The PCI Compliant Database.

Christophe Pettus
PostgreSQL Experts, Inc.
PGCon 2016

Greetings!

- Christophe Pettus
- CEO, PostgreSQL Experts, Inc.
- thebuild.com — personal blog.
- pgexperts.com — company website.
- Twitter @Xof
- christophe.pettus@pgexperts.com

PGX POSTGRESQL EXPERTS, INC.

So, "PCI"?

- PCI is the Payment Card Industry Security Standards Council.
- Sets security standards for any system that processes payment cards.
- What we're really talking about is PCI-DSS, the Data Security Standard.
    - Most recent version: 3.1, April 2015.

Why do I care?

- You like getting paid, don't you?
- Any site that touches payment card information needs to comply with PCI.
- All of it. No exceptions.
- No really, that exception you think you have? You don't.

What does it mean to "comply"?

- You know, that's a really good question.
- To "comply" means that you have passed an audit.
  - Below a certain volume of transactions, you can self-audit.
  - *But you still must comply with every part of PCI, no matter what.*

Who has to comply?

- Any site that ever processes a primary account number (PAN).
  - That's that number on the front of your credit card.
- Even if you don't store it in a database, you still have to comply.

PGX POSTGRESQL EXPERTS, INC.

So, if I comply, I'm safe, right?

- No.
- Passing the audit just means you get to play, not that you get to win.
- If you have a breach, having passed an audit provides *no protection whatsoever* for liability.

This talk.

- Today, let's talk about getting a PostgreSQL database PCI compliant.
- There's a lot more involved in getting fully PCI compliant.
- But PCI compliance is a good jumping-off point for general system security.

Read the Documentation.

- Be sure to get and read a copy of PCI-DSS.
- There's no way to go through all the ins and outs in one talk.
- This focuses on technical matters, as related to a database…
- … but the policies and procedures are also very important.

PGX POSTGRESQL EXPERTS, INC.

Caveat Lector.

- This is the absolute minimum you need to do for PCI compliance.
- By itself, it does not necessarily mean you will pass an audit.
- Think of this as the start of your security journey, not the end!

PCI Structure

- PCI-DSS 3.1 has six areas, with a total of twelve requirements.
- Each one of which has implications for a PostgreSQL system.
- Let's go through each requirement, he said! It'll be fun, he said!

Requirement 1: Firewalls.

- “Install and maintain a firewall configuration to protect cardholder data.”
- In general, this section requires that a server only offers the absolute minimum level of service necessary.
    - So, no IRC server on the PostgreSQL box, OK?

PGX POSTGRESQL EXPERTS, INC.

Requirement 1: Firewalls.

- PostgreSQL server is running PostgreSQL, and just that.
- Only port 5432 is available.
    - Plus *mandatory* management ports.
- Port 5432 is restricted to only application servers that must talk to the database, by specific IP address.

Requirement 1: Firewalls.

- Use pg_hba.conf to restrict traffic to authorized IPs, with mandatory SSL connections.
- Use iptables (or your favorite) to additionally restrict incoming services.

PGX POSTGRESQL EXPERTS, INC.

Requirement 1: Firewalls.

- Do not allow direct public logins via SSH to the database host. Require a hop through a specific bastion host.
- Restrict access to the bastion host by VPN; do not simply trust bare SSH (even on a nonstandard port).
    - Everyone tries 2222 now. C'mon.

PGX POSTGRESQL EXPERTS, INC.

Requirement 2: Security Policies.

- "Do not use vendor-supplied defaults for system passwords and other security parameters."
- Well, doh, right?

Oh, look: a vendor-supplied default.

```
Speedbird-8:~ postgres$ psql postgres

psql (9.5.3)

Type "help" for help.


postgres=#
```

Requirement 2: Security Policies.

- There is no such thing as “trust” mode authentication. Forget it ever existed.
- Always require specific users, even superusers.
- Do not use the postgres Unix or database user. Require specific users.
- LDAP is your “friend,” here.

Requirement 2: Security Policies.

- For system administration, use specific users and sudo; never, ever allow root logins.
- Use a password manager. Always always always.
- For critical passwords, use split passwords with dual custody.

PGX POSTGRESQL EXPERTS, INC.

Requirement 2: Security Policies.

- Versions of TLS below 1.2 don’t exist anymore.
- This includes your public-facing website!
- You have until June 2016. Get on it.

Requirement 2: Security Policies.

- Always subscribe to the pgsql-announce list.
- Always immediately apply any security-related updates.
- Also subscribe to the appropriate security list for your platform.
- Keep up to date with patches, already!

PGX POSTGRESQL EXPERTS, INC.

Requirement 2: Security Policies.

- Make it someone's job.
- Make sure they do it.
- Never, ever allow a critical security patch to go unheeded.
- Ever ever ever.

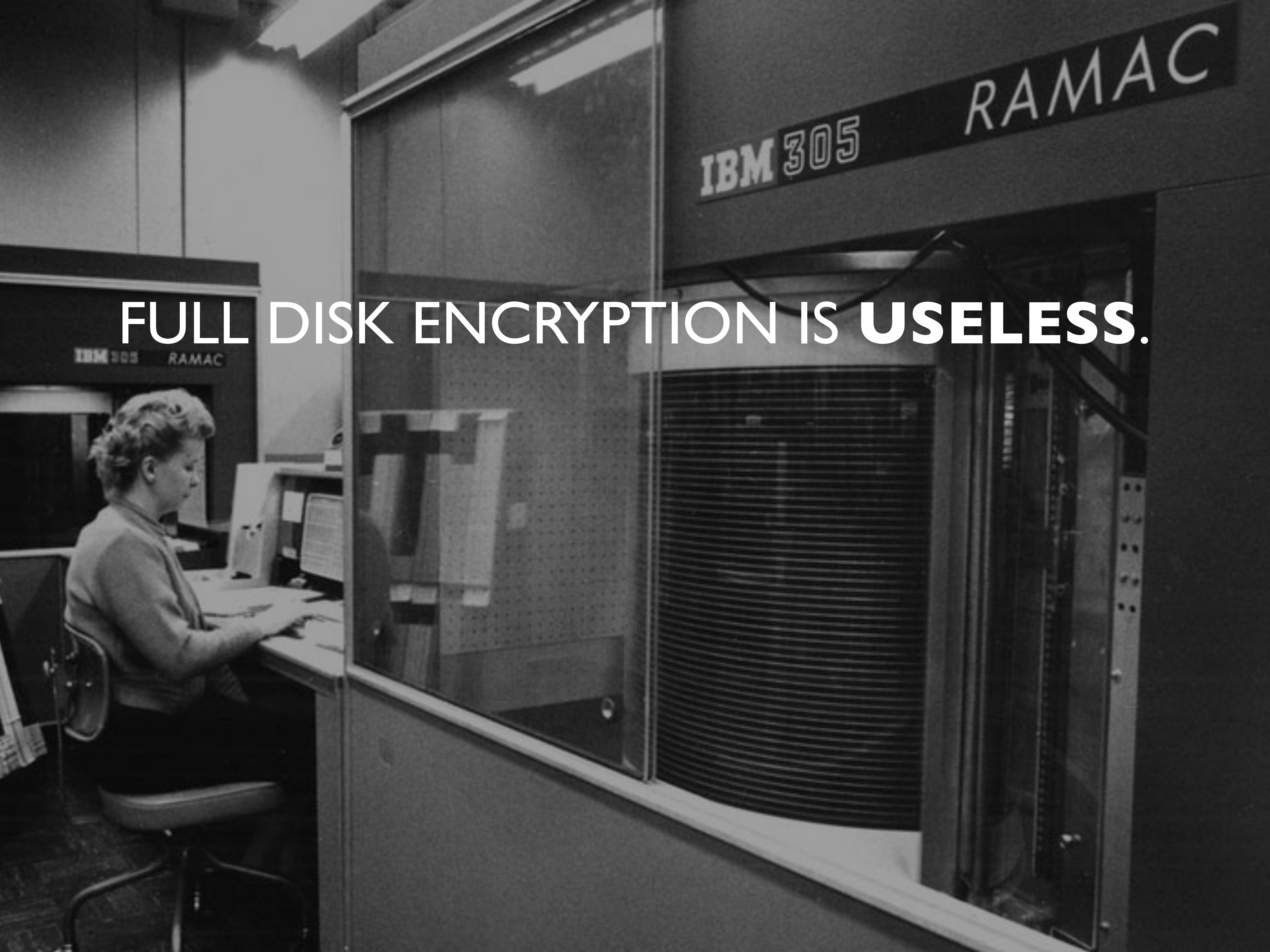
PGX POSTGRESQL EXPERTS, INC.

Requirement 3: Data Security.

- “Protect stored cardholder data.”
- At last! What we’re here for!

Requirement 3: Data Security.

- “No problem! We’ve layered luks on top of lvm on top of EBS, and we’re all set!”
- No.
- Full disk encryption is useless.
- Let me say that again.



FULL DISK ENCRYPTION IS **USELESS**.

FDE protects you against exactly one problem…

- … theft of the media.
- That's it.
- That is about 0.00000002% of the actual intrusions that you have to worry about.
- Easy rule: If psql can read it in cleartext, it's not secure.

PGX POSTGRESQL EXPERTS, INC.

Per-Column Encryption.

- Always encrypt specific columns, not entire database or disk.
- Better performance, higher security.
- Key management is a pain.
- Automatic restart in a high-security environment is essentially impossible.
    - Assume a human will be in the loop.

Primary Account Number.

- Of course, the PAN must be encrypted.
- Algorithm must be a well-known secure one (AES is considered the standard).
- **Never** roll your own crypto.
- Keys cannot be baked into code or stored in repositories.

Masked Number.

- It’s OK to retain the first six and last four of the PAN for display purposes.
  - (Really, just keep the last four and card type.)
- You can also store a hash of the card number for indexing purposes, BUT:

Be careful with hashes!

- It’s very easy to reverse some hashes if you have the masked number!
- Only store four digits, and use a very strong hash like SHA-512.

So, how about pgcrypto?

- pgcrypto is a /contrib module that contains cryptography functions.
- Why not use it to encrypt the PAN?
- I mean, it's just sitting there, right?

PGX
POSTGRESQL
EXPERTS, INC.

```
INSERT INTO super_secret_table(card)
   VALUES(
      pgp_sym_encrypt('4111111111111111',
                      'mysuperpassword'));
```

PGX POSTGRESQL EXPERTS, INC.

```
2016-05-19 10:40:42.524 PDT,"xof","xof",
99245,"[local]",573dfa20.183ad,9,"INSERT",
2016-05-19 10:38:40 PDT,2/0,0,LOG,
00000,"duration: 1.712 ms  statement: INSERT
INTO super_secret_table(card)
VALUES(pgp_sym_encrypt('4111111111111111',
'mysuperpassword'));",,,,,,,,,"psql"
```

Not so great.

- PostgreSQL's text logs could expose the PAN.
- That's another hop the data has to take in cleartext form.
- Always do the encryption in the application, not in the database.

```
CREATE TABLE cardinfo(
    id uuid primary key,
    card_type card_types not null,
    masked_card char(4) not null,
    card_hash varchar(1024) not null,
    enc_pan bytea not null,
    enc_cvv bytea not null,
    expiration_date date not null
);
```

What's wrong with this schema?

- Everything's OK except…
- You cannot store the CVV.
- No, you cannot store it at all.
- Not even encrypted.

Well, OK, you can store it...

- ... for as long as the authorization takes.
- OK, we'll just store it, process the authorization, and clear it out. No problem!
- So, about that PostgreSQL secondary...
- ... with all of those WAL logs backed up?

PGX
POSTGRESQL
EXPERTS, INC.

No storage means “no storage.”

- Not in WAL segments.
- Not in backups.
- Not in text logs.
- Even in encrypted form.
- Ever.
- Just don’t write it to the database.

Requirement 4: Encrypt Data in Flight.

- "Encrypt transmission of cardholder data across open, public networks."
- Goodness gracious, I hope you are doing this.
- Generally, we're entering an TLS-everywhere world, so go with that.
- Remember, no SSL or TLS 1.0-1.1 anymore.

Use “SSL” for PostgreSQL.

- Require SSL connections to PostgreSQL.
- If you are using pgbouncer, use stunnel to get SSL.
- Ideally, use proper certificate management.

Requirement 5: Protect Against Malware.

- "Protect all systems against malware and regularly update anti-virus software or programs."
- Specifically work machines accessing the database.
  - This is generally how large-scale data thefts happen.

Requirement 6: Be a Grownup.

- “Develop and maintain secure systems and applications.”
- Document your system administration procedures. Do security code reviews and audits. Make sure your deployment procedures are solid.

Requirement 6.5.1: SQL Injection Attacks.

- Always use proper parameter substitution in your library!
- Never build SQL by text substitution unless it is absolutely necessary (for example, variable table names).
- All user input is hostile and wants to kill you all the time.

PGX POSTGRESQL EXPERTS, INC.

Requirement 7: Restrict Data by Need-to-Know.

- “Restrict access to cardholder data by business need to know.”

This means…

- … don’t give every developer production system access.
- … identify and qualify the system administrators who need global system access.
- … scrub data that comes out of production for development testing.

Requirement 8: Passwords, yur doin it rong.

- “Identify and authenticate access to system components.”

User accounts must be…

- … associated with a particular human being, not a role.
- … locked out after (no more than) six attempts.
- … immediately revoked for terminated users.

All relevant system passwords must be…

- … complex (and this needs to be enforced, not just policy).
- … changed every 90 days.
- … encrypted in transmission.
- … not the same as one of the last four on that account.

Two-Factor Authentication is now required!

- Two of these three:
  - Password or passphrase.
  - Physical device or smartphone app.
  - Biometric device.

Sessions must be…

- … logged, including user activity during the session.
- … terminated after being idle 15 minutes.

For PostgreSQL…

- … make sure each user has their own unique account.
- … log all connections and disconnections.
- … log all activity by directly-connecting users (as opposed to the application).
- … do not permit logins as the postgres superuser.

Requirement 9: The Glass House.

- "Restrict physical access to cardholder data."
- This means real security (access control, video, mantrap, biometrics) on your server room.
- Make sure your cloud provider provides this for the cloud they are providing to you!

PGX POSTGRESQL EXPERTS, INC.

Requirement 10: Log Everything.

- "Track and monitor all access to network resources and cardholder data."
- Make sure everything is logged, and those logs are kept secure and cannot be tampered with. (rsyslog, etc.)
- Make sure that the log record can be traced back to an individual person.

BUT!

- You cannot log primary account numbers or CVVs in cleartext.
- This is another good reason to encrypt in the application, not in the database.

Requirement 11: Trust, but Verify.

- “Regularly test security systems and processes.”
- Hire external penetration testing firms. Encourage developers to poke at security.
- Hire PCI audit companies that actually understand security, not just run pen test scripts.

This actually happened.

- “We need you to disable your firewall.”
  - “Um, why?”
- “Our penetration test script is failing because the firewall won’t let it through.”
  - “This… sounds kind of like what a firewall is supposed to do, to me.”

PGX POSTGRESQL EXPERTS, INC.

Requirement 12: Write That Down.

- "Maintain a policy that addresses information security for all personnel."
- Make sure all security procedures are documented, policies set, and do proper risk assessment.
- You should be doing this for your database anyway.

Appendix B: The Bargaining Stage of Grief

- What if you simply can't comply?
- Appendix B allows you to write up a "compensating control."
- In effect: "We can't do exactly what the standard says, but we can do this, which is just as good."

PGX POSTGRESQL EXPERTS, INC.

Such as?

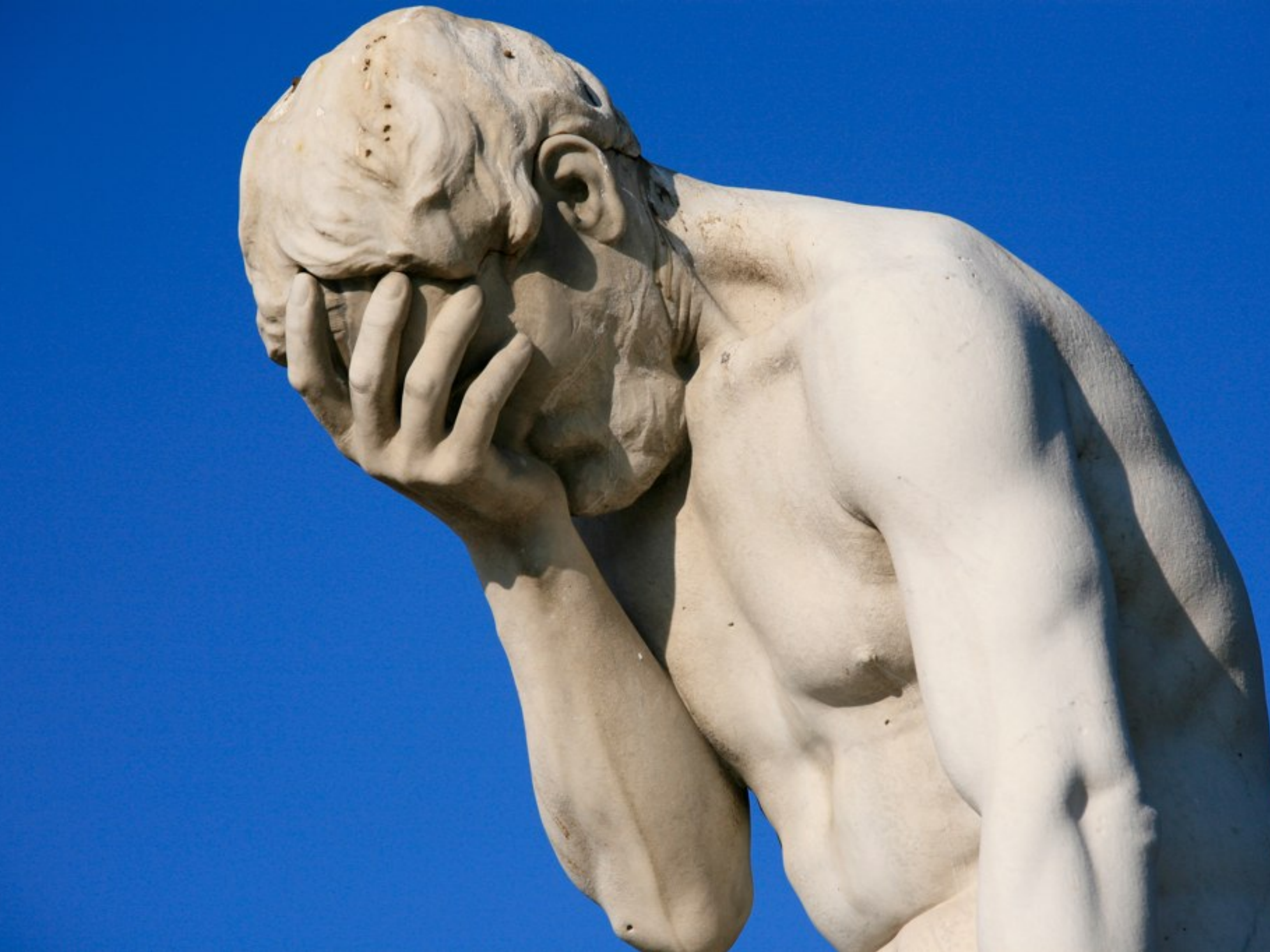
- For example, it may not be practical to manage root login using LDAP.
- In that case, you can block root login and use sudo instead.
- (This is an example in the PCI-DSS standard.)

This is not a Get-Out-of-Jail-Free Card.

- If you don’t need an external auditor, it’s between you, your conscience, and your Errors and Omissions insurance provider.
- External auditors have to sign off on compensating controls.
- Compensating controls need to be just as secure as the requirement they replace, in your particular environment.

By now, you are probably…

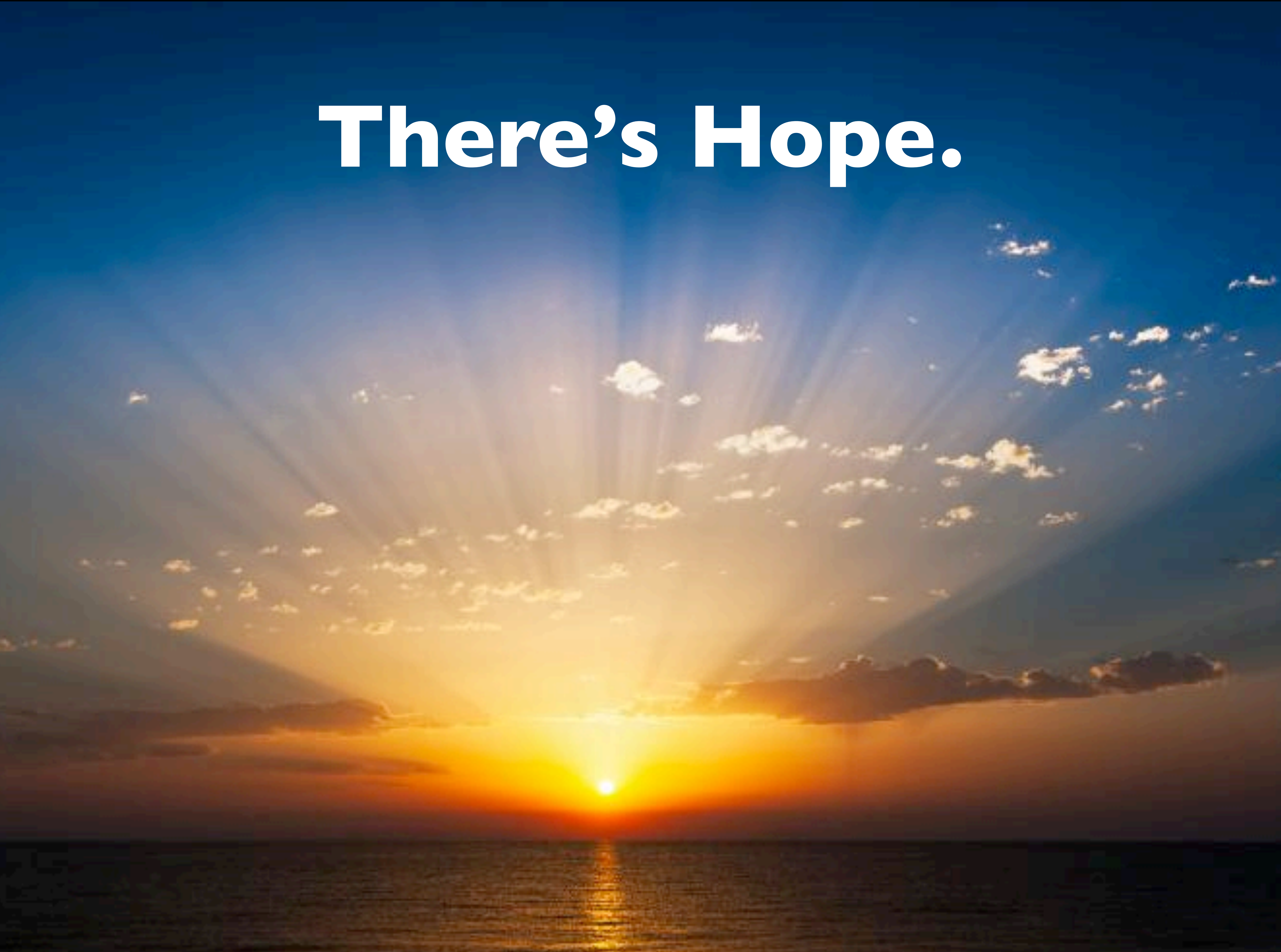
PGX
POSTGRESQL
EXPERTS, INC.



We're doomed.

- Full and correct PCI compliance is a lot of work.
- There's a huge downside risk.
- If there's a breach, you could be liable for every single penny of loss suffered by the banks and consumers.
  - Wait, you thought *banks* took risk? Ha.

There’s Hope.



There’s hope.

- If you don’t have to touch PANs, you can avoid (much of) PCI.
- First steps were services like PayPal, but not suitable for many environments.
- We’re finally getting a better solution:
- **Tokenization**.

Tokenization.

- Replaces the PAN with a token.
- The token is not considered a PAN, so (most of) PCI does not apply…
    - … as long as you never store the PAN, even temporarily.
- Transfers the PCI headache onto the tokenization API vendor.

PGX POSTGRESQL EXPERTS, INC.

Big Tokenization Gotcha.

- Some interfaces do not return the token without an authorization attempt.
- So, you need to do the authorization immediately, because if you store the PAN back into the database (even for a short time)…
- … you're back to PCI-Compliance-Land.

Tokenization Gateways.

- Braintree, Stripe, Cybersource, MasterCard…
- If you can integrate this into your system, it's much much better than dealing with PCI.
- So you can move on to worrying about…

HIPAA

Health Insurance Portability
& Accountability Act

But that’s a different talk.

Questions?

Christophe Pettus
@xof

thebuild.com
pgexperts.com

PGX

POSTGRESQL

EXPERTS, INC.

Thank you!

Christophe Pettus
@xof

thebuild.com
pgexperts.com