Security Architecture

Last edited by **Shuvam Misra** 1 hour ago

This page will attempt to list a set of actionable items to make the Nimbus system more secure. "More" is relative -- we mean "more secure than if you did not implement these items." The points may not have much relation with each other -- each point stands by itself. This page will not include the basic hygiene rules like

- firewalls,
- IP address based access restrictions,
- password strength enforcement,
- password expiry,

etc.

This page has the status of an advisory, not a management decision. It does not specify what Nimbus will implement or has implemented. It is hoped that Nimbus may implement all the measures listed here.

1 Semi-public access

Sometimes, in B2C systems or Extranets with thousands of external parthers, we need to provide access to our system for users who are not part of our organisation, and will not follow many of the hygiene and good practice rules which we may impose on employees. This section addresses the features and measures to be implemented to tighten security for such access.

1.1 Separate the databases

The screens which external parties access must connect to a separate database, not the primary internal database. The two databases will need to sync some tables, and this syncing must be implemented in such a way that tables in the "semi-public" database should not overwrite data in the "primary" database. If this overwriting is permitted, then the key purpose of database separation may be lost.

The "semi-public" database must be smaller than the "primary" database and must contain only the information needed to allow the external users to perform their tasks.

Database sync can be implemented using database replication (see Postgres FWD) or external layers like Kafka

1.2 Workflow for user account creation

The system must implement a workflow by which a new user account is requested, then sanctioned, then created in the system. In theory, this can be implemented in a paper trail too, but paper tends to get "lost".

This workflow is needed because it is usually seen that after some period of usage, the semi-public system has dozens or hundreds of accounts which had been created in good faith but whose antecedents are unknown today. You will not know who asked for the account to be created, and for what purpose. You will see three accounts with very similar names, to be used apparently by the same individual, but you will never know why three were created when one was needed. And so on. This is a progressive decay of the user database which happens in all systems, and happens much faster in systems where the user base has external users. Therefore, the antecedents of each account must be tracked.

1.3 Temporary accounts

The system must support an expiry date against an account. This permits the admin to create a temporary account which will auto-expire after its expiry date. In the absence of this feature, the system will be littered with accounts like "tstuser1" and "tempadmin", which were created in good faith "last Diwali" or "six months ago when we needed to test XYZ", and are still lying around in the system.

1.4 Auto-disable of idle accounts

The system must disable accounts for which there is no access for X weeks or months. Idle accounts are the most common vector for intrusion, because their usage is not noticed and does not interfere with the activities of other, real, active users.

Email / SMS must go to the account a few times before it is finally disabled. A disabled-accounts report must be implemented in the system, for audit purposes by the admin team. This report must at least list out

- the account name
- who asked for the account to be created
- when was it last used for access
- when was it auto-disabled

2 Access to core systems by employees

This section addresses how to tighten security for employees who need to access core systems (the most important data and screens, including financial data) over the Internet. Here we

- need to make the security very tight, and
- we have a set of users who will follow security processes and protocols more willingly than external users.

2.1 2FA

Two-factor authentication is mandatory for access to core systems from the public Internet. This is non-negotiable.

There are multiple ways to implement 2FA: choose any one:

- OTP by SMS
- An authenticator app like Google Authenticator or its open-source, standards compliant cousin FreeOTP from RedHat
- A USB dongle: the most famous of them being Yubikey from Yubico, widely supported on all sorts of operating systems and programming systems

Passwords today are as good as non-existent. Passwords alone will stop a casual intruder who wants to create mischief, but is completely useless to stop a criminal intruder who is persistent -- he can purchase passwords from the dark web or pay other "service providers" to break into your account using keyloggers etc and then give him your password. Any financial services organisation must anticipate and plan for persistent and criminal intruders. **2FA is mandatory.**

2.2 Endpoint device enabling

See "Endpoint device tracking" under Common measures below. That tracking is necessary for all users.

Specifically for users of core systems, each end-point device must be enabled individually by an administrator or by using 2FA. In other words, if a user starts accessing Nimbus from a new PC or a new mobile device for the first time, then the system must flash a special message to the user on email or SMS, and ask the user to confirm that this device indeed belongs to him/her.

Sometimes a 2FA-based approach to enabling a new device may not work. This sometimes happens when a senior officer is travelling to a location where Internet and GSM access are unreliable and the officer is forced to use a kiosk or cybercafe to access the system. For such emergencies, the system must have a facility to allow an admin to enable the new device for the officer, temporarily. The admin may disable the device after 24 hours.

2.3 Remote IP reputation tracking

Every IP address on the Internet has a reputation which is tracked by security service providers. Any IP address from where spam emails originate or from where credit card fraud is attempted is given a bad reputation. IP addresses of VPNs and web proxy anonymisers are also considered poor reputation origination points, because such services are used to mask the real origination IP address of the user. This reputation information is routinely used by credit card processors to block online transactions which are attempted from such IPs.

Nimbus must subscribe to such IP reputation database services, and block access to the core system from IP addresses with poor reputation.

2.4 Clean up end-user desktops

(Here, "desktops" include "laptops".)

The commonest method used today to gain access to confidential information is by breaking into a legitimate user's account. This is done by using malware to infect the user's endpoint device (his desktop) and either accessing the servers directly from it, or by stealing his credentials using keyloggers and other devices. Therefore the following measures must be enforced for all users of core systems who use company-owned desktops:

- do not give the user admin access to his own desktop. If you do, he will disable or subvert security measures, or malware will gain administrator access to his desktop.
- install patch management tools on each desktop, and have a patch management server (Microsoft SCCM is strongly recommended) to enforce policies
- ensure that malware filters are downloading latest signature databases automatically, and block access to the office network from any desktop whose patches and signature databases are older than a certain limit
- install ad blockers in all browsers. Malware is most commonly being transmitted through online ads these days, and infects the desktop when a careless user clicks on an ad. No "official" desktop needs to serve up ads for the entertainment of its user. (Adblock or one of its derivatives is strongly recommended.)
- procure and install LAN traffic monitoring tools which will monitor every packet and warn when it sees signs that there are infected desktops on the LAN. Products like Nevis Networks LANenforcers are widely used.

- configure the office LAN by partitioning it into separate VLANs, so that broadcasts from desktops should not be able to reach servers or too many other desktops. Broadcast packets are often used by malware to spread from one desktop to another. A large LAN should be partitioned into separate small(er) VLANs, and all servers, firewalls, security devices, etc, should be in a separate VLAN of their own. All modern L3 switches support VLANs with ACL based inter-VLAN traffic control. If such L3 switches are not deployed in the office LAN, at least one redundant pair of such L3 switches should be procured and installed as core switches.

3 Common measures

This section lists security tightening measures which must be implemented for both semi-public access and core system access.

3.1 IP address geo-location control

The system must support IP address geo-location mapping and access control. Basically, the admin must be able to do the following:

- monitor which geographic location each login of each user is coming from, by looking at admin reports
- monitor the typical locations from which a user accesses the system and mark them low-risk
- raise an alert if a user is accessing the system from a location other than the usual locations from where he accesses it
- build a white-list of locations from where users are allowed to access the system, and block access attempts from all other locations. This should be extensible to allow a global white-list, followed by user-specific additional white-lists.

Some of these features are supplied out of the box in Azure AD. See this Microsoft page about the location condition in Azure AD conditional access, and this page for some other interesting services.

3.2 Logging for forensic purposes

All operations performed in the business application layer, plus all accesses at the HTTP layer, must be logged. Every SaaS service claims they log all operations, but very often their logs are useless because they do not address the following features of the logging system:

- all logs must be query-able, *i.e.* they must be in some sort of database which must break each record into fields. And there must be **one** database for all logs from all sources, to make querying effective.
- all logs must have the userID or username as one of the query-able fields, to allow querying only the logs related to one user. This is more difficult to achieve than it seems because a lot of software components which generate log entries do so without associating the entries with a specific user. All operations performed by the system (*e.g.* using `cron` jobs) must log something like "SYSTEM" as the username.
- all logs must have the originating IP address as one of the fields, to allow querying of all activities which originated at one IP address. This is more difficult to achieve than it seems because the originating IP address may need to be passed from outer layer systems to internal systems purely for logging purposes.
- all logs must carry timestamps in UTC, to eliminate the confusion which happens when different systems log in localtime of different timezones, and allow effective time correlation of events at different layers.
- all servers must run network time daemons at all times to ensure that there is no clock drift in any of them.

The strategic importance of effective logging comes from the fact that between 67% and 75% of all security intrusions and incidents are perpetrated by *insiders*. This has been brought out repeatedly in research and survey data. See this Security Intelligence report (2017) and this PwC and DSCI report (2011) as just two examples. Perimeter security fails to diagnose, analyse or prevent such incidents, and logging is the only effective tool.

3.3 `tcpdump` and Wireshark

Install `tcpdump` on all servers and keep them running at a low level of logging detail (*i.e.* don't log every byte of every packet, just some basic information). Rotate the logs every day, and delete the logs older than 10-20 days. This will allow the forensic team to analyse any incidents which happen and go back 10-20 days in the network traffic logs to see any unusual traffic patterns, to better understand how an incident was perpetrated or attempted. It is recommended that these logs be looked at every 1-2 months anyway, just to see if any patterns can be visually identified.

3.4 Endpoint device tracking

The system must log the device(s) being used by each user, and must build a database of all devices which each user uses to access the system. If Nimbus includes mobile apps, then the apps must log some information about the mobile device (OS, make, model, OS version, IMEI number, MCC/MNC). For browser based access, each browser instance must be identified uniquely by Nimbus issuing a permanent cookie to the browser, and checking that cookie each time any access hits the servers.

This way, the system must build a profile of all devices which a user uses to access Nimbus. This information must be captured and remembered. Whenever any user starts accessing the system from a new device the first time, a new-device-added log entry must go into the system logs, so that admins can see a report of new devices being added. Such log entries must also carry geo-location information about the geographic location from where the device connected, plus device information in full.