Elevator Pitch

Whitenoise makes any point-to-point IP communication link far more secure through the distribution and management of a single private Identity Management keys. This addresses many of the growing concerns in light of increasing hacks, man-in-the-middle attacks, and even foreign agency surveillance.

Whitenoise Security as Service (SECaaS) uses distributed keys to establish point-to-point encrypted tunnels and to create and distribute session keys for dynamic, secure point-to-point connections for other network uses such as key distribution. Potential uses of our technology include securing “the cloud”, Internet of Things, i.e. any digital context.

Server Features

- Store, create, distribute and manage Whitenoise keys
- Electronically distribute unique private keys
- Perform continuous dynamic one-time-pad authentication of all endpoints

One distributed key

- Maintains continuous identity management and data provenance
- Performs all network security controls
- Is unbreakable

Benefits

- Prevent all known/anticipated cyber attack classes like man-in-the-middle and quantum.
- It is easy to implement and understand, fast, and requires no training.
- It is interoperable and scalable for real-time enrollment of new endpoints and networks.
- Secure identities can be restored and refreshed for victims i.e. Target, Home Depot etc.

Why we are awesome:

A single call to Dynamic Identity Verification and Authentication (DIVA) at single-sign on is all that is required. Whitenoise is the only national security level cryptography that can be deployed in contexts where PKI is not usable because of its excessive computational overhead.

Disruptive? Yes – security with storage for $1/yr per year! Let’s make a Whitenoise key as we speak.

What we are looking for:

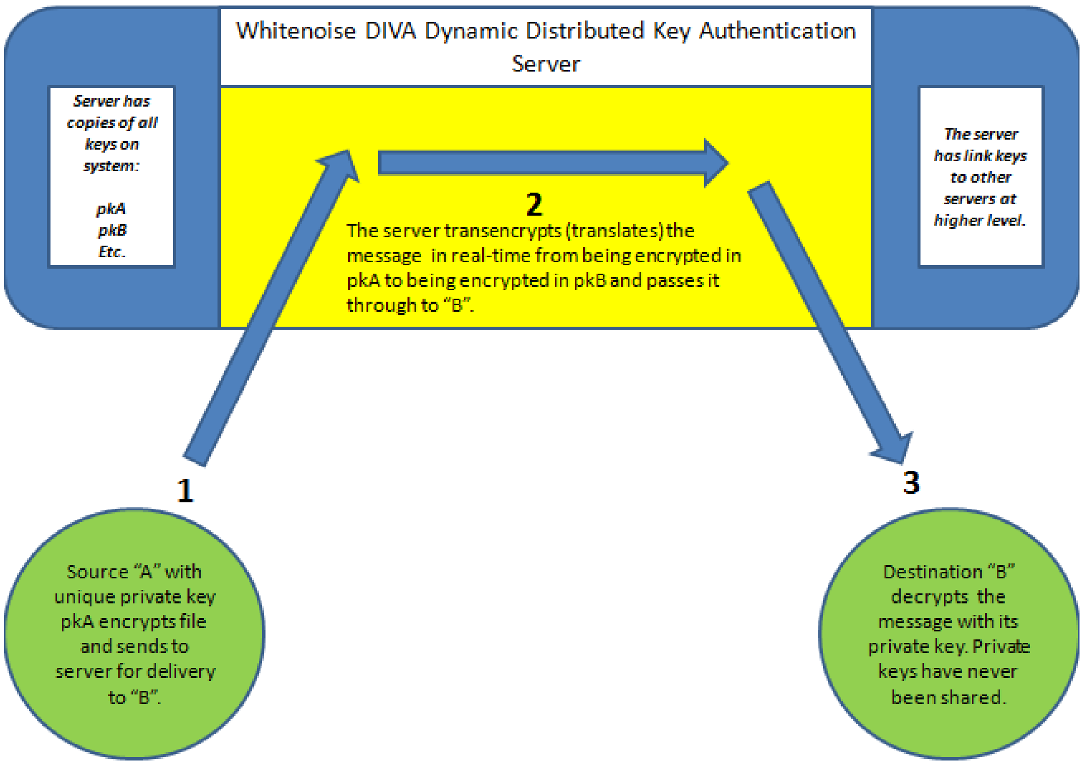
- Investment
- Licensing country by country with exclusives available for protectable patented technologies
- Partnerships
- Purchase order for TCSV offer of security for $1 per endpoint per year
- Purchase order for delivering proof of concept solving your worst security problem
- Entry into your innovation labs (if any) just as we have with Tata and GDNexus

Rules:

In symmetric, dynamic, distributed key systems the server has copies of all the keys on a system. The keys are stored in an encrypted state. The keys are always kept separate from the last current dynamic offsets.

Each endpoint has only its unique, distributed, private/secret key. Secret keys are NEVER shared between endpoints. There is never key or offset exchange after setup. The following illustration shows a system in its simplest configuration.

Direct point-to-point process in symmetric, distributed key systems.

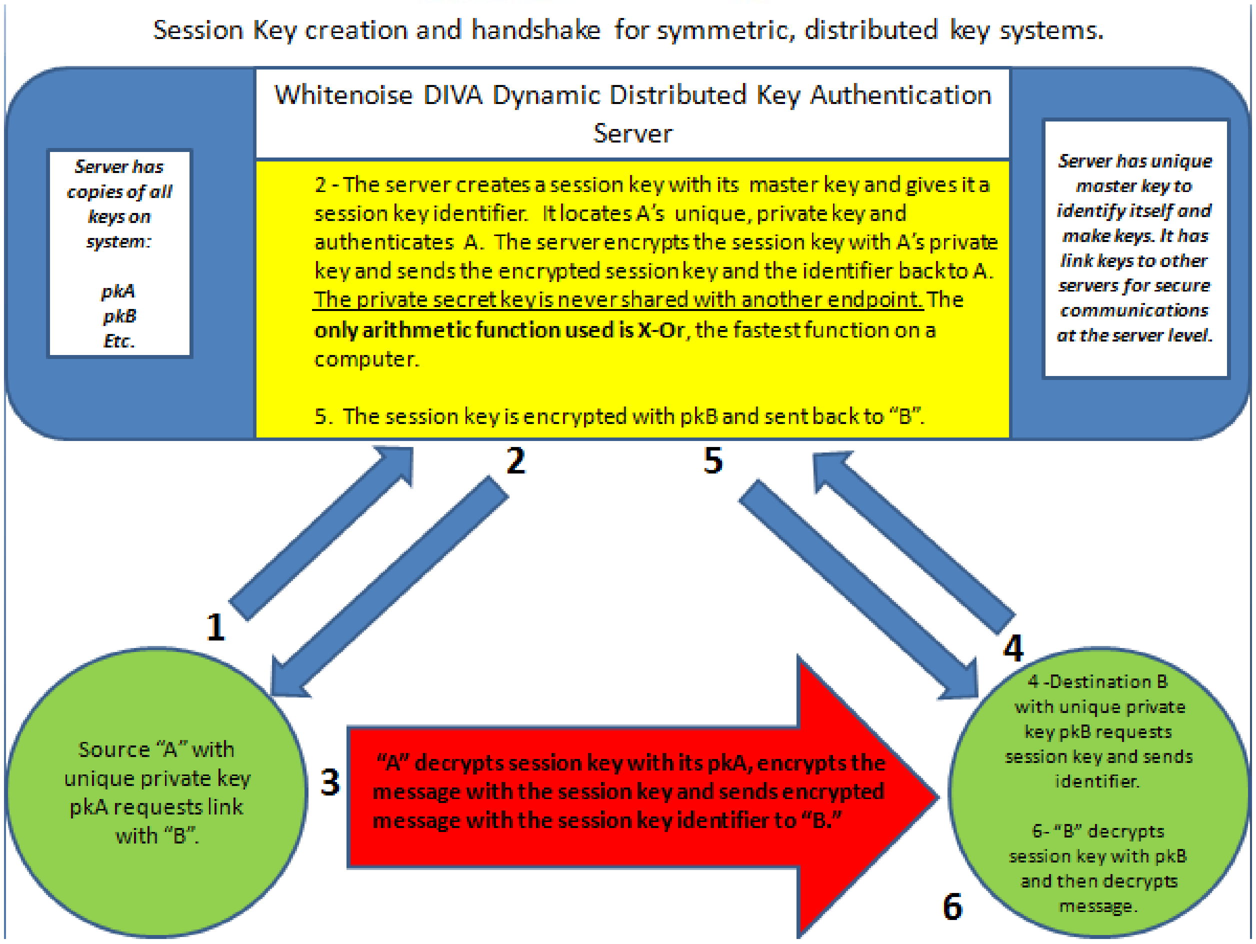


Whitenoise DIVA Dynamic Distributed Key Authentication Server
Server has copies of all keys on system: pkA pkB Etc.
The server has link keys to other servers at higher level.
2
The server transencrypts (translates) the message in real-time from being encrypted in pkA to being encrypted in pkB and passes it through to "B".
1
Source "A" with unique private key pkA encrypts file and sends to server for delivery to "B".
3
Destination "B" decrypts the message with its private key. Private keys have never been shared.

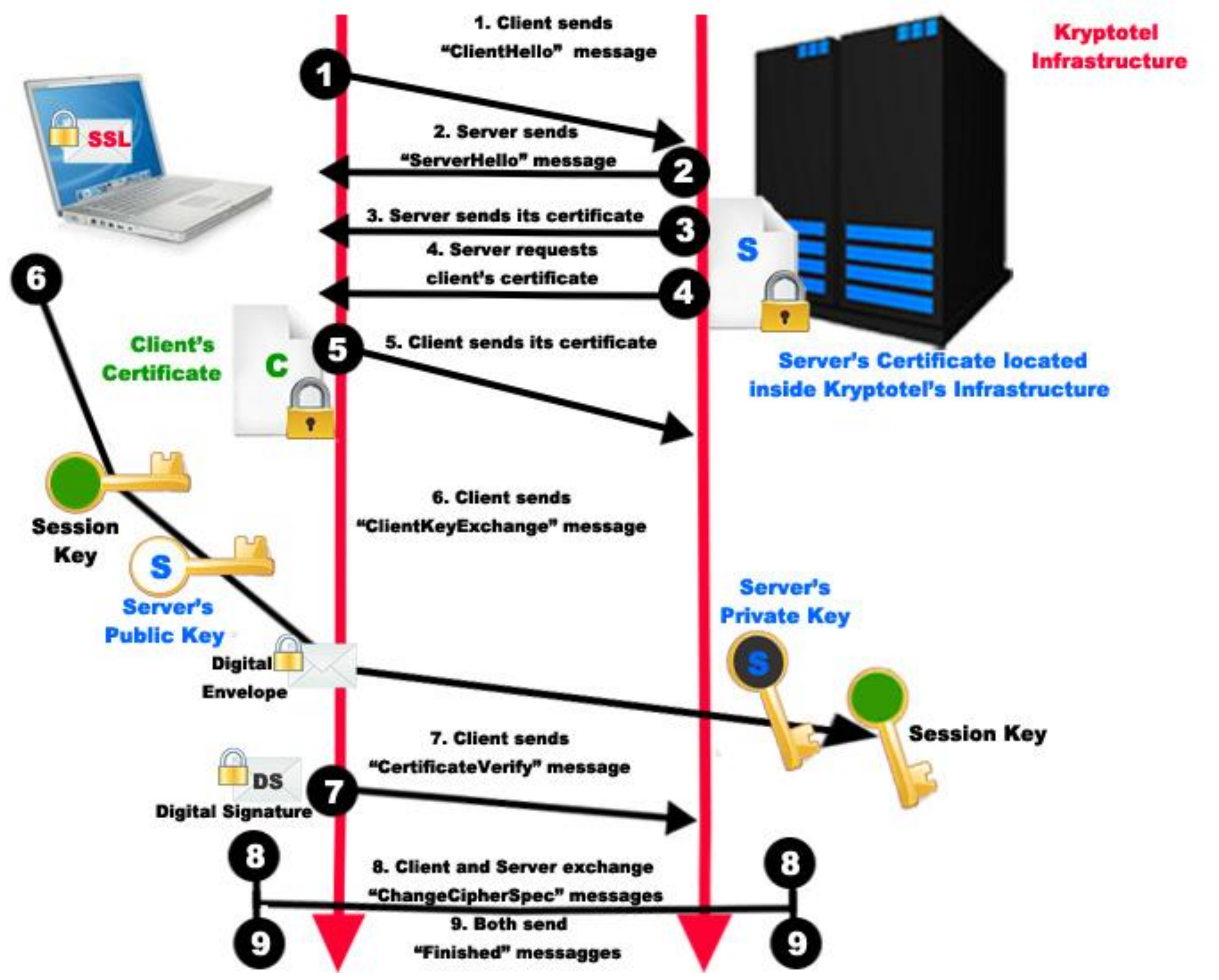
It is desirable to be able to generate session keys in order to communicate with endpoints that do not yet have their own private/secret key. The purpose is to be able to establish secure communications with a new endpoint without first having to copy a key physically to that endpoint as has been traditional in distributed key systems. This allows simple scaling of the network. It facilitates authentication and secure, one-time key distribution. It facilitates the establishment of a secure point-to-point connection between endpoints without intercession (trans-encryption) by the server. The server continues to dynamically and continually authenticate the endpoints but no encrypted traffic is passing through it for potential capture.

The following example shows another secure communication protocol where parties have distributed keys. Initial online enrollment can be seen at:

http://www.wnlabs.com/Presentations/Bringing_in_Legacy_Appliances_to_Secure_Networks.pps


Session Key creation and handshake for symmetric, distributed key systems.
Whitenoise DIVA Dynamic Distributed Key Authentication Server
2 - The server creates a session key with its master key and gives it a session key identifier. It locates A's unique, private key and authenticates A. The server encrypts the session key with A's private key and sends the encrypted session key and the identifier back to A. The private secret key is never shared with another endpoint. The only arithmetic function used is X-Or, the fastest function on a computer.
5. The session key is encrypted with pkB and sent back to "B".
Server has copies of all keys on system:
pkA
pkB
Etc.
Server has unique master key to identify itself and make keys. It has link keys to other servers for secure communications at the server level.
1
2
5
4
3
6
Source "A" with unique private key pkA requests link with "B".
"A" decrypts session key with its pkA, encrypts the message with the session key and sends encrypted message with the session key identifier to "B."
4 -Destination B with unique private key pkB requests session key and sends identifier.
6- "B" decrypts session key with pkB and then decrypts message.

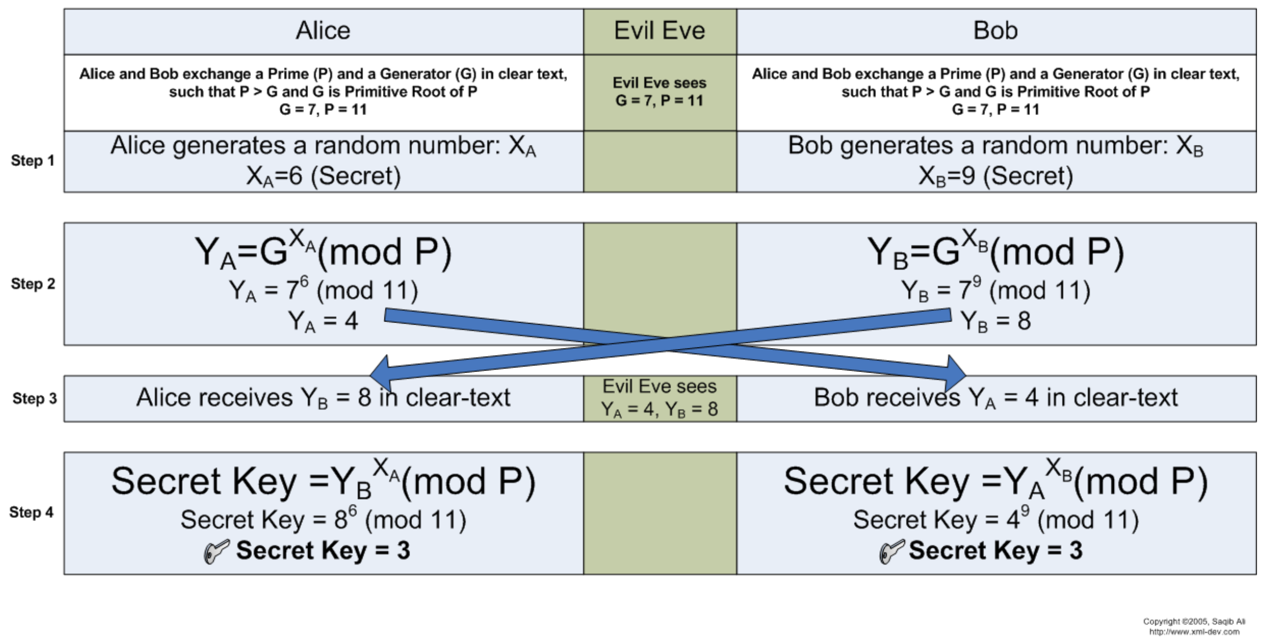
For comparison, let's look at the complexity of a typical asymmetric, public key process and key exchange handshake.


1. Client sends "ClientHello" message
Kryptotel Infrastructure
SSL
2. Server sends "ServerHello" message
3. Server sends its certificate
4. Server requests client's certificate
S
Client's Certificate
C
5. Client sends its certificate
Server's Certificate located inside Kryptotel's Infrastructure
6
6. Client sends "ClientKeyExchange" message
Session Key
S
Server's Public Key
Server's Private Key
Digital Envelope
Session Key
7. Client sends "CertificateVerify" message
DS
Digital Signature
8. Client and Server exchange "ChangeCipherSpec" messages
9. Both send "Finished" messagges

Its complexity is further aggravated by the very intensive mathematics required by PKI. It requires so much overhead in power, space and computational resources that it is literally unusable in many environments including the Internet of Everything where the majority of networked components operate under restrictive environments.

This is one mathematical technique for creating an asymmetric, public key session key. By comparison, after key load, the only operation used by DIVA and DDKI is X-Or. The either-or function is the fastest function available on a computing device.

Diffie Hellman Key Exchange



	Alice	Evil Eve	Bob
	Alice and Bob exchange a Prime (P) and a Generator (G) in clear text, such that P > G and G is Primitive Root of P **G = 7, P = 11**	**Evil Eve sees G = 7, P = 11**	**Alice and Bob exchange a Prime (P) and a Generator (G) in clear text, such that P > G and G is Primitive Root of P** **G = 7, P = 11**
Step 1	Alice generates a random number: X_A X_A=6 (Secret)		Bob generates a random number: X_B X_B=9 (Secret)
Step 2	$Y_A=G^{X_A}(\text{mod } P)$ $Y_A = 7^6$ (mod 11) $Y_A = 4$		$Y_B=G^{X_B}(\text{mod } P)$ $Y_B = 7^9$ (mod 11) $Y_B = 8$
Step 3	Alice receives $Y_B = 8$ in clear-text	Evil Eve sees $Y_A = 4$, $Y_B = 8$	Bob receives $Y_A = 4$ in clear-text
Step 4	Secret Key $=Y_B^{X_A}(\text{mod } P)$ Secret Key = 8^6 (mod 11) **Secret Key = 3**		Secret Key $=Y_A^{X_B}(\text{mod } P)$ Secret Key = 4^9 (mod 11) **Secret Key = 3**

The security of public key systems has always been fatally flawed. They can never prevent Man-in-the-Middle attacks and a host of other attack classes. In the past, computers were so slow that none of the attacks were considered feasible. Now computers are so fast it is simple to ravage our networks because of lack of sustainable identity and data provenance and easily compromised security processes.

Dynamic Distributed Key Infrastructures (DDKI) and Dynamic Identity Verification and Authentication (DIVA) prevent all known and anticipated cyber attack classes:

- Man-in-the-Middle attacks are prevented because there is no key exchange.
- Side Channel attacks are prevented because all operations are order 1 after key load and because there is no access to the key.
- Mathematical and factoring attacks are prevented because keys are created by a binary mechanical process as opposed to arithmetic ones requiring multiplication and mods.
- Botnet attacks are prevented by configuration with server so the botnet never has access to all the key material to authenticate data being sent OUT of a network or computer.
- Brute force attacks are not feasible with the continually changing dynamic offsets.
- Denial of service attacks can be prevented by exploiting unbreakable identity and a proxy for secure network access so that hackers could never get on a network.
- Quantum computing attacks are prevented because every variable is variable.

André Brisson - founder Whitenoise Labs: www.wnlabs.com abrisson@wnlabs.com