Vulnerability Assessment & Penetration Test

The Approach and Methodology

By

Zettawise Consulting Pvt Ltd

Company Introduction

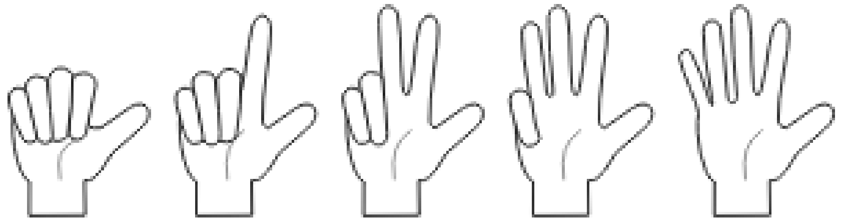
Zettawise Consulting Pvt. Ltd (www.zettawise.in) is an ISO 27001, ISO 9001 and ISO 20000-1 certified technology firm which has established its position prominently in national critical infrastructure protection. It assists governments & large corporate organizations in strengthening their cyber security posture, ensuring compliance with international best practices, improving their cyber attack resilience and enhancing incident handling capabilities.

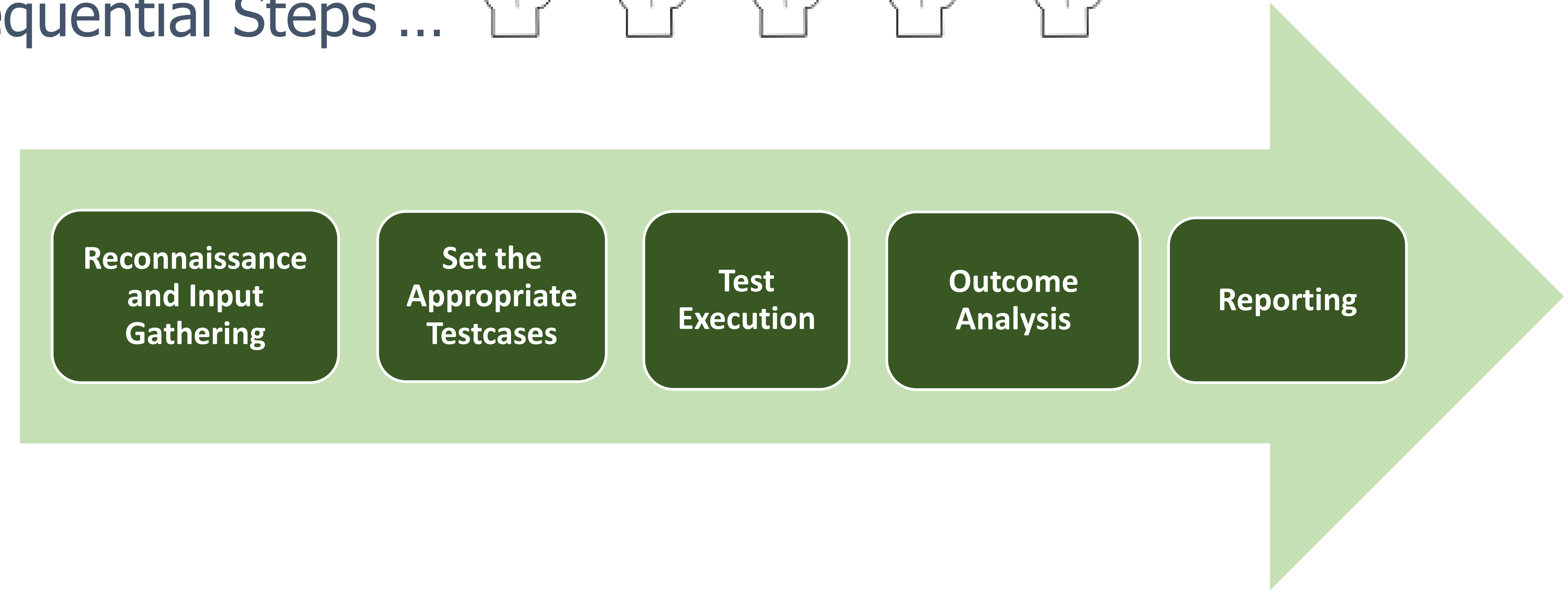
Zettawise's forte is in identification of technical vulnerabilities of different applications, associated APIs and underlying infrastructures through invasive and non-invasive tests by simulating attack scenarios. Our test cases go beyond traditional tool-based assessment to manual validation of business logic exploitation in the correct context.

We are proficient in mitigating cyber risks through high-end and technology driven security solution deployment in collaboration with technology partners like IBM Security and also ensuring higher level of process orientation as a business associate of internationally accredited certification body like BSI.

Zettawise Consulting is a part of the joint workforce of SASTRA (RRU, Ministry of Home Affairs, Govt of India) under the aegis of "AtmaNirbhar and AtmaSurakshit Bharat Mission" of Govt of India.

Five Sequential Steps …




Reconnaissance and Input Gathering
Set the Appropriate Testcases
Test Execution
Outcome Analysis
Reporting

Testing Approach

Whitebox Testing	Greybox Testing	Blackbox Testing
• Credential testing. • Full visibility into the inner workings of the asset. • Sharing full network and system information. • Simulates a targeted attack on a specific system.	• Blackbox testing + Credentialed testing. • Limited information is shared with the tester. • Simulate either an insider threat or an attack that has breached the network perimeter.	• Zero visibility into the asset's functions and workflows. • No knowledge of codebase or infrastructure. • Most authentic as tester demonstrates how an adversary with no inside knowledge would target.

zettawise consulting

Assessment Methodology

Web Application Test

- Reconnaissance
- Threat Modeling
- Scanning and Enumeration
- Exploitation
- Result Analysis
- Reporting

Mobile Application Test (iOS/Android)

- Understanding the Application
- Analysis
- Assessment
- Performing Manual and Automated Tests
- Result Analysis
- Reporting

API Test

- Initiation
- API Analysis
- API Enumeration
- Scope and Roles Testing
- Result Analysis
- Reporting

zettawise consulting

Web Application Test Methodology

Our security testing approach and methodology is based on industry leading practices such as OWASP Testing Guide, PCI Penetration Testing Guide, Penetration Testing Execution Standard, NIST 800-115, Penetration Testing Framework, Information Systems Security Assessment Framework (ISSAF), Open-Source Security Testing Methodology Manual (OSSTMM)

Phase 1

Outline objectives and boundaries.

Record conditions and prerequisites for testing.

Create a structured timetable for testing activities.

Gain insights into the application's features.

Analyze data flow for security assessment.

Phase 2

Scrutinize the application's logic for vulnerabilities.

Verify and assess user access authorization controls.

Plan and execute scans with a combination of manual and automated tools.

Phase 3

Execute dynamic testing methods.

Assess the application for vulnerabilities related to data manipulation.

Identify and evaluate the application for known Common Vulnerabilities and Exposures (CVEs).

Explore and address attack vectors and payloads specific to the application's technology.

Validate identified vulnerabilities and eliminate false positive results.

Document and organize a comprehensive list of identified vulnerabilities.

Gather supporting evidence and create video Proof of Concepts (POCs) for better understanding.

Phase 4

Assess the level of difficulty associated with exploiting identified vulnerabilities.

Offer specific technical solutions or recommendations to address identified vulnerabilities.

Conduct an independent quality review of assessment findings before final report submission.

zettawise consulting

Common vulnerabilities Use Cases –Web Application

Injection:

SQL, NoSQL, OS, and LDAP injection attacks.

Broken Authentication and Session Management:

Inadequate implementation of authentication mechanisms and session management.

Sensitive Data Exposure:

Exposure of sensitive information such as passwords or credit card numbers.

XML External Entities (XXE):

Exploiting vulnerable XML processors to disclose internal files, retrieve data, or execute remote code.

Broken Access Control:

Inadequate enforcement of access controls, allowing unauthorized access to data or functionality.

Security Misconfiguration:
Poorly configured security settings that may lead to exploitation.

Cross-Site Scripting (XSS):
Injection of malicious scripts into web pages viewed by other users.

Insecure Deserialization:
Exploiting vulnerabilities in the deserialization of untrusted data.

Using Components with Known Vulnerabilities:
Use of outdated or vulnerable third-party components.

Insufficient Logging and Monitoring:
Lack of proper logging and monitoring, making it difficult to detect and respond to security incidents.

zettawise consulting

API Testing Methodology

Phase 1

Evaluate and scrutinize the API endpoints.

Identify and assess the authentication mechanisms in place, including Basic HTTP authentication.

Review and test the application's input validation procedures to identify potential vulnerabilities.

Examine the use and security of access tokens within the API.

Assess the implementation and security of cookies used in the API.

Create a structured timetable outlining the sequence and timing of testing activities.

Establish the necessary environment and ensure testing tools are ready for use in API assessments.

Phase 2

Verify that all API endpoints are adequately protected behind authentication mechanisms to prevent broken authentication processes.

Conduct input fuzzing tests to identify potential vulnerabilities and weaknesses in the API input handling.

Assess the API for security by testing for unhandled or unexpected HTTP methods that could pose a risk.

Scrutinize API requests and responses to identify any anomalies, security gaps, or potential vulnerabilities.

Phase 3

Identify and assess the application for vulnerabilities related to unauthorized access.

Assess the application to identify and address potential vulnerabilities leading to data leakage.

Test for vulnerabilities associated with fuzzy input and injection attacks.

Investigate potential vulnerabilities related to parameter tampering.

Conduct testing of data validation processes to identify and address weaknesses.

Evaluate the application's access permissions to ensure proper and secure access control.

Test for vulnerabilities related to insecure direct object references to prevent unauthorized access to sensitive resources.

Phase 4

Assess the level of difficulty associated with exploiting identified vulnerabilities.

Offer specific technical solutions or recommendations to address identified vulnerabilities.

Conduct an independent quality review of assessment findings before final report submission.

*Our security testing approach and methodology is based on industry leading practices such as OWASP Testing Guide, PCI Penetration Testing Guide, Penetration Testing Execution Standard, NIST 800-115, Penetration Testing Framework, Information Systems Security Assessment Framework (ISSAF), Open-Source Security Testing Methodology Manual (OSSTMM)

zettawise consulting

Common Vulnerabilities Use Cases-API

Broken Object Level Authorization
APIs often expose object identifiers, creating a significant attack surface for Object Level Access Control issues. Object-level authorization checks are crucial in functions accessing data using user-provided IDs.

Broken Authentication
Incorrect authentication implementation allows attackers to compromise tokens or assume other user identities, compromising the API's overall security.

Broken Object Property Level Authorization
Addresses the root cause of API3:2019 and API6:2019, focusing on the lack of authorization validation at the object property level, leading to information exposure or manipulation.

Unrestricted Resource Consumption
Successful attacks on required resources may lead to Denial of Service or increased operational costs for services such as emails/SMS/phone calls or biometric validations.

Broken Function Level Authorization
Complex access control policies can result in authorization flaws, allowing attackers access to other users' resources or administrative functions.

Unrestricted Access to Sensitive Business Flows
Vulnerable APIs expose business flows without compensating for potential harm caused by excessive automated usage.

Server-Side Request Forgery
SSRF flaws can occur when an API fetches a remote resource without validating the user-supplied URI, enabling attackers to coerce the application into sending crafted requests.

Security Misconfiguration
APIs and supporting systems may have complex configurations, and misconfigurations can expose vulnerabilities due to oversight or non-adherence to security best practices.

Improper Inventory Management
APIs expose numerous endpoints, making updated documentation and a proper inventory of hosts and deployed API versions crucial to mitigate issues.

Mobile App Testing Methodology

Phase 1

Set up and install the APK file on Android security testing devices for assessment.

Conduct reconnaissance to gather information and perform threat modeling to identify potential security threats.

Ensure that all components of the app are identified and documented for assessment.

Create a structured timetable outlining the sequence and timing of testing activities.

Phase 2

Set up proxy interception to inspect the flow of data between the app and external sources.

Gain insights into the core business functionality of the app to pinpoint areas where data enters and exits.

Determine where the application stores data, whether at rest, in transit, or on display, and assess the sensitivity of this information.

Phase 3

Formulate test cases to assess the security of data storage, focusing on privacy concerns and potential vulnerabilities.

Develop test cases to evaluate the effectiveness of cryptographic mechanisms implemented in the application.

Formulate test cases to verify the robustness of authentication processes and the security of session management.

Assess the security of network communications by formulating test cases that evaluate the encryption methods used.

Develop test cases to evaluate how the application interacts with the underlying platform, ensuring security measures are maintained.

Phase 4

Assess the level of difficulty associated with exploiting identified vulnerabilities.

Offer specific technical solutions or recommendations to address identified vulnerabilities.

Conduct an independent quality review of assessment findings before final report submission.

Ref: Our security testing approach and methodology is based on industry leading practices such as OWASP Testing Guide, PCI Penetration Testing Guide, Penetration Testing Execution Standard, NIST 800-115, Penetration Testing Framework, Information Systems Security Assessment Framework (ISSAF), Open-Source Security Testing Methodology Manual (OSSTMM)

Common Vulnerabilities Use Cases- Mobile App

Improper Credential Usage
Refers to issues related to how credentials, such as usernames and passwords, are handled and stored within mobile applications.

Inadequate Supply Chain Security
Focuses on the security of the entire supply chain involved in the development and distribution of mobile applications.

Insecure Authentication/Authorization
Concerns weaknesses in the processes of authenticating and authorizing users within mobile apps, potentially leading to unauthorized access.

Insufficient Input/Output Validation
Highlight vulnerabilities arising from inadequate validation of data inputs and outputs in mobile applications, which may lead to security issues.

Insecure Communication
Relates to the security risks associated with insecure transmission of data between mobile apps and backend servers.

Inadequate Privacy Controls
Addresses concerns about the inadequate implementation of privacy controls in mobile applications, leading to potential privacy breaches.

Insufficient Binary Protections
Refers to the lack of proper protection mechanisms for the binary code of mobile applications, making them susceptible to reverse engineering.

Security Misconfiguration
Involves security issues arising from misconfigurations in mobile applications, which may expose sensitive information or introduce vulnerabilities.

Insecure Data Storage
Focuses on vulnerabilities related to the insecure storage of sensitive data within mobile devices or external storage.

Insufficient Cryptography
Highlights concerns about the improper use or inadequate implementation of cryptographic techniques in mobile applications, potentially leading to security vulnerabilities.

Penetration Testing Methodology for Infrastructure:

Phase 1

Outline objectives and boundaries.

Record conditions and prerequisites for testing.

Create a structured timetable for testing activities.

Gain insights into the network architecture and topology.

Analyze network data flow and critical points for security assessment.

Phase 2

Identify and map all network devices, including routers, switches, firewalls, and servers.

Gather network policies and access control lists (ACLs).

Enumerate services and open ports on target devices.

Collect information about network protocols in use.

Phase 3

Scrutinize the network's configuration and security controls for vulnerabilities.

Verify and assess user access authorization controls across network devices.

Identify and evaluate the application for known Common Vulnerabilities and Exposures (CVEs).

Plan and execute scans with a combination of manual and automated tools to identify vulnerabilities.

Evaluate firewall and intrusion detection/prevention system configurations.

Document and organize a comprehensive list of identified vulnerabilities.

Gather supporting evidence and create Proof of Concepts (POCs) for better understanding.

Phase 4

Document and organize a comprehensive list of identified vulnerabilities.

Assess the CVSS Score and the level of difficulty associated with exploiting identified vulnerabilities.

Compile a detailed report outlining the identified vulnerabilities, exploitation techniques, evidence and remediation recommendations.

*Our security testing approach and methodology is based on industry leading practices such as OWASP Testing Guide, PCI Penetration Testing Guide, Penetration Testing Execution Standard, NIST 800-115, Penetration Testing Framework, Information Systems Security Assessment Framework (ISSAF), Opensource Security Testing Methodology Manual (OSSTMM)

Common vulnerabilities for Network/Infrastructure:

Unpatched Software and Firmware:
Using outdated software or firmware can expose systems to known vulnerabilities that attackers can exploit.

Insecure Network Protocols:
Utilizing protocols that are outdated or insecure can lead to data interception and unauthorized access. Examples include older versions of SMB, Telnet, or FTP.

Improper Network Segmentation:
Failing to segment networks properly can allow attackers to move laterally within the network which can increase the risk of widespread compromise.

Weak Encryption:
Using weak or outdated encryption methods for data in transit or at rest can make it easier for attackers to decrypt sensitive information.

Denial of Service (DoS) Vulnerabilities:
Network infrastructure may be susceptible to DoS attacks that overwhelm resources and disrupt services.

Security Misconfiguration:
Poorly configured security settings that may lead to exploitation.

Unsecured Network Services:
Running unnecessary or insecure services on network devices can create potential entry points for attackers.

Poorly Managed Network Access Controls:
Inadequate management of network access controls can lead to unauthorized users gaining access to restricted areas of the network.

Using Components with Known Vulnerabilities:
Use of outdated or vulnerable third-party components.

Insufficient Logging and Monitoring:
Lack of proper logging and monitoring, making it difficult to detect and respond to security incidents.

Some common tools being used:

zettawise consulting

NMAP

Metasploit

Kali

Wireshark

Acunetix

ZAP

Burp-Suite

John the Ripper

THC Hydra

FFuf

SQLMAP

Nessus Pro

Testing Standards/Framework Followed

PENETRATION TESTING EXECUTION STANDARD PTES

OWASP API SECURITY TOP 10

MITRE | ATT&CK™

OWASP TOP 10

Open Source Security Testing Methodology Manual OSSTMM www.osstmm.org

NIST 800-53 Revision 5 Security Controls

zettawise consulting

Some of Business Cases:

General Directorate of Residency and Foreign Affairs (GDRFA) has created a smart channel where using Smart Gates, UAE citizens and residents can pass through by simply looking at the green light, with no need to scan any identification document. The security of this system is of paramount importance because this would have a direct impact on national security.

Zettawise did end-to-end security testing for the smart channel. It included about two dozen web apps, a similar number of third-party integrations and APIs and a couple of mobile apps.

Dubai Health Authority (DHA) provides a quality healthcare system in Dubai by setting and ensuring policies and strategies for healthcare in public and private hospitals and clinics in Dubai. DHA has created a unified Medical Fitness System (Salem) which is intended for use by Registered Companies and Typing Centers.

Zettawise is conducting the Vulnerability Assessment and Penetration Testing of the applications and infrastructure associated with the SALEM portal. This includes multiple associated applications, over 30+ user roles and other supporting digital infrastructure.

Noqodi, a fintech company, part of Emaratech, is dedicated to electronic payments through Dubai Pay. It offers customers various services to consumers such as e-wallet, direct payments, eCash and other multiple channels. It also helps merchants accept payment using these channels, online payments, provides point of sales machines and analytics of transactions. It has tied up with over 40+ government and semi-government entities for easier payment using the e-wallet.

Zettawise conducted an assessment on the security of the various applications of Noqodi including customer portal, merchant portals, back office, mobile apps APIs for Noqodi and GV pay, V1 and V2 of the payment APIs. In addition, the security of the infrastructure supporting the above was also tested.

Some of Our Prestigious Clients

AMBUJA NEOTIA

ESSEL MINING & INDUSTRIES LTD (ADITYA BIRLA GROUP)

BANDHAN FINANCIAL SERVICES

TCG CREST

POWER SYSTEM OPERATION CORPORATION

MEDICA SUPER SPECIALITY HOSPITAL

SAHAJ

RAKUTEN

ALUMNUS SOFTWARE LTD

PEERLESS HOSPITAL

GKB OPTICALS

PS GROUP

IMS INDIA

SYNERGIC SOFTEK SOLUTIONS PVT. LTD.

WEBEL TECHNOLOGY LIMITED

MACAWS INFOTECH

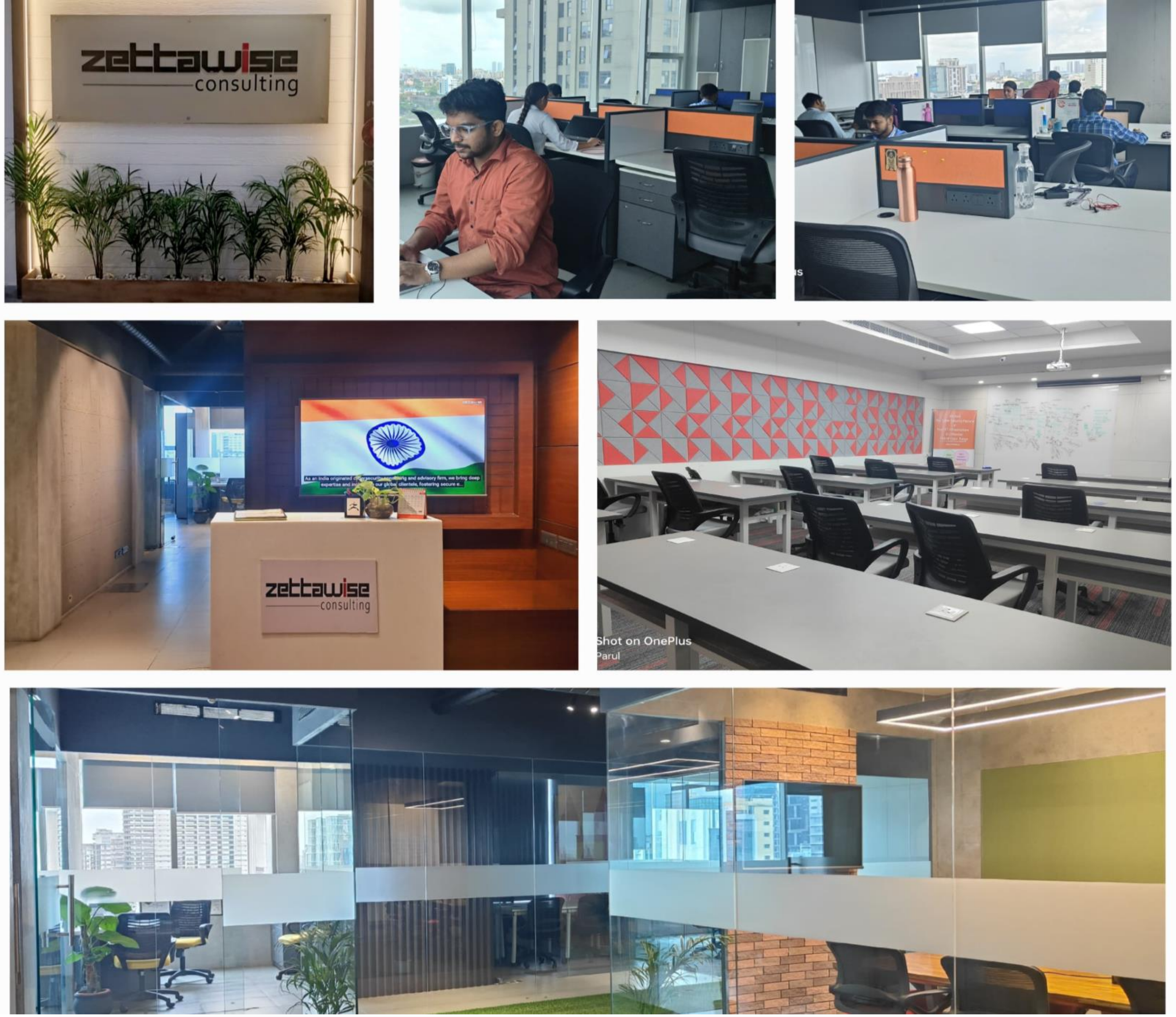
NOVASOL (DUBAI)

KOLKATA POLICE

PRIME INFOSERV

PRICORIS

State of The Art Zettawise Cyber Command Centre



zettawise consulting

To Know More, Contact

Zettawise Consulting Pvt. Ltd.
11th Floor, Godrej Genesis. EP & GP Block
Suit No. 1102. Sector – 5
Salt Lake Electronics Complex.
West Bengal 700091. India.
Call: +91 7980889376 / +91.9830098446
Email: contact@zettawise.in
Web: www.zettawise.in

zettawise consulting