP address.

Pool Manager - DNS Anycast system
The manager will be responsible for building and maintaining a DNS Anycast system that is highly distributed and can control DNS requests to proxy nodes. They will also be responsible for uptime checking and load testing the nodes before entry as well as client onboarding. The goal of this system is to lower the technical barriers of deployment and make this process easier.

Peer to Peer Overlay
Due to scaling issues of the Ethereum network, a peer to peer overlay network is used to increase message volume on our network. All nodes contain this component, and every message is cryptographically signed by the node's private key (like an ethereum transaction would be) and is broadcast to their peers. This allows Gladius software to send thousands of messages per second while retaining many of the benefits of a blockchain, minus the transaction fees or delay.

2.3 - Peer to Peer Overlay Network

To combat issues with scaling of the network, we implemented an overlay network at the pool level. This overlay keeps track of network state, and each update of that state requires it to be cryptographically signed by the pool manager or the node sending the message using their Ethereum key. This system ensures a node can validate content on the pool or that it should proxy data to that site. This also allows Gladius to verify a state object when it is loaded as part of a network sync.

Right now, there are two types of messages Gladius supports, node update messages and pool update messages. Messages also allow for delta updates, so the whole object (node data or pool data) doesn't have to be updated if a change occurs. After a few updates are performed, the final network state can be represented in JSON like the below message:


```
▼ object {2}
  ▼ pool_data {1}
    ▼ required_content {2}
      ▼ data [1]
            0 : blog.gladius.io/5573edfbcfb09f07956702f07f21ea2b24ba1dc98f3f09e21815d8219d1ebd87
      ▼ signed_message {4}
        ▼ message {2}
          ▼ content {1}
            ▼ pool {1}
              ▼ required_content [1]
                    0 : blog.gladius.io/5573edfbcfb09f07956702f07f21ea2b24ba1dc98f3f09e21815d8219d1ebd87
            timestamp : 1535656897
          hash : K3Uw+UAdQeTDnzTWe4FXWja0rs2NsHWBU/vrPa0CP3Y=
          signature : C8V7yPF0mnnKgUdWP3OLA7go1EZSidTvqJ7nh6d7R+4tlnVmuFbTdTg4Jgcn1O0diIJO9e/uC+eczH4vKR/AMwA=
          address : 0x6531a634Bbb040B00f32718fa8d9Fa197274f1D0
  ▼ node_data_map {2}
    ▼ 0x3701f76f30B1bCfD48475b5405f22C6787fd0014 {4}
      ▼ content_port {2}
          data : 8084
        ▼ signed_message {4}
          ▶ message {2}
            hash : AemggstYJm+Z4hSZTsyTXhPsjYwcPGGa+9//fI67SUU=
            signature : Ms2OLyaHvSlmJdrL2n9WjJiht4pyuyU7/81MMyrLBMg6zWiNQT1+c6juffW6q59ipKCZDKukpO0dumAxJOhYJAE=
            address : 0x3701f76f30B1bCfD48475b5405f22C6787fd0014
      ▶ disk_content {2}
      ▶ ip_address {2}
      ▶ heartbeat {2}
    ▶ 0x51619A0b38ac59Dff551C26dc0D1edB33c28d866 {4}
```


Pool Data

Breaking down the pool data field, in this example we have a required content list. You can also see the signed message that last updated it, as well as a timestamp of that message to protect against replay attacks where and old captured message is rebroadcast to the network to cause

problems. Another feature of our peer to peer network that can be seen here is the file hash stored right in the file name: “blog.gladius.io/5573edfbcfb09f07956702f07f21ea2b24ba1dc98f3f09e21815d8219d1ebd87”. By doing this we can verify content in the browser to prevent malicious nodes, as well as protect content nodes from downloading incorrect files from their peers.

Node Data

The node data map is a map of the node address to data about that node. Only that node can update its own field, because it has to sign every update which can then be verified by its peers. The data stored here is how we can actually know which nodes have what data, as well as how to access it, and if the node is alive or not through it’s “heartbeat” that it sends to network. This is how we create content links that get downloaded in the browser.

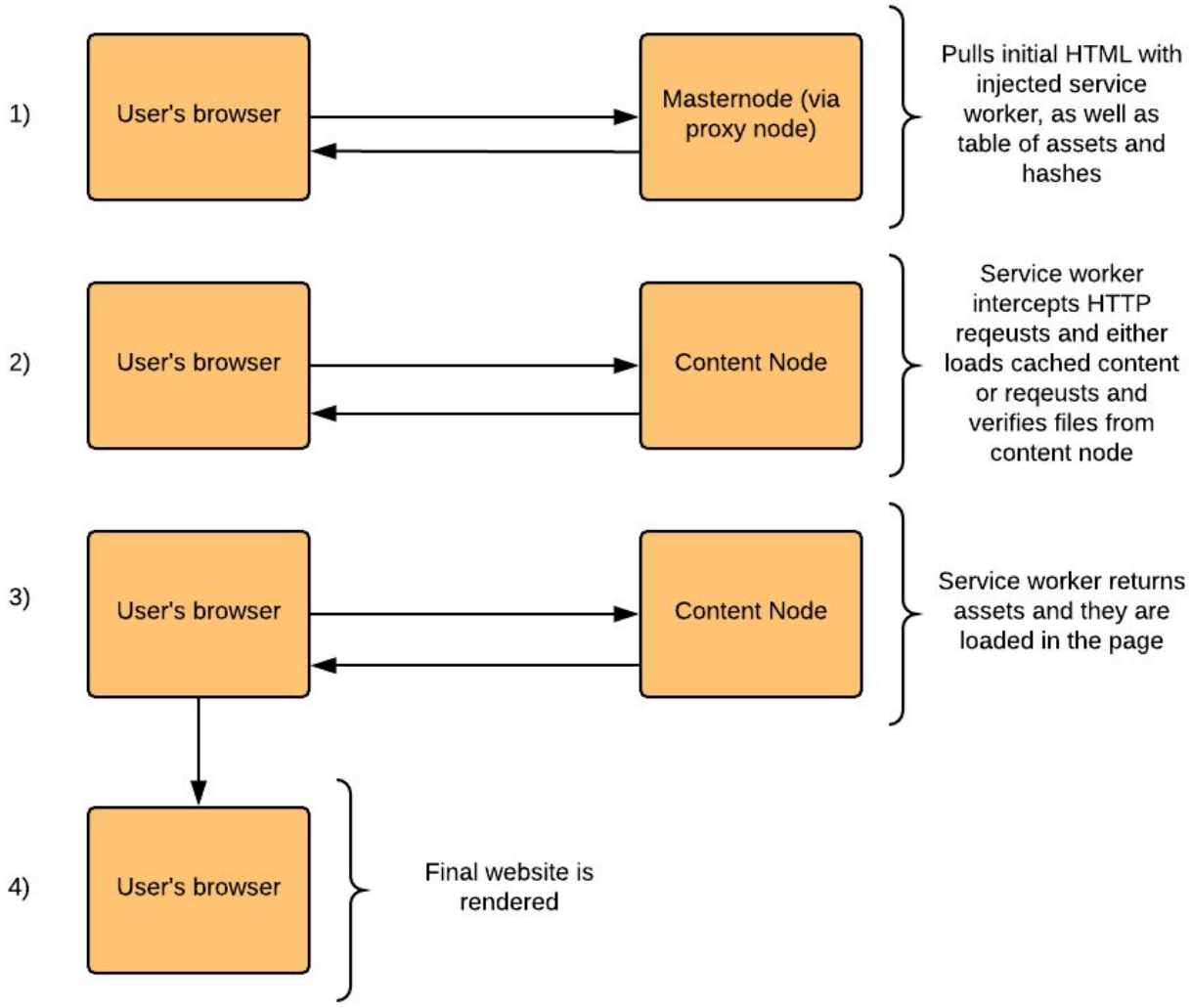
2.3 - Content Delivery Network

Overview

The Gladius network can deliver content faster, and more effectively, than existing architecture because it is decentralized. By having many nodes spread over a large area, a potential client will be connected to a node that is geographically close to them. Instead of having a datacenter located far away but with high capacity, a client will connect to a node with less capacity but that is very topographically close to them. Content nodes keep static files in a cache so once the network has seen a file, requests of that file will be accelerated for every new client.

Content Integrity

A concern of using decentralized content servers is the possibility of malicious actors serving unsafe or illegal content to users. We mitigate this possibility by first serving a small service worker from a trusted master node (through the proxy node) that intercepts the browsers requests for that origin, and in turn asynchronously fetches the matching website content from content peers. These fetched assets are stored in the user’s browser, but not rendered until the integrity of these assets has been verified by calculating the SHA256 hash of each retrieved asset against the expected hashes provided by the master node. Once an asset has passed the integrity check, the service worker will return it and it is rendered on the DOM. The benefit of this approach is the DOM needs little to no modification, and allows things like compression, challenges, and node failover. See the illustration below for further details.



2.4 - Edge Firewall

To combat DDoS attacks Gladius proxy nodes inside the pool perform a number of computations to ensure requests are not malicious. Proxy nodes need to be more available than content nodes because they hold an end users session state.

2.4.1 - Client connection

Clients (browsers, api users) and attackers are directed to the correct entrypoint of the network is through a DNS system. Gladius will be designing APIs to work with existing DNS systems.

We are able to do this because of the ability of our network to keep track of uptime of nodes by monitoring their heartbeat, and find out their IP when it changes. This allows Gladius to quickly build zone files with simple translations from our network state.

Once this system is deployed, it's simply a case of using the best Geoip lookup or other predefined DNS entries to get clients to the right proxy node.

2.4.2 Mitigation Techniques

Node Level - IP Based
Because nodes do not have access to the actual traffic data, they are responsible for handling only IP based DDoS mitigation.

- **Rate Limiting**
  By identifying IP addresses that are constantly making website requests, Gladius can block these requests from getting access to the website. Once the threshold of requests is hit (which is set by the service requestor) the IP will no longer be able to access the site.
- **IP Address Matching**
  Similar to rate-limiting but smarter, IP address matching will be able to group similar IP address together that have known associations with each other. This will take shared information from the pool to block threats ahead of time.
- **Intelligent Geo Matching**
  Additionally Gladius can analyze requests and find geographic anomalies to detect and block attacks by region.

Master Node - WAF Based
Master nodes strip the SSL, and are able to view the raw data. This allows them to do analysis on all the traffic before it is sent to the end server. Here we can look for common attack vectors such as XSS and SQL injection as well as more advanced rules that will continually evolve to meet the industry requirements. Gladius will also actively be researching and acquiring new exploits before they are used maliciously.

Overall a large amount of the protection utilizes anomaly detection to stop malicious attacks. A key feature of the Gladius Network is that pools over time will learn about common attackers and block them preemptively for all other sites and services that are being protected using tools like IP reputation.

2.5 - Smart Contracts

All communication between Gladius modules is encrypted. Every entity in the network (node, pool, client) will generate and store their public key on the Ethereum Blockchain. This allows for

secure off chain encrypted communication in our applications. Below is a detailed explanation of the Gladius Smart Contracts that are in development.

Marketplace Contract

The marketplace is the core smart contract of the Gladius Network. The marketplace is the orchestrator of all the pools in the network. It allows anyone to create a Gladius pool using the marketplace smart contract. The marketplace acts as a source of truth for the network and is able to query how many pools have been generated and who are the owners of each pool (based on Ethereum address). The marketplace also has the concept of public and private pools. A public pool is one that will be advertised to any node that is looking to join a pool via the Gladius Node software. A private pool will not be listed in the Node software and must be searched for explicitly by address. To be able to join the Gladius Marketplace, pools must pay a fee.

To a node, the marketplace serves as a database to browse all public pools. It allows the Gladius Node software to filter and query pools based on their information such as node count, location, rating, etc. In the same vein, the marketplace also allows websites seeking protection to find pools to match their needs of serving content and providing DDoS protection.

Pool Factory Contract

The pool factory goes hand in hand with the marketplace. While users will go through the marketplace to create pools. In the background, the marketplace is actually using the pool factory to "physically" create and track each pool. The pool factory is the true base level orchestrator and tracker of all of the pools in the network. However, for the sake of simplicity and flexibility, the marketplace should always be used to create and track pools. Since the pool factory is a separate contract than the marketplace, Gladius is able to swap it with a new contract if, for example, a serious security vulnerability were to be found in the pool contract without having to swap out the entire marketplace contact. This also allows Gladius to remotely push out security updates on our pools without anyone having to update their software.

Pool Contracts

A pool contract will be created and tracked through the marketplace contract. As mentioned above pools can either be publicly and easily viewable through the marketplace or be "private". A private pool will not be shown through the Gladius Node unless searched for explicitly. Pool contracts keep track of which nodes join the pool, their application server, pool data, master nodes, and seed nodes. Pool contracts serve as a record of a pool and its nodes and is an added layer of security that cannot be taken down unless the entire Ethereum blockchain is taken down. As mentioned, pool contracts will contain plain text data about them (pool data) that is useful when users view pools using the Gladius Node software. Pools will have information

such as node count, bandwidth, rating, etc... so that potential nodes can make informed decisions on which pool(s) they would like to join.

Balance Contract

Each pool will have their own balance contract. This contract will be the code that will take in payment from a website to a pool and then distribute that payment to the pool manager a well as the nodes in the pool. Nodes will be scored relative to the rest of the network by the pool manager, and payments for nodes will be a proportional payout based on their percentage of the total score of the pool. This approach prevents nodes from profiting significantly from DDoS'ing their own pool or creating fake content requests.

Payments will be taken from a pool's client and put into the pools balance contract which acts as an escrow account. Then after a specified amount of time the contract will distribute funds to all of the nodes and the pool manager. The pool manager can then ensure all of the nodes have been paid through each payments transaction hash and manually assist anyone who's payment may have failed for some reason. By using the Ethereum blockchain all payments are transparent and anyone will be able to see how much came in, how much came out, and how much each individual received.

2.6 - Applications To Pools

Reasoning

When conceptualizing the Application Server, we wanted to introduce an ease of life component into the Gladius Network Service. One of the major hangups when utilizing the blockchain as a database was user cost. During our first private beta we quickly saw that the hardest part of the application process was acquiring Ethereum (ETH) in order to pay transaction costs to apply to a pool.

The second disadvantage of using the blockchain for applications was time to execution. During our initial tests with our beta group, we found application times to take several seconds on average. This made our application seem sluggish and not performant in comparison to typical services. Imagine if it took half a minute to several minutes to sign up for Facebook. This was not ideal and had to be addressed.

The final reason to move applications to a server was to increase the security of our users. When using the blockchain to handle applications, any and all data is public, on chain, forever. For our first beta, we did implement data encryption prior to submitting any data to the blockchain, however, the potential for software bugs or security flaws in permanent public data did not seem appealing to Gladius or its users.

Due to these core issues, we decided that a separate server that the pool controls could solve all of these issues while still leveraging the benefits of blockchain technology.

Data Structure

The Application Server structure is designed in a way that collects limited information about a user and reports it to the Pool Manager alone. Due to its nature as an application store, the server can collect very basic information: wallet address, name, email, ip address (for communication with the network), location, biography (reason of application), and location (if reported).

We designed the server to be able to live on its own and to be utilized only inside a pool. In other words, the data that gets collected is only accessible by the user or Pool Manager. Each pool would also have their own independent instance of an Application Server. The only piece of information that could identify a user between servers is their wallet address.

Authentication

Our Application Server does not use passwords for authentication, but instead uses one of the benefits of the blockchain, message signing. Any request to the application server is signed to identify the user making the request to ensure they have permission to access or verify submitted data is theirs. We are able to create these unique profiles quickly and at no cost to the user. The identifiable key is stored on the user's machine and is protected by a passphrase locally. When the user's key is unlocked, we can sign and ensure data communication is from the reported user. This makes profile sign on, registration, and data transmission virtually seamless but secure.

Decentralized Nature

One of the first ideas to solve the application process was to create a single source of truth. Creating a single server meant other issues. When dealing with DDoS attacks, having a single point of failure and reliance is a problem. We designed the application server to be "removable" for the core parts of our system. If the Application Server went down, it isn't needed to maintain any other part of the network.

Another reason to keep the server on a pool by pool basis is to maintain a decentralized network. We did not want a single aspect of the network to be able to be compromised threatening the reliability, or have Gladius control all user data. Each user's profile or application lives independently on each pool giving a pool a community of its own, maintained by the Pool Manager.

Resources

DataCenter Knowledge (2017). New Workloads, Cost Pressures Drive Up Data Center Power Densities. Downloaded from https://www.datacenterknowledge.com/power-and-cooling/new-workloads-cost-pressures-drive-data-center-power-densities on September 21, 2018.

Deloitte (2017). Global TMT Predictions 2017. Downloaded from https://www2.deloitte.com/us/en/pages/technology-media-and-telecommunications/articles/tmt-predictions.html on September 20, 2018.

Forbes (2015). 30% Of Servers Are Sitting "Comatose" According To Research. Downloaded from https://www.forbes.com/sites/benkepes/2015/06/03/30-of-servers-are-sitting-comatose-according-to-research/#1c35b43959c7 on September 24, 2018.

ITIC (2017). 2017 Reliability and Hourly Cost of Downtime Trends Survey. Downloaded from http://itic-corp.com/blog/2017/05/hourly-downtime-tops-300k-for-81-of-firms-33-of-enterprises-say-downtime-costs-1m/ on September 20, 2018.

Kaspersky-B2B (2017). IT Security Risks Survey 2017. Downloaded from https://www.msspalert.com/cybersecurity-research/kaspersky-lab-study-average-cost-of-enterprise-ddos-attack-totals-2m/ on September 20, 2018.

Markets and Markets (2017). Content Delivery Network Market by Type - Global Forecast to 2022. Downloaded from https://www.marketsandmarkets.com/Market-Reports/content-delivery-networks-cdn-market-657.html on September 28, 2018.

Markets and Markets (2018). DDoS Protection and Mitigation Market by Component - Global Forecast to 2023. Downloaded from https://www.marketsandmarkets.com/Market-Reports/ddos-protection-mitigation-market-111952874.html on September 28, 2018.

Radware (2017). History of DDoS Attacks. Downloaded from https://security.radware.com/ddos-knowledge-center/ddos-chronicles/ddos-attacks-history/ on September 20, 2018.

ZDNet (2018). New world record DDoS attack hits 1.7 Tbps days after landmark GitHub outage. Downloaded from https://www2.deloitte.com/us/en/pages/technology-media-and-telecommunications/articles/tmt-predictions.html on September 20, 2018.

Legal Disclaimer

The beta version of the Software (the “Beta Software”) is believed to contain defects, errors or inaccuracies that could cause failures, corruption and/or loss of data or information. By accessing the Beta Software, you acknowledge that (i) the primary purpose of your use of the Beta Software is to enable us to conduct testing, obtain feedback, and to identify defects in theBeta Software, and (ii) you have been advised to safeguard important data, to use caution and notto rely in any way on the correct functioning or performance of the Beta Software and/or any accompanying materials.

THE BETA SOFTWARE IS AVAILABLE "AS IS", AND GLADIUS NETWORKS LLC (“GLADIUS” or “WE”) EXPRESSLY DISCLAIMS ANY AND ALL WARRANTIES, WHETHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, ACCURACY, COMPLETENESS, AND PERFORMANCE. WE MAKE NO WARRANTIES, WHETHER EXPRESS OR IMPLIED, AS TO THE SUITABILITY OR USABILITY OF THE BETA SOFTWARE OR ANY OF ITS CONTENT, AND WE DO NOT WARRANT THAT THE BETA SOFTWARE WILL MEET YOUR REQUIREMENTS OR THAT OPERATION OF THE BETA SOFTWARE WILL BE UNINTERRUPTED OR ERROR FREE.

TO THE EXTENT NOT PROHIBITED BY APPLICABLE LAW, YOU ASSUME ALL RISKS AND ALL COSTS ASSOCIATED WITH THE TESTING, INSTALLATION, OR USE OF THE BETA SOFTWARE, INCLUDING, WITHOUT LIMITATION, ANY DAMAGE TO ANY EQUIPMENT, SOFTWARE, INFORMATION OR DATA, AND IN NO EVENT WILL GLADIUS BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY DAMAGES WHATSOEVER, INCLUDING DAMAGES FROM LOST PROFITS, LOSS OF GOODWILL, OR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, OR DAMAGES FOR NEGLIGENCE OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR WORK

STOPPAGE, LOSS OF DATA, COMPUTER FAILURE OR MALFUNCTION, OR FOR ANY OTHER DAMAGE OR LOSS. IN NO EVENT SHALL GLADIUS BE LIABLE FOR ANY DAMAGES EVEN IF GLADIUS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL GLADIUS' TOTAL LIABILITY TO YOU FOR ALL DAMAGES (OTHER THAN AS MAY BE REQUIRED BY APPLICABLE LAW) EXCEED THE AMOUNT OF FIFTY DOLLARS ($50.00). THE FOREGOING LIMITATIONS WILL APPLY EVEN IF THE ABOVE STATED REMEDY FAILS OF ITS ESSENTIAL PURPOSE.

YOU ARE CHOOSING TO RUN GLADIUS SOFTWARE AT YOUR OWN RISK. GLADIUS IS NOT RESPONSIBLE FOR ANY COSTS OR FEES ASSOCIATED WITH YOUR INTERNET SERVICE PROVIDER, INCLUDING ANY EXCESS FEES OR OVERAGE CHARGES FOR EXCEEDING DATA PLANS. GLADIUS IS NOT RESPONSIBLE FOR YOUR INTERNET SERVICE AGREEMENT OR ANY CHANGES IN SERVICE RESULTING FROM THE USE OF GLADIUS SOFTWARE.

GLADIUS, IN THEIR SOLE DISCRETION, WILL MAKE ALL DECISIONS REGARDING REWARDS AND EARNINGS RELATED TO THE BETA PROGRAM.