

Security Measures Applied By ADSelfService Plus

Security Measures Declaration

Document Objective: To showcase all the measures implemented by ADSelfService Plus to ward off security threats. The document also shows how safe a self-service password reset software ADSelfService Plus is. While doing so, the document also sheds light on the best methods to be followed to leverage the efficiency of ADSelfService Plus security features.

The annexure of this document offers you the various methods of deploying ADSelfService Plus. Please refer to it to find out a suitable deployment strategy that will offer maximum security to your organization.

Beneath the description of every security feature, best practices that can be followed to enhance security further have been given. Do not miss them.

Table of Contents:

Document Objective.....	Page 2
Overview of ADSelfService Plus Security Measures.....	Page 4
ADSelfService Plus Working Principle.....	Page 7
* Components that make ADSelfService Plus	
* Working Principle	
General Questions Asked in connection with Working Principle.....	Page 10
ADSSP Functional Flow.....	Page 11
Protocols Used by ADSSP to communicate with Active Directory And Client.....	Page 18
ADSelfService Plus Security Features: Detailed Description.....	Page 19
* ADSelfService Plus' Defense Against SQL Code Injection	
* Controlling end-user behavior to get optimal security answers	
* The Fight against Man-in-the-middle Attacks	
* ADSelfService Plus' reply to passphrase guessing/dictionary attacks/bruteforcing	
Annexure I (Installation Requirements).....	Page 25
Annexure I (Deployment Suggestions).....	Page 26

Overview:

ADSelfService Plus is a meticulously planned self-service password reset software, offering a multitude of security features, which ward off/effectively tackle all possible threats looming over password self-service.

Right from controlling end-user's attitude towards devising security answers to carrying out a safe password reset, ADSelfService Plus takes care of every aspect of password self-service.

You will appreciate the fact that there is a set of features dedicated to every aspect of a self-service password reset cycle: formulation of secondary identification, identity establishment, and the password reset operation itself.

For your convenience, all the security features of ADSelfService Plus and the threats they thwart or issues they solve have been listed below. Before you start reading an in-depth description of these features, you may want to glance through the summary:

Threat	Handlers
<p>User's poor choice of security answers:</p> <p>ADSelfService Plus is an end-user product, where end-users build their secondary authentication information. What if users – inadvertently or deliberately – treat creating security answers frivolously? Therefore it is desirable to regularize their answers; bring in some order.</p>	<p>To make end-users formulate security answers that do not get easily defeated, ADSelfService Plus offers features to:</p> <ol style="list-style-type: none"> 1. Prevent users from using any word of the question in their answers. 2. Prevent users from providing the same answer to multiple questions. 3. Prevent users from reusing a security question. 4. Prevent users from modifying security questions. 5. Force users to keep their answers at lengths specified by administrators. (To ensure passphrases have the necessary length to survive dictionary or brute-force attacks.) 6. Mandatory Security Questions (To force users from answering any question that administrator deems will elicit a quality security answer.)
<p>Easy access to security answers once the database storing them is compromised.</p> <p>Security answers theft during transmission.</p>	<p>7. ADSelfService Plus stores security answers using a one-way hash algorithm. While stored in the database or in transit, they are in an absolutely irretrievable format.</p> <p>Not even those who administrate the product can retrieve a user's security answers! And those who intercept it will only be disappointed!</p>

<p>Man-in-the-middle attacks:</p>	<p>8. Security answers are stored using one-way hash algorithm.</p> <p>9. Ability to enable HTTP over SSL to ensure extra safe communication between users (web browser) and ADSelfService Plus.</p> <p>10. Ability to enable LDAP over SSL to ensure extra safe communication between Active Directory and ADSelfService Plus.</p>
<p>Bot-based attacks/Automated attacks:</p>	<p>11. CAPTCHA to avoid malicious bots from carrying out denial-of-service attacks.</p>
<p>Password bruteforcing and guessing:</p> <p>Bruteforce is a technique where a hacker tries out all possible combinations of a password and feeds them into the system to gain entry. Similarly, a hacker can also try out his guesses to gain entry.</p>	<p>12. Block User</p> <ul style="list-style-type: none"> • ADSelfService Plus allows administrators to define a threshold for number of unsuccessful attempts, breaching which the user account from which such attempts were made would be locked out. • Administrators can also determine the time a user remains locked out upon failing security question authentication. • CAPTCHA also plays a vital role in keep dictionary attacks at bay.
<p>Password guessing</p>	<p>13. Time limit for answering security questions: To avoid giving the hackers sufficient time to guess out the answer to a security question, ADSelfService Plus allows administrator to set time limit for answering a security question.</p>
<p>SQL Code Injection:</p> <p>With the use of specially constructed query strings, which exploit the vulnerabilities in SQL, hackers can fool software applications to expose their database, or even harm them.</p>	<p>14. ADSelfService Plus uses a special SQL handling framework, exclusively designed to fight back such intrusions.</p> <p>15. Fight without fighting!</p> <ul style="list-style-type: none"> • For the most part, ADSelfService Plus bypasses fetching data from its database. <p>(For more on this, check out Query Handler described in the latter part of this document.)</p>

<p>Exploitation of inactive user accounts:</p> <p>The most dangerous attacks come from within the organization. Disgruntled employees, who still might have access to their accounts at the time of resigning, can use it for wrong reasons.</p>	<p>16. Restrict Inactive Users</p> <ul style="list-style-type: none"> This feature identifies and blacklists all the inactive, abandoned, & disabled user accounts in Active Directory, denying them access to ADSelfService Plus, lest someone should reset their passwords by exploiting them.

To add an extra measure of safety, ADSelfService Plus offers administrators the ability to:

<ul style="list-style-type: none"> Enforce password reset history settings during password reset! This is a feature NOT available even in ACTIVE DIRECTORY.
<ul style="list-style-type: none"> Edit security questions provided by ADSSP and create new ones.
<ul style="list-style-type: none"> Display security questions one after another. (If all the questions are shown at once, hackers can note them down and go about researching their answers.)
<ul style="list-style-type: none"> Modify error messages: Sometimes showing more detailed error messages can give away vital information. So, administrators can decide how their error messages should be.
<ul style="list-style-type: none"> Simple and self-sustainable ADSelfService Plus is only 35 MB in size! And it has an inbuilt Tomcat, inbuilt database (MySQL), a database management framework, and an anti-SQL code injection framework. This makes ADSelfService Plus highly self-sustaining; install on a simple and well-connected computer with access to a domain controller, it turns into a server offering password self-service. <p>All this self-sustainability and simplicity makes ADSelfService Plus extremely compatible with any kind of deployment. (Check deployment suggestions in the Annexure.)</p>

ADSelfService Plus also offers a “**Password Meter**” (password strength analyzer) and **displays the domain’s password policy** to guide users to choose the right kind of password.

ADSelfService Plus and its working principle:

Understanding the operation of this self-service password reset software, its design, interactions with Active Directory/client will put you in a better position to recognize its safety and appreciate its efficiency.

Components that make ADSelfService Plus:

Comprising a Tomcat, MySQL database, a framework to handle the database, and core self-service functionality (Java based), ADSelfService Plus is a ready-to-deploy self-service password reset software. It is absolutely self-sustainable and a complete product, as there is no need for the customer to supply a server or database for its functioning: install it on any computer and that computer becomes a server, a database/DMS, all ready to offer password self-service!

The reason for keeping ADSelfService Plus simple and self-sustainable is not just to offer ease of deployment, but more, which you will learn in the “Deployment Suggestions” section of this document.

Working principle: The Basics

In a nutshell:

ADSelfService Plus adopts a three-pronged approach to facilitate a password reset:

- First, it enrolls and formulates identification criteria for end-users. (**Enrollment Process**).
- Second, it uses these criteria to establish end-user identity when they request for password reset. (**Verification Process**)
- Third, it calls for the service of a Windows API to change the password for the end-user. (**Password Reset Process**)

Simple it may look when given in a nutshell and as bulleted points, but it is indeed a complex process, each and every step of which has been carefully provided with necessary security features.

First take a look at the underlying principle:

- a. How the product identifies people who have forgotten their primary identification – passwords.
- b. What type of secondary authentication it uses.
- c. How it resets passwords.

Identity Management through Secondary Identification:

As a software solution that offers password self-service to end-users who forget their passwords (primary identification) and get stranded without log-in, the primary function of ADSelfService Plus is identity management. For this ADSelfService (referred to as ADSSP hereafter) creates secondary identification for all its users by employing “challenge question and response” system of fallback authentication.

In this kind of system, end-users register certain questions that they should be asked when they seek password restoration; correct entries, as given during the time of registration will allow them to restore passwords, while wrong ones result in the denial of entry.

Applying the same principle, ADSSP builds alternate identification criteria for its users. It requires them to enroll by answering challenge questions, either created by themselves or administrators. These answers are stored in an **ENCRYPTED FORMAT (USING ONE-WAY HASH ALGORITHM)** in the product’s intrinsic database, serving as the

users' identity. Whenever end-users seek password self-service, their identity is established by the answers they provide to the challenge questions.

[Challenge Question is a question which elicits a personalized response from users, such as “As a child, what did you wished to be when you grew up?”.]

Once identity established, they are allowed to reset their forgotten passwords or unlock account. And for password reset, ADSSP makes use of the same Windows API that a password reset operation from ADUC would call for. Therefore there is no fear of password being stored or cached or stolen during its transit. It all comes with the same guarantee of safety as in a normal ADUC based password reset.

NOTE: Besides password self-service, ADSelfService Plus provides a people search engine and also empowers end-users with an ability to update Active Directory with their contact information without IT helpdesk's aid. Both these services do not require end-users to enroll with the product.

FAQs in regard to the ADSelfService Plus password reset mechanism:

Where are these secondary identification details stored?

All the alternate identification details are safely stored away in ADSSP's intrinsic database.

Does it mean anyone who can access the database can read those security answers, and what about the man-in-the-middle attacks?

For extra measure of security, security answers are stored in an irretrievably encrypted format using a one-way hashing algorithm. Therefore there is no way for even an IT administrator to find out what an end-user's security answer is! And even in transit, the answers are in hash form, so it is of no use to hackers who intercept them.

My organization rules require that all the databases exist in one dedicated server room. Can I separate ADSSP from its database? Will it affect performance or safety?

You may very well do so. Separating ADSSP from its inbuilt database will not affect performance. It is perfectly safe.

Does the ADSSP password method compromise security?

To reset password, ADSelfService Plus uses the same Windows API used by ADUC to reset a user's password. This, coupled with the fact that ADSSP does not store passwords anywhere, makes it a foolproof password reset mechanism.

Is the password stored anywhere in the product or its database?

NO!

ADSelfService Plus does not store the passwords chosen by the users in its database or anywhere within. Once the identity of the user is established, it immediately calls for the service of a Windows API to reset the password. This function deposits the end-user entered password directly into Active Directory.

In fact, once it completes its identity management task, ADSelfService rather serves as a safe conduit between the end-user and Active Directory. It never really stores any password.

Now that you know the basic principle of ADSelfService Plus, it is time to learn how it goes about resetting a password.

- a. What exactly happens during a password reset operation?
- b. The behind-the-scenes of password reset operation.
- c. How ADSSP interacts with Active Directory and end-user?

All this puts you in a position to appreciate ADSSP's prudence in providing security.

Functional Flow:

ADSelfService Plus is a carefully designed software comprising a complex network of APIs, each having its own agenda. Directing and controlling all of them is ADSSP Password Reset Module (APRM), a master process supervising module, which gets initiated as soon as a user interacts with ADSSP by entering his or her username and domain. This APRM is responsible for orchestrating the password reset, account lock, AD update, and everything that ADSSP can do.

A password reset – though appears to happen in a few minutes – is a complex process, with a series of checks between user-submitted data and the reset password, to ensure high-level security.

First get to know all the functional units, which are responsible for a password reset. You already know the **APRM**. Other functional units are given below, listed in the order they get executed:

Query Handling Framework: This is ADSelfService Plus' database query framework. Anytime a functional subunit of ADSelfService Plus should fetch any data from the database or a user-submitted data needs to be checked against the product database, this framework is called for. It is capable of ascertaining the innocuousness of the user input and detecting any SQL code anomalies, which might fool SQL to act favorably to hackers. (More details about this on Page 19)

Domain Affiliation Checker: Once a user enters his username and domain, this API cross-checks user's affiliation to the stated domain. In other words, it ensures anyone unrelated to domain or organization does not use this application.

Policy Settings Assessor: Checks the policy that applies to the user. Though a user might be a legitimate member of a domain, enrolled for password self-service, he can use the product only if administrator has defined a policy for him and given a go-ahead to use the self-service system.

What is a policy?

In ADSSP terms, a policy is:

Self-Service Options that Users can enjoy + Security Question Configuration + Organization Units (OU) to which this setting applies.

In a nutshell, it is a rule which states what self-service capability of ADSSP a user can avail himself of and the security questions he has to answer during enrollment.

Enrollment Status Checker: Only if the user has enrolled and formulated his alternative identification criteria, ADSelfService Plus would be able to help him self-service password. This API checks a user's enrollment status.

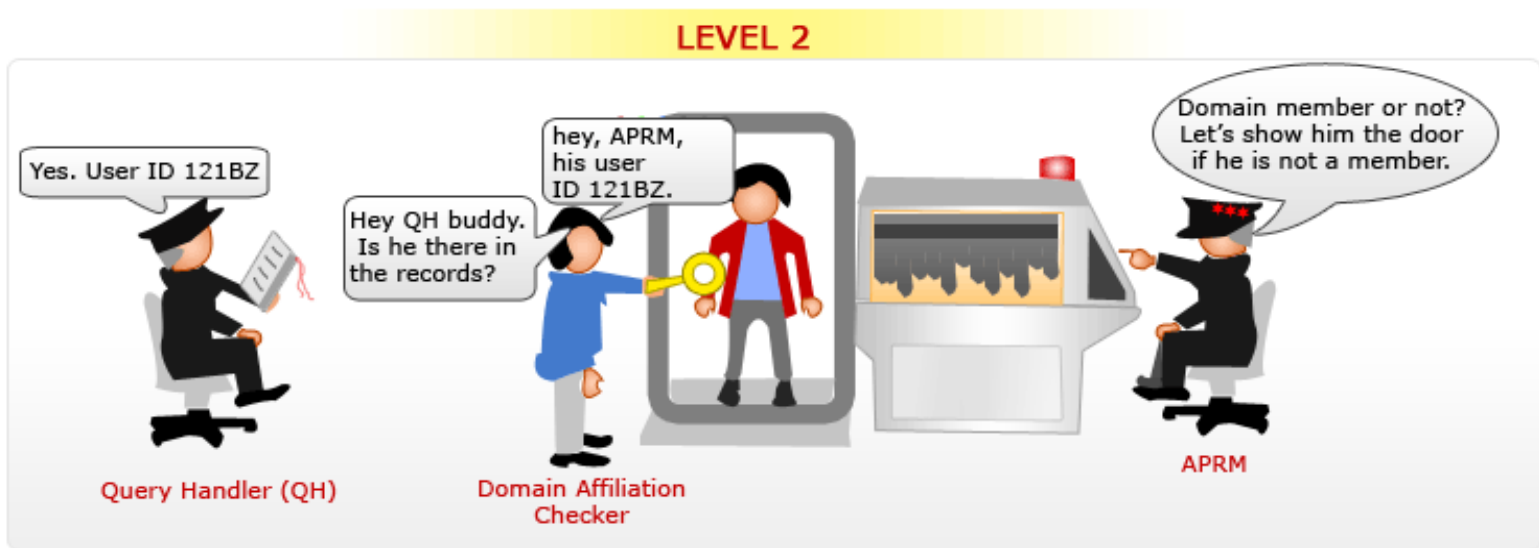
Following is the “behind the scenes” of a password reset:

User enters username and domain name.



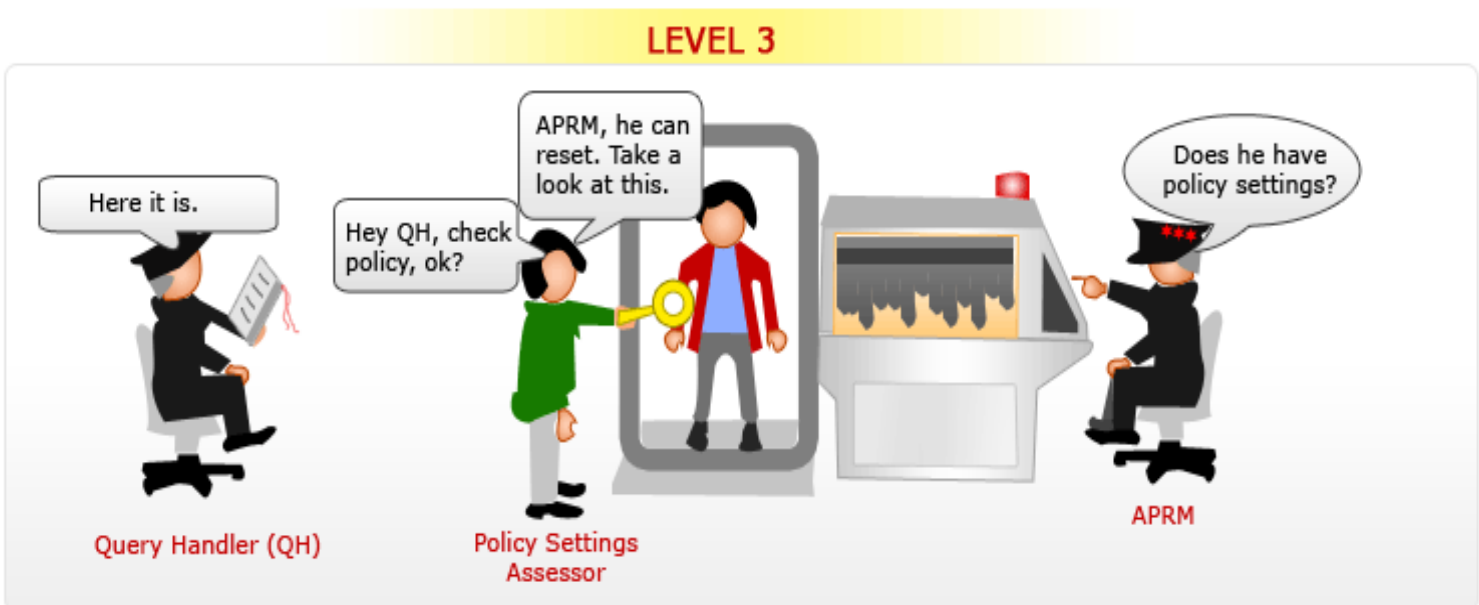
User's Domain Legitimacy Check: APRM is invoked. After initial sanity checks and client-side validation, it calls for Domain Affiliation Checker and submits the username to see if the user is really who he claims to be.

Domain Affiliation Checker uses Query Handler and runs a check against domain users list in the product's database. If the user is really a member of the domain, his userID is returned; else, APRM is informed that he is not a member, which in turn throws appropriate error message and prevents log in. (**ADSSP assigns a unique ID to all users in its database. You will know the role it plays in keeping SQL code injection at bay, when you read all about Query Handler in the next section.**)

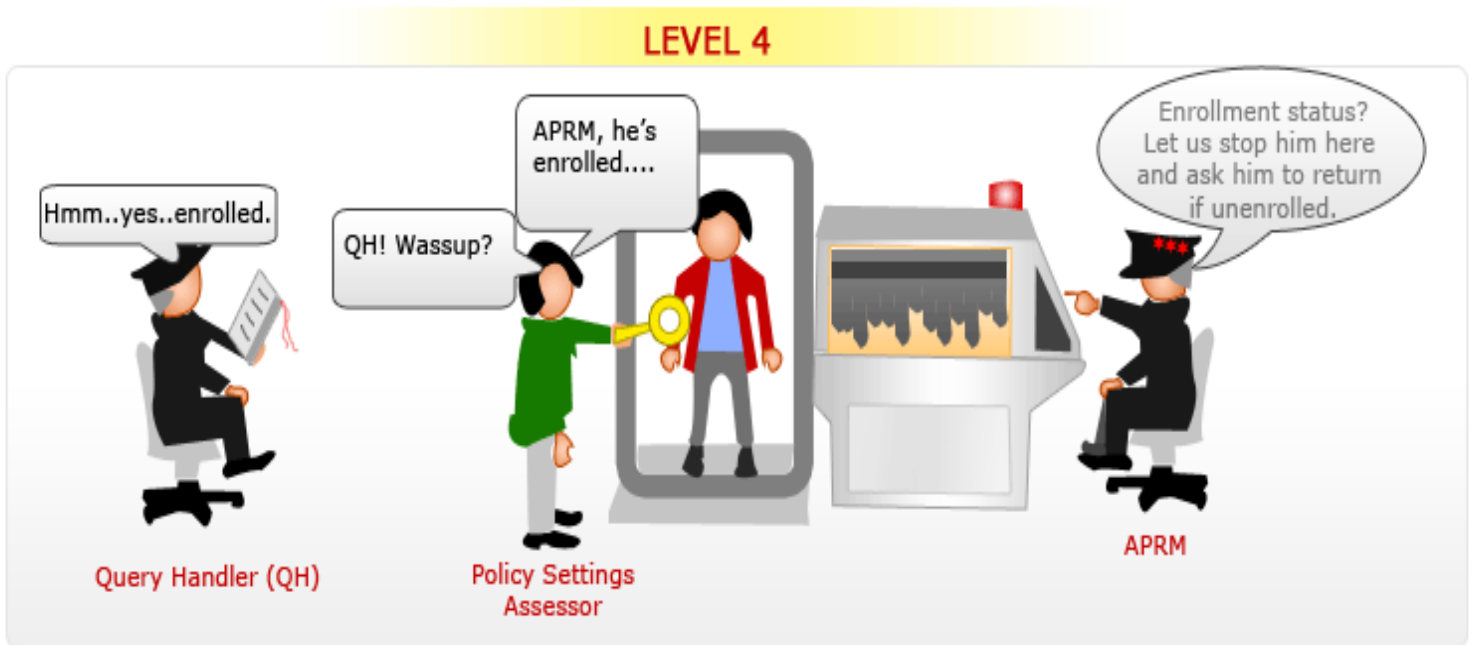


(An important shift occurs here: instead of username, userID will be used hereafter by APRM and other APIs.)

Policy Settings Check: Thereafter, APRM invokes Policy Settings Assessor to check if the user is permitted password self-service, if so, what his policy settings are. To fetch policy settings, this API also makes use of Query Handler.



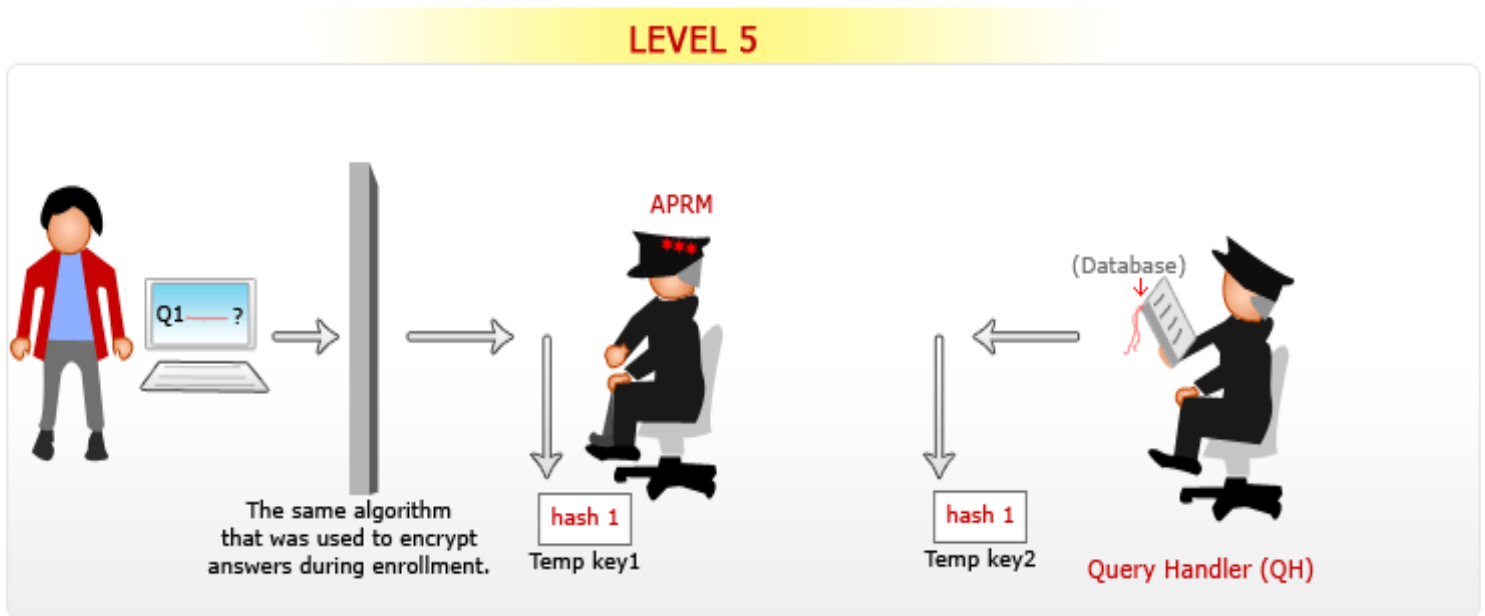
Enrollment Status Check: Only after receiving a positive answer and settings information from Policy Settings Assessor, will APRM proceed further. For a positive answer, APRM summons Enrollment Status Checker to verify whether user has registered by answering security questions. If the user is enrolled, APRM is all set to proceed further; else, it denies the user access.



Identity Check through Security Questions:

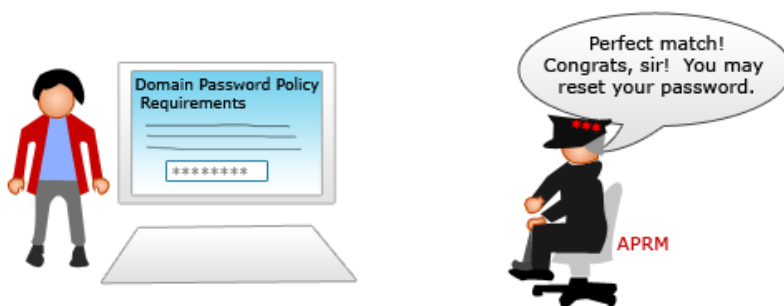
- As soon as APRM receives a go-ahead from Enrollment Status Checker, it fetches the security questions of the userID's account and displays it to the user one by one.
- User answers a question, which gets encrypted via the same one-way hash algorithm that was used during enrollment. The encrypted answer is stored in a temporary key.
- Now, APRM pulls out this userID's answer (stored in an encrypted format) from the product database (with the help of Query Handler, of course) and loads it in a temporary key.
- Then the temporary keys – one holding hash of the security answer user just entered and the one loaded with what APRM retrieved from the database – are compared with each other.

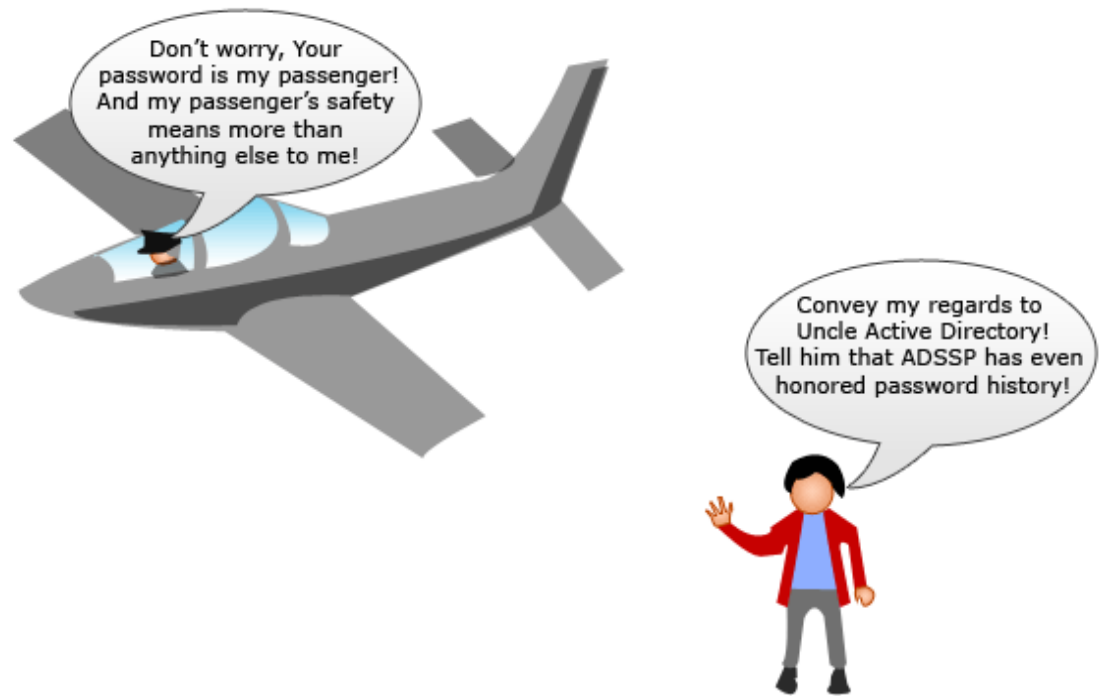
- A match will make APRM present the second question to the user. The whole process goes on like this until user all the questions in his account to prove his identity.



$$\begin{array}{c} \text{hash 1} \\ \text{Temp key1} \end{array} = \begin{array}{c} \text{hash 1} \\ \text{Temp key2} \end{array}$$

Password Reset Phase: Once APRM is convinced about user's identity, it presents him the screen for password reset.





- As soon as the user enters the password, the APRM contacts the appropriate domain controller, presents application account credentials and identifies itself.
- It then uses the same password reset API that comes into use when you reset passwords for users through ADUC, relays the password to the DC, and updates it. It never stores or even caches the password; it only acts like a relay between user and Active Directory for the password, once it completes all its identity management duties.



That is how ADSelfService Plus orchestrates a password reset!

ADSSP Application Account:

To communicate with Active Directory, ADSSP needs an application account. This ideally should be the domain controller's administrator credentials.

However, if you are not willing to provide administrator credentials, you can provide a domain user's account which has Active Directory read/write permission as well as reset password permission.

Now, take a look at what type of protocols ADSelfService Plus uses to contact client and Active Directory.

ADSSP and end-user interaction:

All communications between ADSSP and end-user happen via a simple and self-explanatory web browser interface. These server-client interactions happen in HTTP protocol by default.

MUST DO:

While ADSSP and client communication via HTTP may be safe in a closed LAN, you **MUST** implement HTTPs protocol between ADSSP and clients, if the client is situated outside a LAN and would use internet to access ADSSP. In cases like geographically disparate WAN or use over internet, please apply enable HTTP over SSL, so that client-server communication is encrypted.

ADSSP and Active Directory interaction:

ADSSP interacts with Active Directory using LDAP protocol. It is a must to add in ADSSP all the domains it will serve. By design, ADSSP discovers domains and domain controllers as soon as it is installed. If not make it a point to add everything as necessary.

Must Do:

When you deploy ADSSP in a WAN, please make sure you enable LDAP over SSL to ensure ADSSP and domain controllers communicate in LDAPS (encrypted) protocol. This way communication between Active Directory and ADSSP will stay absolutely safe.

Note:

For both HTTPs and LDAPS, you can install any certification authority (CA), as ADSSP supports almost all the CAs available in the market and several open source CAs.

ADSelfService Plus Security Features:

ADSelfService Plus' defense mechanism against SQL Injection

The only database that ADSelfService Plus will query is its inbuilt database during secondary authentication, report fetching, and certain other operations. And ADSSP has two important defense mechanisms against SQL injection code attacks:

- ✓ ADSSP employs a specially designed query handler framework to check and handle all SQL database queries, which does not allow users to input manipulative SQL code. **(Read the description below)**
- ✓ Besides this ADSSP also employs Zen-like self-defense philosophy in its fight against SQL code injection attacks: The best way to win a confrontation is to avoid it! A fight without fighting! ADSSP rather bypasses querying databases! **(Read the description below)**

Both these make SQL code injection attacks against ADSelfService Plus virtually impossible!

Description of the handler: The SQL query handler framework is an exclusively designed functional subunit, which forms the relay between user input and product database.

- a. Its first and foremost duty is to check the quality and innocuousness of user input, ensure it is harmless, and then pass it through other querying functions.
- b. A query builder in this framework accepts this sanitized input and builds appropriate queries. This framework does not support:
 - Sub-queries.
 - Nested queries
 - Single quotes and host of other carefully chosen characters, which SQL might mistake as a part of its intrinsic code.
 - If a user enters any of the above, the framework would rather treat/convert them as search criteria, which would result in “No Data Found” error!
- c. The handler has the capability of interfacing with any kind of database that you prefer, but this feature is temporarily disabled in ADSSP for now. There are plans to make it all-inclusive in the future.

The framework also checks for other malicious codes, and threats like client-side validation bypass are thwarted successfully too.

Fight without Fighting:

Besides all the basic security measures against SQL injection, including client and server-side user input validation, ADSelfService Plus does an important thing: it never uses the

regular and globally used SQL techniques, nor does it query its intrinsic database directly!

Why carry out validation checks against database in a way that hackers can exploit? Why use SQL structures that hackers know well and can exploit?

In certain cases, ADSelfService Plus never really queries the database in the first place, which in itself solves majority of issues related to SQL code injection. For example, when it needs to verify user's identity using security answers, ADSSP follows the steps below, instead of plain old system of querying database and checking user-submitted answers against it:

1. From the above paragraphs, you must have learnt that every username registered with ADSSP is assigned a unique identifier (UID). When a user submits his username, say, for example J.Smith, the query handling framework stores this input in its register, checks for code intrusion, and then queries database only to pick out the corresponding UID of the username.
2. Meanwhile, another functional subunit gets ready; it is informed of the UID and it carries out a check in its members database, to see if the user really exists. If the user exists and permitted to use this service (which is determined by other APIs as stated above), it loads a temporary key with security answers corresponding to this UID.
3. On the other end, the user-entered answers are also stored in temporary keys. And a comparison between these temporary keys is done, the positive outcome of which helps determine the identity of the user.

In this method, there is no room for code injection at all! This way the vital identification information are safe and hacking becomes a mirage for hackers!

Even if this database is detached and maintained on a separate server, there is no change in ADSelfService Plus database retrieval mechanism.

Controlling end-user behavior to get optimal security answers:

End-user's poor choice of security answers:

Users may end up constructing security answers that could be easily defeated. And there might be users with lackadaisical attitude towards the whole system, who might trivialize the enrollment process by selecting extremely poor answers.

Therefore ADSSP dedicates a set of features to **correct END-USER BEHAVIOR**, making them construct answers as desired by IT administrators.

Two most common foibles of end-users while constructing security answers:

- a. Some users tend to use the same answer for all the questions!
- b. Some would want to use a word or a phrase from the security question as their security answer!

Pretty conveniently ADSSP avoids this from happening. It offers two important features to administrators to:

- a. Prevent users from providing the same answer to multiple questions
- b. Prevent users from using any word of the question in their answers.

Length of the security answers:

The length of a security answer is a key factor in keeping dictionary attacks at bay. Longer the answer, tougher its chances of being compromised. ADSSP allows administrators to control the length of the security answers provided by the users. They can set a minimum specification for answer length.

It should be noted that this feature, in conjunction with the option to prevent users from using any part of the question in their answer, will make end-users provide security answers that are long and not easily guessable based on question. Therefore this feature can even thwart password guessing done by low-level research carried out on a user's Facebook or MySpace accounts.

Suggestion: One good suggestion that will get the better of guessers:

Educate end-users to add to their answers some extra characters – always.

<A favorite catchphrase or song><answer to security question>

For example, following this format the answer to “What is your favorite holiday spot?” would be:

Gin-soaked boy Hawaii

[where “Gin-soaked boy” is the user's favorite song (by the band Divine Comedy) and Hawaii is answer to the question]

All that the user needs to do is keep prefixing or tagging every security answer with this memorable song of his.

Now, anyone who tries to deduce answers by researching the user's holiday spot preferences on social networking sites would only be disappointed to see his researched answers getting defeated.

Though this appears far-fetched and surreal, this is easily achievable with the help of ADSelfService Plus question & answer configuration features. Fix the minimum required length of a security answer as 12 or 14. Now, users have no choice but to add something to their answers to achieve that word length, and the first thing that comes to their mind is what you taught them. And let them know that they can even use their favorite catchphrase, or song, or cheeky answers, or zany words, as long as they are able to recollect them. None of these items is something that they cannot easily remember!

While this keeps out password guessing by near-and-dear or through Facebook/MySpace researching, the only thing left is “dictionary attacks/bruteforcing”. And this can be easily handled by ADSSP’s lockout mechanism!

Mandating security questions:

Good security questions should never go unanswered. If administrators deem a question they constructed will elicit a quality security answer, they can very well force users to answer it!

Man-in-the-middle Attacks:

In this favorite ploy of hackers, they tend to intercept the traffic between a server and client, while looking for private and vital information that could fetch them access or some benefit.

ADSSP gets rid of man-in-the-middle attacks by allowing all the traffic from and to be encrypted. Primarily, ADSSP communicates with Active Directory and the client, using LDAP and HTTP protocols. It is highly compatible with any kind of certification authority (CA), to enable LDAP or HTTP over SSL.

Therefore there is nothing to worry about throwing open ADSSP to internet users, as any of your trusted third party CA encryption could be implemented.

Some of the CAs used by ADSelfService Plus clients:

Comodo.
OpenCA.
GoDaddy.
Verisign.
Digicert.
Thawte.
GeoTrust.

....

....
....

You name it, we support it!

Suggestion:

When you deploy ADSelfService Plus to be accessed over internet – even if for use within WAN – please ensure you enable HTTP and LDAP over SSL.

ADSelfService Plus’ reply to Passphrase guessing/dictionary attacks/bruteforcing:

When someone intends to crack a password, he tries out his guess at the authentication system until he gets through. However, this is a tedious process and could take his entire life to crack a simple 8-character password. Instead, hackers use automated tools, which have pre-loaded password lists. The tools feed passwords from this list into the login textbox, one by one but at great speeds, until it succeeds to break in. Since most of the users have everyday words or names as passwords, hackers use dictionaries and name lists in their hacking tools. Whether seasoned or naïve, every hacker relies on dictionary attack tools to crack passwords.

To prevent hackers by bruteforcing passphrases (security answers) with their wild guesses or dictionaries, ADSSP employs a lockout system. When someone enters wrong security answers more than the threshold allowed, ADSSP locks down his account for a definite period. Administrators can decide the threshold for number of unsuccessful attempts and lockout period.

Some password cracking tools can try out 300,000 passwords in a second. But what good is it when the system locks the user account in just 3 invalid attempts?!

Besides lockout mechanism, ADSSP also employs CAPTCHA, which also plays a very important role in keeping dictionary attacks, guessing and bruteforce at bay. Unless and until CAPTCHA is reproduced, ADSSP does not complete password reset for the users. With dictionary attack tools unable to identify CAPTCHA in the first place, there is no question of reproducing it. Thus it would require hacker’s intervention, which means password cracking will now happen at speeds permissible by human abilities.

When these two features are employed together, then the possibility of successful dictionary attacks is nil.

Suggestion:

If your organizational security policy entails you to do vigilant perusal of audit reports at different intervals in a day, set the lockout period so that it is proximal to review time of “failed attempt at security questions report” provided by ADSSP.

Example: If you are required to perform a check every 4 hours in your shift on this report, then let the lockout time be 4 hours.

If there is no such policy, then you can opt to lock out the account for 1 hour.

Conclusion:

It is clearly evident that it is nearly impossible to compromise ADSelfService Plus security cordon. Choose suggestions provided in this document as per your organizational requirements and offer a secure password self-service.

ANNEXURE:

ADSelfService Plus Installation Requirements:

Hardware Requirements:

Processor: P4 – 1.0 GHz.

RAM: 512 MB.

Disk Space: 200 MB.

Software Requirements:

Supported server operating systems:

- Windows 2000.
- Windows XP.
- Windows 2003.
- Windows Vista.
- Windows 7.
- Windows 2008 Server R2.

Supported browsers:

- Internet Explorer 5.5 and above
- Netscape 7.0 and above
- Mozilla 1.5 and above
- Firefox 1.5 and above

Suggestions:

Make it a point to unblock in firewall mysqld-nt and Java(TM) 2 Platform Standard Edition binary, when you are installing ADSelfService Plus on a firewall-enabled Windows XP or 2003 machines.

ANNEXURE II

Deployment suggestions:

Depending on your organizational security policies or needs, you can deploy ADSSP in any one of the following ways or as needed. To accommodate any kind of deployment strategy, ADSSP has been designed to be simple, at the same time, self-sufficient.

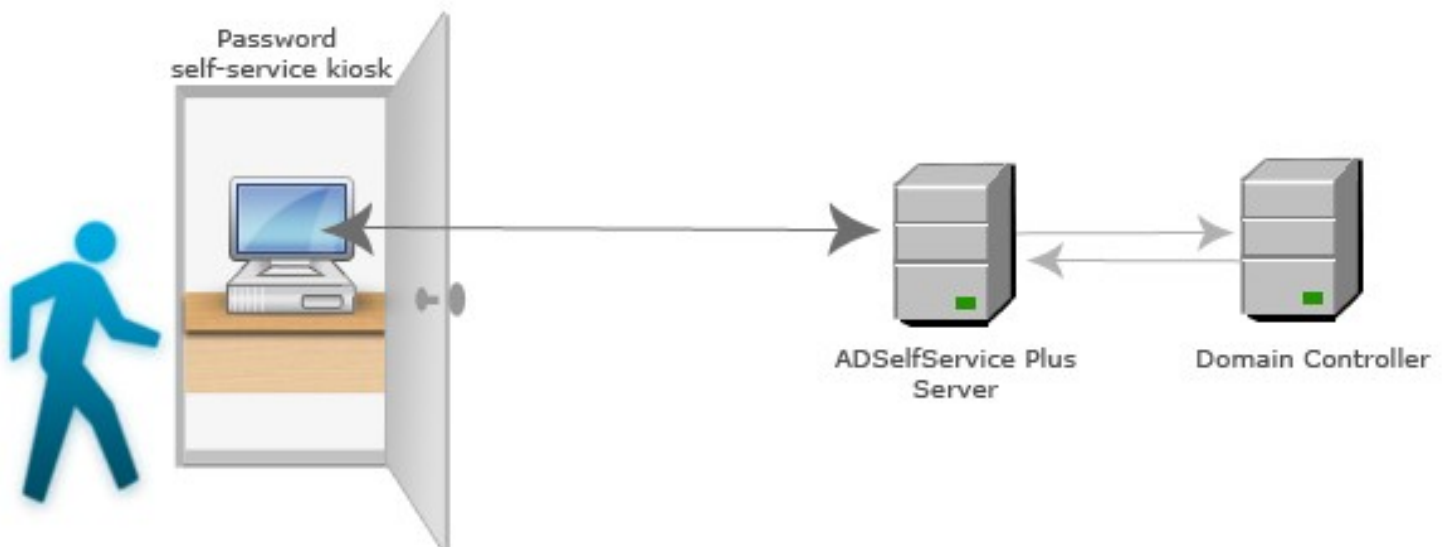
Kiosk mode/Standalone server deployment:

For organizations that have extraordinary security policies, when it comes to awarding of password self-service to end-users (such as avoidance of web-based access to self-service system, vigilance of self-service users through CCTV camera, and so on), it is best to go for kiosk-mode/ standalone server deployment.

Install ADSSP on a technically fulfilling computer (check page Annexure for Installation Requirements), or on a server if you desire so, and ensure its connectivity to the necessary domain controller(s).

Put up this machine in a kiosk (or a safe and accessible place) under CCTV surveillance. That is it! A password self-service kiosk machine is ready to serve any legitimate end-user seeking password self-service.

Unbelievably, the entire process takes only a few minutes!

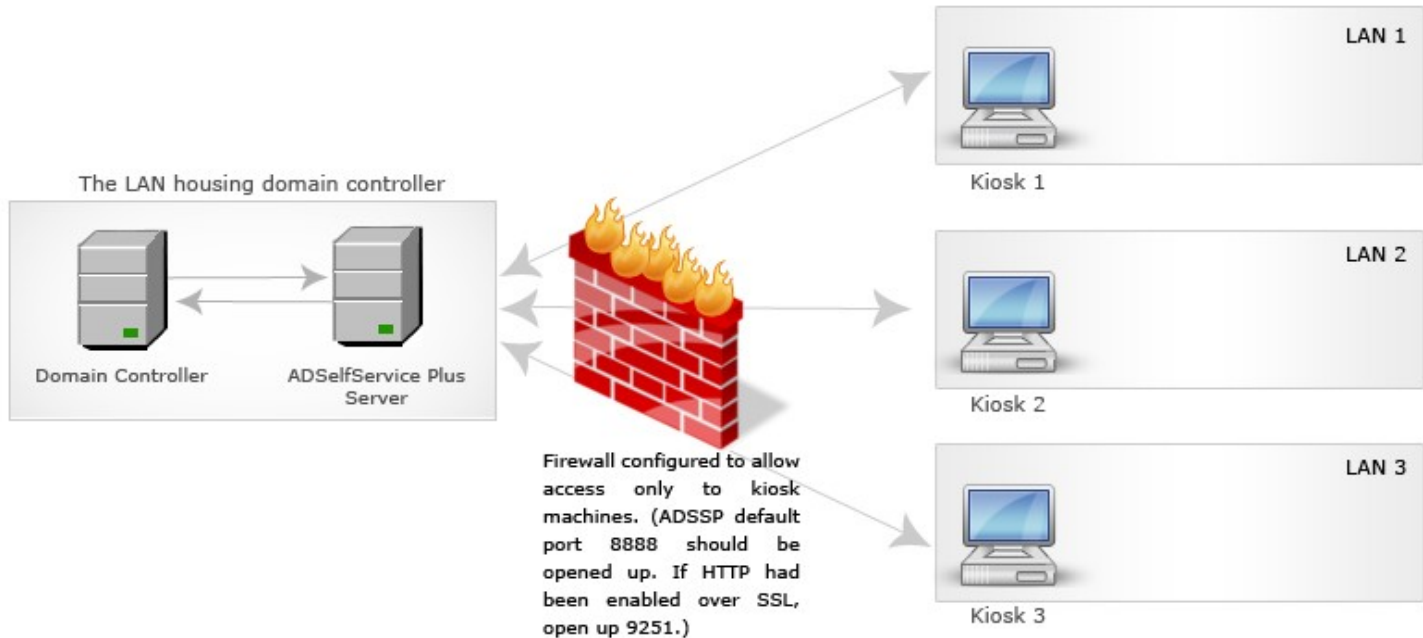


Suggestions:

1. Ensure your kiosk machine does not offer any other service than ADSelfService Plus.
2. If you desire, you may emulate ATM model, and control kiosk access with control devices, say access card. Coupled with camera surveillance, this adds an extra measure of security.
3. Follow organizational policy for checking audit reports. ADSSP provides a set of audit reports, which provides information about the users who used it to self-service their passwords or update Active Directory.

Multiple kiosks, multiple LANs, same building:

1. Install ADSSP on a machine accessible throughout the LAN or only by domain controllers of the organization. This will be the server, which will offer password self-service of ADSSP to the kiosks. Install a personal firewall on this machine. Ensure only port 8888 (or 9251 in case HTTP has been enabled over SSL) of the firewall is open.
2. Put up kiosks in desired locations in the building. In each kiosk, set up a dedicated computer, which should, ideally, do nothing apart from providing access to ADSSP through a browser. Add necessary personal firewall or equivalent software. Ensure only port 8888 (or 9251) is open.
3. In the server firewall, set rules in such a manner that only kiosk machines can contact ADSSP server and access its service. Users should be forced to use only the kiosks.



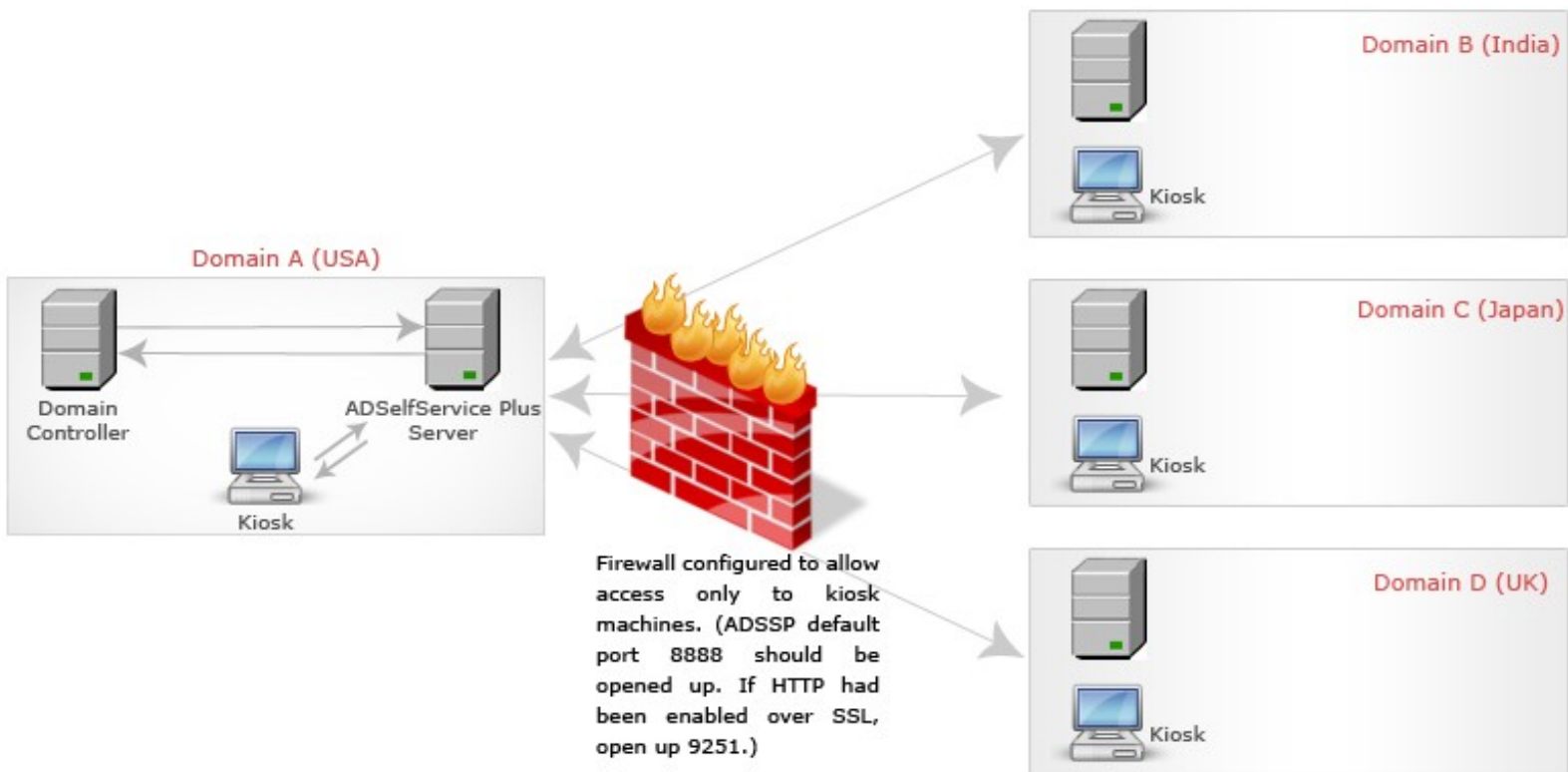
Firewall suggestions:

1. Always use application layer or stateful inspection firewall.
2. Set appropriate access rules in the firewall for controlled access, such as prohibiting access to a range of IP addresses unrelated to the organization, etc.
3. Ensure to include organizational security policies in firewall.

Multiple kiosks, different geographical locations:

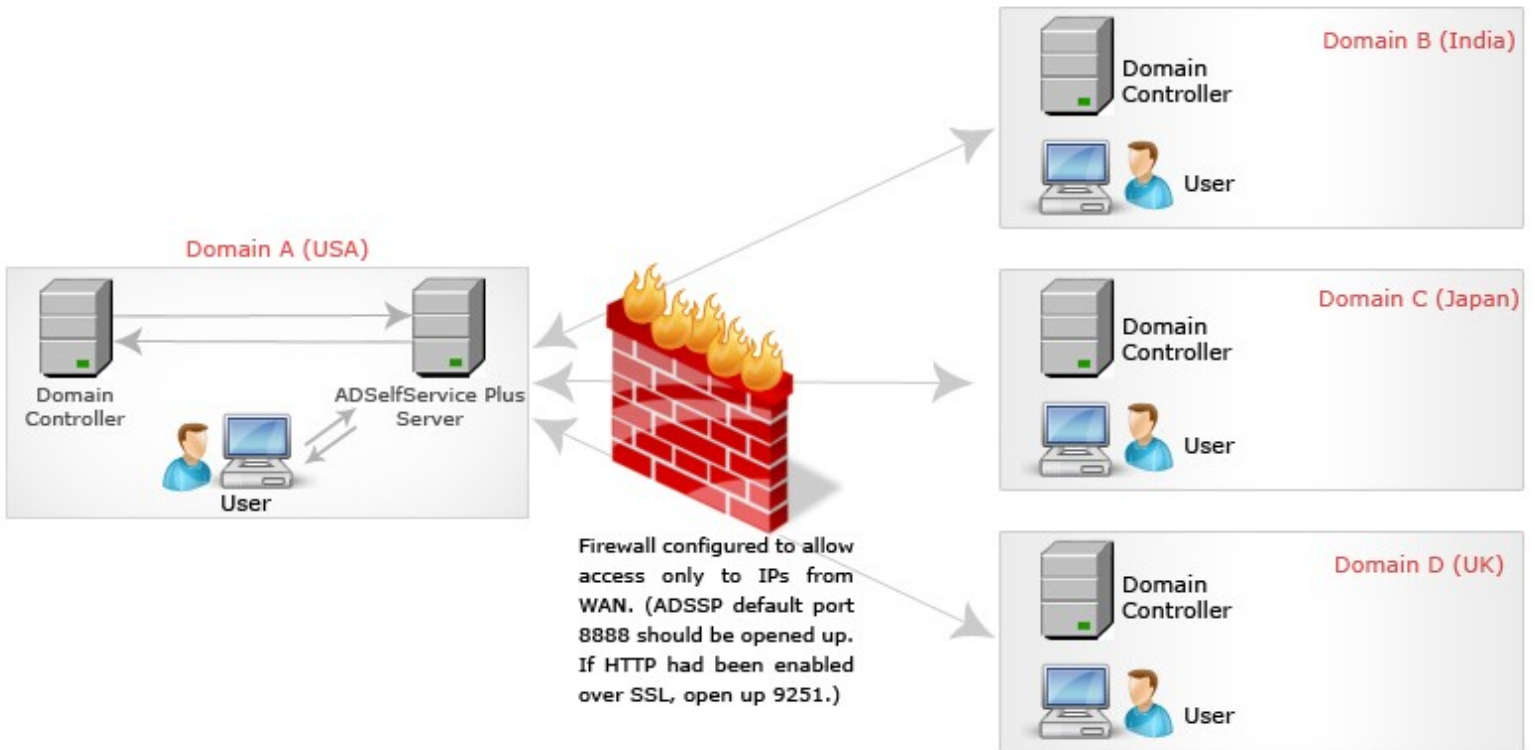
This is not much different from multiple kiosk system in a LAN, except that the ADSSP server will serve even extranet kiosks.

Apply your organizational security standards to the server and kiosk machines as applicable to extranet deployment. Set rules in the server firewall allowing access only to domain controllers and kiosk machines IPs; block the rest.



Multiple domains in a WAN (no access to public IPs or laptops):

1. Install ADSSP on a server or a dedicated machine behind a firewall.
2. In this firewall, enable ADSSP port 8888 (or 9251 if HTTPs is enabled over SSL) only.
3. Configure the firewall to allow only IPs from the WAN to access ADSSP. Block all other extranet IPs or possible intrusions. Check incoming packets for source and destination, sieving in only those from intended users
4. Do not rely on packet filtering alone. Please use stateful inspection or application level inspection.



Access over internet:

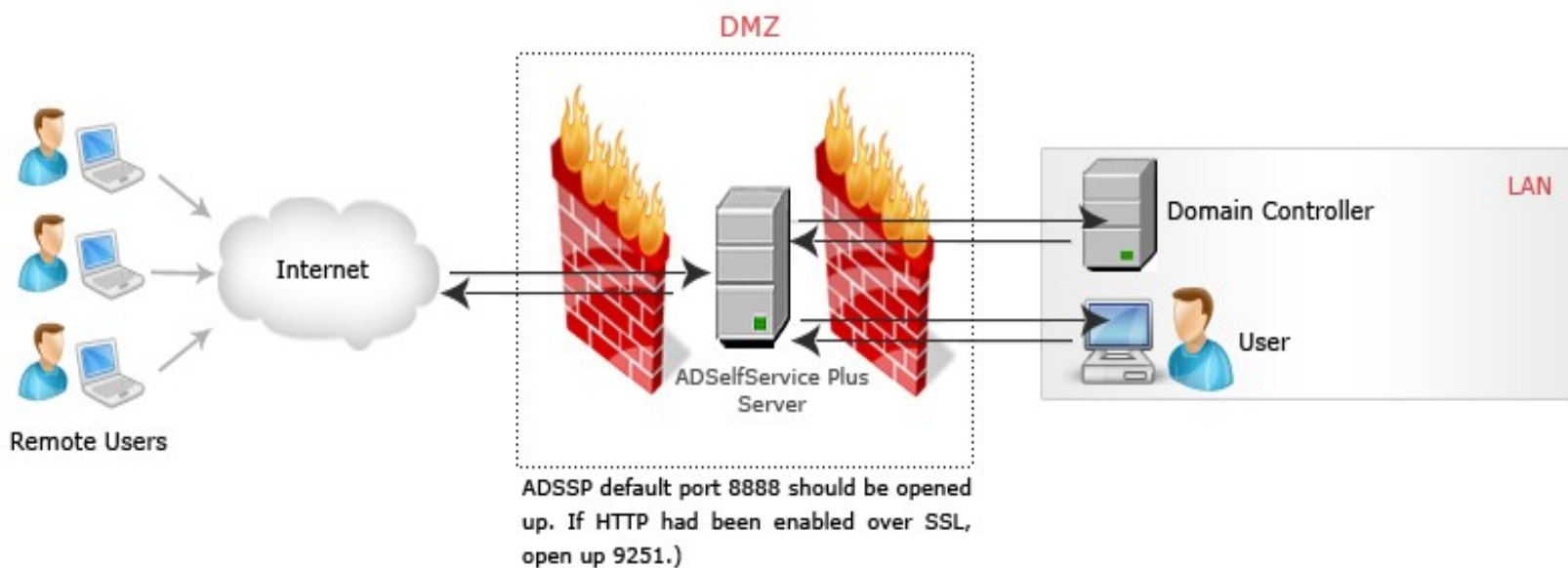
When you allow your laptops users or work-from-home staff to use ADSSP, ensure you follow these steps:

Use dual-firewall DMZ:

Sandwich ADSSP between two firewalls; one between internet and ADSSP, and second between ADSSP and intranet.

Since you cannot block public malicious IPs without affecting good ones nor do MAC address filtering to ensure only known laptops connect to the DMZ, ensure you set a good lockout policy in ADSSP.

Make it a point to check the ADSSP audit reports daily.



Firewall recommendations:

- It is best to use application layer firewall and stateful inspection firewall.
- Set rules to check unwanted intrusions.
- When exposing ADSSP to internet (for WAN), set appropriate rules, as to who can access the application. Configure the firewall to only allow data traffic from/to certain IP addresses available/viable in your organization. Block others.
- Audit your firewall regularly (or as per your organizational security policy).
- Regularly backup firewall rules and settings.
- Also use antivirus and anti-intrusion software.