VOL. 10. NO. 4 MAY 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480 /jitk.v10i4.6628

JITK (JURNAL ILMU PENGETAHUAN DAN TEKNOLOGI KOMPUTER)

ZTSCAN: ENHANCING ZERO TRUST RESOURCE DISCOVERY WITH MASSCAN AND NMAP INTEGRATION

Reikal Taupaani[1]; Ruki Harwahyu[2*]

Departement of Electrical Engineering, Faculty of Engineering[1,2]
University of Indonesia, Depok, Indonesia[1,2]
https://ee.ui.ac.id[1,2]
reikal.taupaani@ui.ac.id[1], ruki.h@ui.ac.id[2*]

(*) Corresponding Author
(Responsible for the Quality of Paper Content)

Abstract— *Implementing Zero Trust Architecture (ZTA) requires a comprehensive understanding of network assets as a fundamental step in implementing security policies. This study proposes ZTscan, an automated tool to increase the efficiency of network asset resource discovery. This proposed tool is then made open source in Github for anyone to evaluate and extend. The research constructs a GNS3-based testing scenario to evaluate the performance of the proposed tool against other scanning tools, including standalone Nmap, Masscan, RustScan, and ZMap. The evaluation focuses on three key metrics: accuracy, scanning speed, and generated data throughput. Experimental results demonstrate that ZTscan achieves 100% accuracy, matching Nmap_Pingsyn while outperforming faster tools such as Masscan, ZMap, and RustScan in precision. ZTscan completes scans 10.64%, faster than Nmap TCP SYN scan while maintaining comparable high accuracy. In terms of throughput, ZTscan reaches a stable peak throughput that is 13.8% lower than Nmap TCP SYN scan without causing disruptive traffic spikes. The findings of this study serve as a reference for resource discovery strategies in ZTA implementation, particularly in scenarios that require fast and accurate network scanning while minimizing potential disruptions or network interference.*

Keywords: *network scanning, Nmap, masscan, resource discovery, zero trust architecture.*

Intisari— *Menerapkan Zero Trust Architecture (ZTA) menuntut pemahaman menyeluruh terhadap aset jaringan sebagai langkah awal dalam penerapan kebijakan keamanan. Penelitian ini mengusulkan ZTscan, yang merupakan alat otomatis untuk meningkatkan efisiensi dalam proses resource discovery aset jaringan. Alat yang diusulkan ini kemudian dibuat open source di Github agar dapat dievaluasi dan dikembangkan oleh siapa saja. Penelitian ini membangun skenario uji berbasis GNS3 untuk mengevaluasi kinerja metode yang diusulkan dibandingkan dengan alat pemindaian lainnya, yaitu Nmap standalone, Masscan standalone, RustScan dan Zmap. Evaluasi dilakukan berdasarkan tiga metrik utama: keakuratan hasil, kecepatan pemindaian, dan throughput data yang dihasilkan. Hasil eksperimen menunjukkan bahwa ZTscan mencapai akurasi 100%, sejajar dengan Nmap_Pingsyn, sementara mengungguli alat yang lebih cepat seperti Masscan, ZMap, dan RustScan. ZTscan menyelesaikan pemindaian 10,64% lebih cepat daripada pemindaian TCP SYN Nmap dengan akurasi yang sebanding. Dalam hal throughput, ZTscan mencapai puncak throughput stabil yang 13,8% lebih rendah dibandingkan pemindaian TCP SYN Nmap tanpa menyebabkan lonjakan trafik yang mengganggu. Hasil penelitian ini dapat menjadi referensi dalam strategi resource discovery untuk implementasi ZTA, terutama dalam skenario yang membutuhkan pemindaian jaringan yang cepat dan akurat dengan menekan potensi terjadinya disrupsi atau gangguan jaringan.*

Kata Kunci: *pemindaian jaringan, Nmap, masscan, penemuan sumber daya, arsitektur tanpa kepercayaan.*

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 4 MAY 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i4.6628

INTRODUCTION

Various types of attacks, such as web defacement, ransomware infections, and personal data theft, reflect the weak cybersecurity posture in Indonesia. The Cybersecurity Landscape report [1], notes that one of the main causes of the high number of cyberattacks is misconfiguration, which occurs when IT administrators fail to implement or properly configure adequate security controls on IT assets. For example, an unsegmented internal network becomes an easy target for attackers to conduct lateral movement and malware propagation [2],[3]. So far, network security solutions have primarily focused on protecting against external threats using a perimeter-based approach, such as employing EDR (Endpoint Detection and Response), IDS (Intrusion Detection Systems), and NGFW (Next-Generation Firewalls) [4],[5],[6]. However, this approach is less effective in addressing threats operating within an unsegmented internal network [7],[8],[9]. Therefore, securing internal network traffic requires a more comprehensive approach, namely by implementing ZTA at the network level [10],[11],[12].

ZTA (Zero Trust Architecture) is a modern network security paradigm that eliminates implicit trust in entities within the network [13]. Implementing ZTA requires full visibility of all network assets as an initial step [14]. However, many organizations face challenges in adopting ZTA due to limitations in identifying and inventorying network assets, especially those that have not yet established a mature cybersecurity posture and still rely on traditional perimeter-based security models [7]. To determine the state of network assets, resource discovery techniques are used, which involve scanning processes to identify devices connected to the network and the services running on them [15]. Efficient resource discovery methods are crucial for organizations that lack an adequate asset inventory system, enabling them to transition to ZTA effectively.

The study [14] proposes a migration framework towards ZTA consisting of six key steps. One of the crucial initial steps is context assessment, which involves understanding the current state of network assets. One method that can be used in this stage is resource discovery by leveraging network scanning tools. The findings of this study also emphasize the importance of automation in asset identification processes, enabling organizations to improve efficiency and accuracy in detecting and managing network resources. Research [16] highlights the challenges of resource discovery in the context of ZTA. One of the primary challenges is the significant resource requirements, both in terms of time and cost. Additionally, the resource discovery process can potentially cause network disruption if the scanning tools generate excessive traffic [17]. This issue serves as the focus of this study, streamlining the resource discovery process to make it more efficient, accurate, and minimally disruptive to the network.

Several studies have already explored the optimization of Nmap, including [18],[19], which developed a graphical user interface (GUI) to facilitate Nmap usage for beginners. However, their proposed tool does not provide a performance evaluation of Nmap, such as scanning speed, detection accuracy, or network impact. This limitation indicates that further research is needed, not only to improve the usability of network scanning tools but also to optimize their performance and efficiency in the context of ZTA implementation. A study by [20] investigates the application of Nmap in network security assessment across various companies. The study demonstrates that Nmap is an effective tool for identifying open ports, running services, and operating systems used by a host. However, this research does not specifically evaluate the impact of scanning on network performance or its effectiveness compared to other tools s uch as Masscan or RustScan.

Research [21] compares the efficacy of three port scanning tools which is Nmap, Zmap, and Masscan, to assess accuracy and system resource efficiency. The study finds that all three tools have similar accuracy levels in identifying network assets, with no significant differences in false positives or false negatives. However, the study focuses more on how scanning affects host utilization rather than its overall impact on the network. By not considering the throughput generated by each tool, this study does not fully reflect the real-world impact of scanning, particularly on live networks. Study [15] applies Metamorphic Testing (MT) to evaluate and compare Nmap and Masscan. This study confirms the fundamental differences between the two tools: Nmap operates in a synchronous mode, making it more accurate for in-depth scanning, whereas Masscan operates asynchronously, allowing for much faster scanning but with lower accuracy in identifying detailed services. The results indicate that Nmap is more suitable for in-depth security analysis, while Masscan is more effective for large-scale rapid probing. However, this study is limited to these two scanning tools and does not evaluate

other popular alternatives in the cybersecurity community, such as Zmap and RustScan.

Research [22] highlights that even brief Nmap scanning activities can cause measurable increases in CPU and memory utilization, leading to performance degradation that persists beyond the end of the scanning event. This finding underscores the potential risks of network disruption from resource discovery processes, particularly in live operational environments. It provides critical evidence that resource-efficient and minimally disruptive scanning strategies are essential, reinforcing the importance of optimizing Nmap or similar tools in ZTA implementation efforts. Previous studies have evaluated individual scanning tools like Nmap, Masscan, Zmap, and RustScan in terms of speed, accuracy, or system resource impact. However, there remains a lack of systematic integration between high-speed and high-accuracy scanning methods, particularly for supporting ZTA migration. Additionally, limited research addresses scanning performance across multi-subnet networks with diverse host types while considering the network stability impact caused by throughput. To address this gap, this study introduces the following novelties:

1) This study proposes *ZTscan*, a Python-based integration of Masscan and Nmap to combine high-speed and in-depth scanning.
2) Evaluates ZTscan not only based on accuracy and speed but also on network throughput impact.
3) Tests ZTscan in a simulated multi-subnet network with various host types for more realistic analysis.
4) Provides a practical and efficient network discovery solution to assist organizations with limited infrastructure in transitioning to ZTA.

The results of this study position ZTscan as a helpful tool for organizations aiming to implement ZTA, particularly in the critical initial phase of resource discovery. By seamlessly blending Masscan's rapid scanning with Nmap's detailed analysis, ZTscan enables more accurate and efficient resource discovery while minimizing network disruptions.

MATERIALS AND METHODS

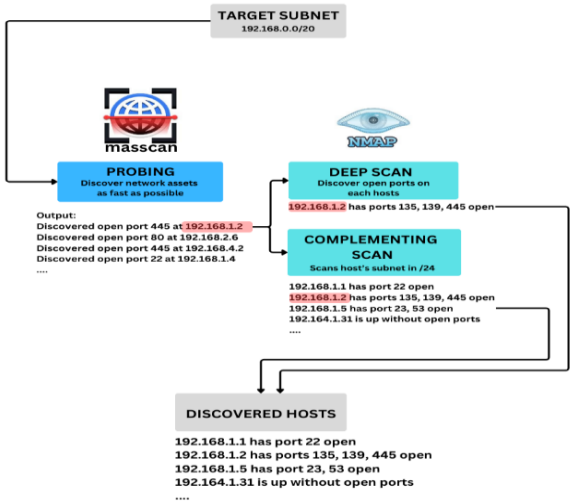
ZTscan Proposed Method

This research proposes a tiered approach to resource discovery by combining Masscan and Nmap to improve network scanning efficiency and accuracy. There are three main functions in this scanning process: Probing, Complementing Scan, and Deep Scan, as shown in Figure 1. The theoretical basis for this approach is drawn from Study [15] ,[21], which evaluates and compare Nmap and Masscan. The study confirms the fundamental differences between the two tools: Nmap operates in a synchronous mode, making it more accurate for in-depth scanning, whereas Masscan operates asynchronously, allowing for much faster scanning but with lower accuracy in identifying detailed services. Based on these findings, this research leverages the strengths of both tools by using Masscan for rapid probing (Probing) and Nmap for detailed verification and analysis (Complementing and Deep Scan).

1) Probing

The scanning process begins with the probing stage, where Masscan is used to perform a quick scan of the target subnet, for example, 192.168.0.0/20 . Masscan is chosen for its ability to conduct large-scale scanning at high speed. At this stage, only the 20 most commonly used ports are checked to avoid excessive network load [23]. The output of this stage is a list of hosts with open ports.

However, Masscan has limitations in detecting active hosts that do not have open ports. If a host does not respond to scanning because all its ports are closed or a firewall blocks responses to the scan, that host will not appear in Masscan's scan results.



Source: (Research Results, 2025)

Figure 1. ZTscan proposed method

2) Complementing Scan

To overcome the limitation of detecting hosts without open ports, an additional scan is performed on the target subnet. In this stage, each detected host triggers a scan of its corresponding /24 subnet to discover other potentially active hosts that were not detected in the initial probing phase.

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 4 MAY 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i4.6628

The Complementing Scan is initiated in parallel with the Deep Scan using Python multithreading to accelerate the scanning process. The scan of a specific subnet is conducted only once when it is identified to prevent redundant scans. This scan operates in fast mode (-F), scanning only 100 commonly used ports to optimize the subnet scanning process [24].

3) Deep Scan

At this stage, each host detected from the probing phase undergoes a more detailed scan to identify additional open ports and running services. This scanning process runs in parallel with the Complementing Scan.

This stage aims to enhance accuracy and reduce false positives by implementing the TCP SYN port scan (-PS) using Nmap. This method allows the scanning process to continue even if a host blocks ICMP (ping), as commonly found in Windows operating systems with active firewalls [24].

Implementation in Python

The implementation of the ZTscan method combines Masscan and Nmap to optimize network scanning for resource discovery. This implementation uses Python to integrate both scanning tools, automate the process, and ensure scalability and accuracy. The script performs the three key functions: Probing, Complementing Scan, and Deep Scan, leveraging multiprocessing to improve efficiency. The Python script for ZTscan is available on GitHub[25].

The core structure of the implementation involves calling external tools (Masscan and Nmap) through Python's subprocess module and handling parallel tasks using the concurrent.futures library for multithreading. The output of each scan is processed and stored in a CSV file for later analysis.

```
Input
target_subnet : The network range to be probed
Function
Function probe(target_subnet):
  Run Masscan on target_subnet
  For each line in Masscan output:
    Parse IP and port
    If valid IP and port:
      If IP not in results:
        Start deep_scan(IP) and complementing_scan(IP)
        in parallel
      Else:
        Print "Host already scanned"
  Wait for deep_scan and complementing_scan to finish
```

Source: (Research Results, 2025)

Figure 2. ZTscan Probing Pseudocode

The probing phase (Figure 2) begins by executing Masscan on the specified target subnet, scanning a limited set of common ports to quickly identify active hosts. As Masscan produces output, each line is parsed to extract the IP address and port information. If the extracted IP and port are valid, the script checks whether the IP has already been recorded in the scan results. For new hosts, two parallel tasks (using Python Multiprocessing) are immediately launched: a Deep Scan to perform detailed port and service detection on the host, and a Complementing Scan to explore the host's /24 subnet for additional active devices. Hosts already recorded are skipped to avoid redundant work. Once all Masscan output has been processed, the function waits for all deep and complementing scan tasks to complete before concluding the probing phase.

```
Input
target_host : IP address detected during the probing phase
Function
Function deep_scan(target_host):
  Run Nmap on target_host
  Parse Nmap results for open ports and subnet
  If target_host not in scan results:
    Write results to 'scan_result.csv'
  Else:
    Update scan results with new ports
Function complementing_scan(target_host):
  Run Nmap on target_host + /24
  Parse Nmap results for open ports and hosts
  For each host in results:
    If host not in scan results:
      Write host and ports to 'scan_result.csv'
    Else:
      Update scan result with new ports
```

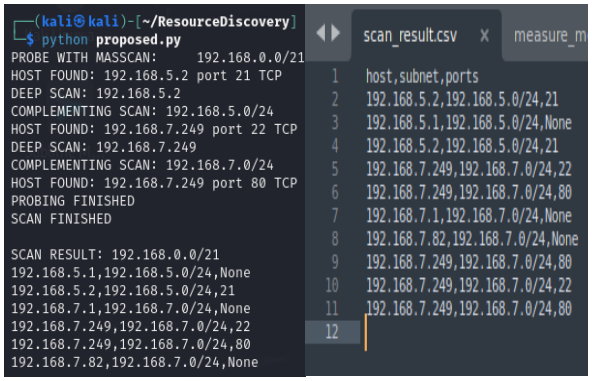
Source: (Research Results, 2025)

Figure 3. ZTscan Complementing and Deep Scan Pseudocode

The Deep Scan function is responsible for performing a detailed analysis of each discovered host. It starts by running Nmap on the target host to gather information about open ports and associated subnet details. After parsing the Nmap output, the script checks whether the host already exists in the scan results. If the host is new, its scan data is written into 'scan_result.csv'; if it already exists, the scan results are updated by adding any newly discovered ports.

In parallel, the Complementing Scan function expands discovery by scanning the entire /24 subnet of the target host using Nmap. After parsing the subnet scan results, each host found is checked against the existing scan records. New hosts are added to 'scan_result.csv', while existing entries are updated with any additional open ports identified. This approach ensures that even hosts without initially open ports are detected and recorded,

providing a broader and more accurate network view.



Source: (Research Results, 2025)

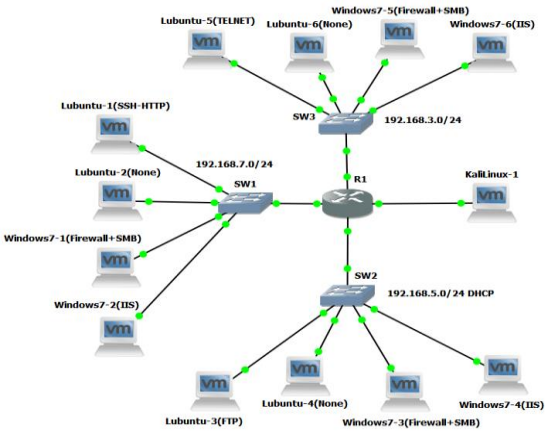
Figure 4. ZTscan Output

The final output of this entire process is a list of discovered hosts in CSV format (Figure 4), complete with information on open ports. This resource discovery result is essential for implementing ZTA, as it serves as the foundation for designing the Zero Trust network architecture [14]. This approach enables a more efficient and systematic scanning process by leveraging Masscan's speed for initial probing and Nmap's precision for in-depth analysis. Additionally, the complementary subnet scanning mechanism ensures that neighboring hosts are also detected, providing broader and more accurate scan coverage.

Testing Scenario in GNS3

To evaluate the effectiveness of the proposed method, this study builds a simulated scenario using GNS3, representing an organization's internal network topology. The evaluation compares ZTscan with other scanning tools, namely standalone Nmap, standalone Masscan, RustScan, and Zmap. Each method's performance is measured based on three metrics: result accuracy, scanning speed, and network throughput.

The testing is conducted in a simulated environment with a network topology consisting of three subnets, as shown in Figure 5. Each subnet is assigned a /24 CIDR block and contains Lubuntu and Windows 7 virtual machines (VMs). The method used are Internal Scanning, where the scanning which agents are located within the enterprise network and scan the internal address space (local-to-local scan) [26]. The entire scanning process targets 192.168.0.0/21 to cover all subnets.



Source: (Research Results, 2025)

Figure 5. Testing Scenario Network Topology

The testing was conducted in a simulated environment using GNS3 with a network topology consisting of three subnets. Each subnet used a /24 CIDR block and contained Lubuntu and Windows 7 virtual machines (VMs). The entire scanning process targeted 192.168.0.0/21 to cover all subnets.

Each subnet contained hosts with unique characteristics. Some hosts ran Lubuntu, while others used Windows 7 with different firewall configurations. Windows 7 (Firewall+SMB) had its firewall enabled and port 445 open, which blocked ICMP ping responses. Some hosts had active services with open ports, while others had no accessible services but still responded to ICMP ping. Routers and Lubuntu (None) in each subnet had no open ports, making them undetectable using port-based scanning methods.

The scenario reflected real-world challenges in network scanning, where not all devices responded in the same way. Hosts with active services and open ports were easily detected using Masscan. Hosts that had no open services but still responded to ICMP ping required more specific techniques, such as Nmap TCP ping scans.

The scanning tools were executed from a Kali Linux machine configured with 2 vCPUs and 2 GB RAM. Each tool was tested five times in the test topology to obtain consistent and accurate data. The tools and commands used for comparison against ZTscan are as follows:

1) Nmap standalone

Nmap is the most commonly used network scanning tool for discovering hosts and services within a network[27]. In the first approach, a basic scan (Nmap_Basic) was executed using the command `nmap -n <target>`. This scan was conducted without performing a reverse DNS

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 4 MAY 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i4.6628

lookup, which significantly reduced the overall scanning time by eliminating the additional step of resolving IP addresses to domain names.

To address scenarios where ICMP echo requests might be disabled, an alternative scanning method was applied using the command `nmap -n -PS <20_common_ports> <target>`. This technique utilized a TCP SYN (Nmap_Pingsyn) ping to 20 commonly used ports, allowing the scanner to identify active hosts based on their response to these connection attempts. By leveraging this approach, Nmap was able to detect hosts that would otherwise remain invisible to traditional ICMP-based discovery methods.

2) Masscan *standalone*

Masscan was employed as a standalone tool due to its exceptional speed in performing large-scale network sweeps. Designed for high-performance asynchronous scanning, Masscan is capable of scanning entire network ranges in significantly less time compared to traditional scanners [15][28]. Masscan was executed (Masscan_Basic) using the command `sudo masscan --rate=5000 -p <20_common_ports> <target>`. The `--rate=5000` parameter controlled the scanning speed, allowing up to 5,000 packets per second to be sent without compromising accuracy. This high-speed approach enabled rapid identification of active hosts and open ports within the target network, making it particularly effective for large-scale reconnaissance.

3) RustScan

RustScan was utilized for its ability to rapidly identify open ports while seamlessly integrating with Nmap for deeper analysis [27]. The command executed (Rustcan_Basic) was `rustscan -p <20_common_ports> --ulimit 5000 --accessible -a <target>`. The `--ulimit 5000` option increased the file descriptor limit, allowing up to 5,000 simultaneous connections, which significantly enhanced scanning efficiency.

4) ZMap

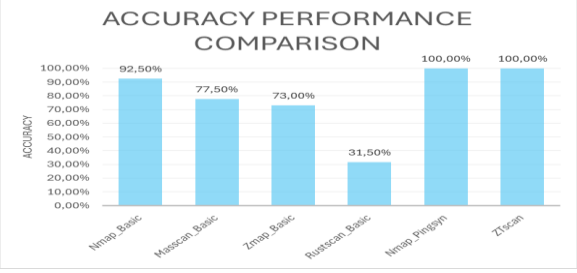
ZMap was designed for large-scale internet surveys, making it one of the fastest network scanners available. Unlike traditional scanners, ZMap employs a stateless scanning approach, allowing it to send probe packets without maintaining connection states. This method significantly reduces overhead, enabling ZMap to scan the entire IPv4 address space in under an hour using a single packet per port scan [29]. The command used (Zmap_Basic) was `sudo zmap -p <20_common_ports> -q <target>`. Here, `-p <20_common_ports>` specifies the commonly scanned ports, while `-q` enables quiet mode, minimizing unnecessary output. This configuration allowed ZMap to efficiently identify open ports across a wide network range with minimal resource consumption.

Performance Metrices

The evaluation of scanning performance in this research is based on three key metrics: scanning accuracy, scan time, and network throughput during the scan. Scanning accuracy is assessed by comparing the number of hosts and open ports detected by each scanning tool against the actual number present in the predefined network scenario. This helps determine the reliability of each tool in accurately identifying network assets.Beyond accuracy, the efficiency of asset detection is also a major concern. To measure this, the scan duration is recorded from the moment the scan starts until all hosts in the network are detected. The performance of each tool is then compared to determine which one identifies network assets the fastest. During the scanning process, network throughput is monitored to assess the impact of scanning tools on network traffic. This monitoring is conducted via router R1 in the topology illustrated in Figure 5. Traffic data generated by the scanning tools is collected and analyzed to understand how each tool affects bandwidth usage and network stability throughout the scanning process.

RESULTS AND DISCUSSION

The accuracy comparison results are illustrated in Figure 6. Nmap_Basic, with an accuracy of 92.50%, demonstrates a high detection rate but struggles with hosts that block ICMP ping. Since Nmap relies on ICMP for host discovery, it considers such hosts inactive and does not proceed with further scanning.

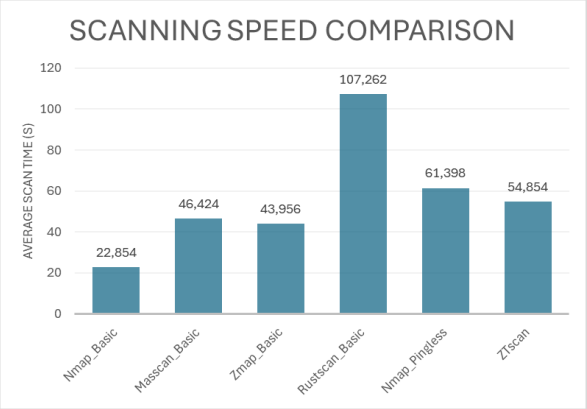


Source: (Research Results, 2025)

Figure 6. Accuracy Performance Comparison

Masscan_Basic (77.50%) and ZMap_Basic (73.00%) exhibit lower accuracy compared to Nmap. This is because both tools are designed for high-speed scanning and do not perform in-depth port probing for each discovered host. On the other hand, ZTscan and Nmap_Pingsyn achieved 100.00% accuracy, successfully identifying all hosts and open ports in the test scenario. This indicates that TCP SYN ping (used in Nmap_Pingsyn) and ZTscan's approach effectively detect hosts that block ICMP by scanning directly on target TCP ports.

These results suggest that ZTscan and Nmap_Pingsyn are the most reliable choices for accurate network scanning, particularly in complex network environments where ICMP-based discovery may fail.The research results section contains exposure to the results of the analysis related to the research question.



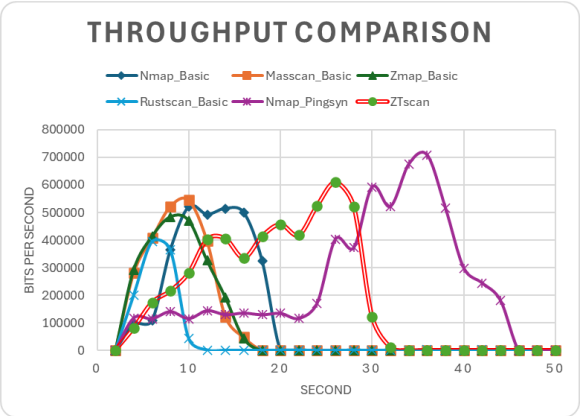
Source: (Research Results, 2025)

Figure 7. Scanning Speed Comparison

The scan speed comparison is illustrated in Figure 7. Nmap_Basic, with a completion time of 22.854 seconds, is the fastest scanning method in this test. This efficiency is due to its simple, direct approach, which only scans known targets without additional verification steps. Masscan_Basic (46.424 seconds) and ZMap_Basic (43.956 seconds), despite being known for their high-speed scanning capabilities, performed slower than Nmap_Basic in this test. This is likely due to the methodology used, where both tools still needed to handle a broader subnet range, increasing processing time.

RustScan_Basic recorded the longest scanning time at 107.262 seconds. While RustScan is typically recognized for its speed, the delay in this scenario stems from its process of verifying each detected host using Nmap, adding an extra layer of analysis. Nmap_Pingsyn (61.398 seconds) and ZTscan (54.854 seconds), while slower than Nmap_Basic, delivered more accurate results, particularly in complex network environments where basic ICMP-based detection might fail.

From these results, Nmap_Basic is the fastest tool, but its lower accuracy makes it less suitable for comprehensive network discovery. ZTscan and Nmap_Pingsyn provide a balance between speed and accuracy, making them more reliable choices for scanning scenarios that require thorough asset detection.



Source: (Research Results, 2025)

Figure 8. Throughput Comparison

The comparison of throughput generated by each scanning tool is shown in Figure 8. Throughput, measured in bits per second (bps), reflects the amount of data transmitted during the scanning process. A higher throughput indicates greater network traffic generated by the tool.

At the start of the scan, all tools exhibit a sharp increase in throughput. Masscan_Basic and ZMap_Basic record the highest peaks, reaching 544,550.4 bps and 471,158.4 bps at the 10-second mark, respectively. This result aligns with their design as ultra-fast asynchronous scanners that send a large volume of packets in a short time. ZTscan demonstrates a more moderate throughput increase but still surpasses Nmap_Basic and RustScan. Nmap_Pingsyn, while generating lower throughput compared to high-speed scanners, maintains a steady traffic rate throughout the scan.

As the scan progresses, Masscan and ZMap experience a gradual decline in throughput after their initial spike. This happens because most of their packets do not receive further responses after the initial sweep. RustScan consistently records low throughput, indicating a more conservative scanning approach compared to the others. Meanwhile, ZTscan gains stability and reaches 610,081.6 bps at the 26-second mark, highlighting its efficiency in performing thorough scans without excessive traffic spike.

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 4 MAY 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i4.6628

In the later stage, Nmap_Pingsyn experiences a sudden throughput spike at the 38-second mark, peaking at 515,736.8 bps. This suggests that its TCP SYN ping method continues working even after other tools have completed their scans. Meanwhile, ZTscan maintains relatively stable throughput compared to Nmap_Pingsyn, with fewer sharp fluctuations. Masscan, ZMap, and RustScan exhibit a drastic drop in throughput after 30 seconds, confirming their suitability for rapid scanning rather than deep network analysis.

ZTscan exhibits greater stability in traffic distribution compared to Nmap_Pingsyn, which experiences a late-stage spike. The smoother traffic pattern of ZTscan reduces the risk of sudden network disruptions, which could impact active network operations. This aligns with ZTA requirements, where accuracy in asset identification must be balanced with minimal disruption to live network environments.

CONCLUSION

This research introduces ZTscan, a method that integrates Masscan and Nmap to optimize resource discovery within Zero Trust Architecture (ZTA). Masscan's speed in detecting active hosts and open ports, combined with Nmap's precision in analyzing services and operating systems, allows ZTscan to strike an optimal balance between speed, accuracy, and scanning traffic stability. Testing reveals that ZTscan achieves 100% accuracy, placing it on par with Nmap_Pingsyn while surpassing faster scanning tools like Masscan, ZMap, and RustScan in precision. In terms of speed, ZTscan completes scans in 54.854 seconds, faster than Nmap_Pingless (61.398 seconds) while maintaining comparable high accuracy. While Nmap_Basic completes the fastest at 22.854 seconds, it does so with much lower detection reliability. In terms of throughput, ZTscan demonstrates a stable and moderate traffic pattern, peaking around 610,081.6 bps without causing disruptive spikes. Unlike Masscan_Basic and ZMap_Basic, which exhibit early traffic surges, and Nmap_Pingsyn, which shows late-stage throughput instability, ZTscan maintains a consistent transmission rate throughout most of the scanning process. This stable behavior minimizes network disruption risks, aligning with Zero Trust Architecture (ZTA) requirements for secure and reliable resource discovery.

Findings from this study position ZTscan as an optimal resource discovery solution for ZTA, seamlessly blending Masscan's rapid scanning with Nmap's in-depth analysis while mitigating the network instability often caused by high-intensity scans. However, this study also has several limitations. Testing was conducted in a controlled and relatively small-scale network environment, which may not fully represent performance under large-scale or highly heterogeneous enterprise networks. Furthermore, ZTscan currently relies on a fixed scanning strategy and does not yet adapt its behavior dynamically based on network conditions or observed responses. For future research, it is recommended to: Test ZTscan in larger and more diverse network topologies, including multi-subnet, VPN, and cloud-based environments, Integrate AI/ML-based host profiling to enhance automation in interpreting scan results and prioritizing assets and Explore the development of adaptive scanning strategies that dynamically adjust scan speed, intensity, and validation techniques based on real-time feedback from the network. By addressing these areas, ZTscan can continue evolving into a smarter, more scalable, and more adaptive solution, capable of supporting robust Zero Trust initiatives across a wider range of network conditions and organizational needs.

ACKNOWLEDEMENT

This research is funded by the Scholarship Program of the Ministry of Communication and Digital Affairs.

REFERENCE

[1] Direktorat Operasi Keamanan Siber BSSN, 'Lanskap Keamanan Siber Indonesia 2023', Jakarta, 2024.

[2] N. Basta, M. Ikram, M. A. Kaafar, and A. Walker, 'Towards a Zero-Trust Micro-segmentation Network Security Strategy: An Evaluation Framework', in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium 2022: Network and Service Management in the Era of Cloudification, Softwarization and Artificial Intelligence, NOMS 2022*, Institute of Electrical and Electronics Engineers Inc., 2022. doi: 10.1109/NOMS54207.2022.9789888.

[3] Z. Adahman, A. W. Malik, and Z. Anwar, 'An analysis of zero-trust architecture and its cost-effectiveness for organizational security', *Comput Secur*, vol. 122, Nov. 2022, doi: 10.1016/j.cose.2022.102911.

[4] Z. P. Putra, R. Harwahyu, and E. Hebert, 'Performance Evaluation Elastic Security as Open Source Endpoint Detection and

Response for Advanced Persistent Threat Cyberattack', *International Journal of Electrical, Computer, and Biomedical Engineering*, vol. 2, no. 2, Jun. 2024, doi: 10.62146/ijecbe.v2i2.49.

[5] R. Harwahyu, F. H. E. Ndolu, and M. V. Overbeek, 'Three layer hybrid learning to improve intrusion detection system performance', *International Journal of Electrical and Computer Engineering*, vol. 14, no. 2, pp. 1691–1699, 2024, doi: 10.11591/ijece.v14i2.pp1691-1699.

[6] S. Lee, J.-H. Huh, and H. Woo, 'Security System Design and Verification for Zero Trust Architecture', *Electronics (Basel)*, vol. 14, no. 4, p. 643, Feb. 2025, doi: 10.3390/electronics14040643.

[7] Y. Cao, S. R. Pokhrel, Y. Zhu, R. Doss, and G. Li, 'Automation and Orchestration of Zero Trust Architecture: Potential Solutions and Challenges', Apr. 01, 2024, *Chinese Academy of Sciences*. doi: 10.1007/s11633-023-1456-2.

[8] N. Nahar, K. Andersson, O. Schelen, and S. Saguna, 'A Survey on Zero Trust Architecture: Applications and Challenges of 6G Networks', *IEEE Access*, vol. 12, pp. 94753–94764, 2024, doi: 10.1109/ACCESS.2024.3425350.

[9] P. Dhiman *et al.*, 'A Review and Comparative Analysis of Relevant Approaches of Zero Trust Network Model', Feb. 01, 2024, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/s24041328.

[10] W. Yeoh, M. Liu, M. Shore, and F. Jiang, 'Zero trust cybersecurity: Critical success factors and A maturity assessment framework', *Comput Secur*, vol. 133, Oct. 2023, doi: 10.1016/j.cose.2023.103412.

[11] M. Medhat, S. G. Sayed, S. M. Abd-Alhalem, and A. E. Takieldeen, 'Whitelisting Requirements for Effective Cyber Defense Solutions', in *2023 International Telecommunications Conference, ITC-Egypt 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 484–489. doi: 10.1109/ITC-Egypt58155.2023.10206403.

[12] S. Gupta Bhol, J. R. Mohanty, and P. Kumar Pattnaik, 'Taxonomy of cyber security metrics to measure strength of cyber security', *Mater Today Proc*, vol. 80, pp. 2274–2279, Jan. 2023, doi: 10.1016/j.matpr.2021.06.228.

[13] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, 'Zero Trust Architecture', Gaithersburg, MD, Aug. 2020. doi: 10.6028/NIST.SP.800-207.

[14] P. Phiayura and S. Teerakanok, 'A Comprehensive Framework for Migrating to Zero Trust Architecture', *IEEE Access*, vol. 11, pp. 19487–19511, 2023, doi: 10.1109/ACCESS.2023.3248622.

[15] Z. Zhang, D. Towey, Z. Ying, Y. Zhang, and Z. Q. Zhou, 'MT4NS: Metamorphic Testing for Network Scanning', in *Proceedings - 2021 IEEE/ACM 6th International Workshop on Metamorphic Testing, MET 2021*, Institute of Electrical and Electronics Engineers Inc., Jun. 2021, pp. 17–23. doi: 10.1109/MET52542.2021.00010.

[16] C. Itodo and M. Ozer, 'Multivocal literature review on zero-trust security implementation', *Comput Secur*, vol. 141, p. 103827, Jun. 2024, doi: 10.1016/J.COSE.2024.103827.

[17] T. Kasama, Y. Endo, M. Kubo, and D. Inoue, 'Please Stop Knocking on My Door: An Empirical Study on Opt-out of Internet-wide Scanning', *IEEE Access*, 2025, doi: 10.1109/ACCESS.2025.3551691.

[18] F. Mohammed, N. A. A. Rahman, Y. Yusof, and J. Juremi, 'Automated Nmap Toolkit', in *ASSIC 2022 - Proceedings: International Conference on Advancements in Smart, Secure and Intelligent Computing*, Institute of Electrical and Electronics Engineers Inc., 2022. doi: 10.1109/ASSIC55218.2022.10088375.

[19] J. M. Redondo and D. Cuesta, 'Towards improving productivity in NMAP security audits', *Journal of Web Engineering*, vol. 18, no. 7, pp. 539–578, 2019, doi: 10.13052/jwe1540-9589.1871.

[20] J. Asokan, A. Kaleel Rahuman, B. Suganthi, S. Fairooz, M. Sundar Prakash Balaji, and V. Elamaran, 'A Case Study Using Companies to Examine the Nmap Tool's Applicability for Network Security Assessment', in *12th IEEE International Conference on Advanced Computing, ICoAC 2023*, Institute of Electrical and Electronics Engineers Inc., 2023. doi: 10.1109/ICoAC59537.2023.10249544.

[21] J. M. Pittman, 'A Comparative Analysis of Port Scanning Tool Efficacy', Mar. 2023, [Online]. Available: http://arxiv.org/abs/2303.11282

[22] M. El-Hajj, 'Leveraging Digital Twins and Intrusion Detection Systems for Enhanced Security in IoT-Based Smart City Infrastructures', *Electronics (Switzerland)*,

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 10. NO. 4 MAY 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v10i4.6628

vol. 13, no. 19, Oct. 2024, doi: 10.3390/electronics13193941.

[23] R. D. Graham, 'MASSCAN: Mass IP port scanner', 2024. [Online]. Available: https://github.com/robertdavidgraham/masscan

[24] G. Fyodor. Lyon, *Nmap network scanning : official Nmap project guide to network discovery and security scanning*. 2009. Accessed: Apr. 29, 2025. [Online]. Available: https://nmap.org/book/toc.html

[25] R. Taupaani, 'ZTScan', Mar. 2025. [Online]. Available: https://github.com/numburanggata/ResourceDiscovery#

[26] J. H. Jafarian, M. Abolfathi, and M. Rahimian, 'Detecting Network Scanning Through Monitoring and Manipulation of DNS Traffic', *IEEE Access*, vol. 11, pp. 20267–20283, 2023, doi: 10.1109/ACCESS.2023.3250106.

[27] R. Aliyev, 'A Comprehensive Spectrum of Open Ports: A Global Internet Wide Analysis', in *12th International Symposium on Digital Forensics and Security, ISDFS 2024*, Institute of Electrical and Electronics Engineers Inc., 2024. doi: 10.1109/ISDFS60797.2024.10526469.

[28] X. Yu, Z. Hu, and Y. Xin, 'A New Approach Customizable Distributed Network Service Discovery System', *Wirel Commun Mob Comput*, vol. 2021, 2021, doi: 10.1155/2021/6627639.

[29] Z. Durumeric, D. Adrian, P. Stephens, E. Wustrow, and J. A. Halderman, 'Ten Years of ZMap', in *Proceedings of the 2024 ACM on Internet Measurement Conference*, New York, NY, USA: ACM, Nov. 2024, pp. 139–148. doi: 10.1145/3646547.3689012.