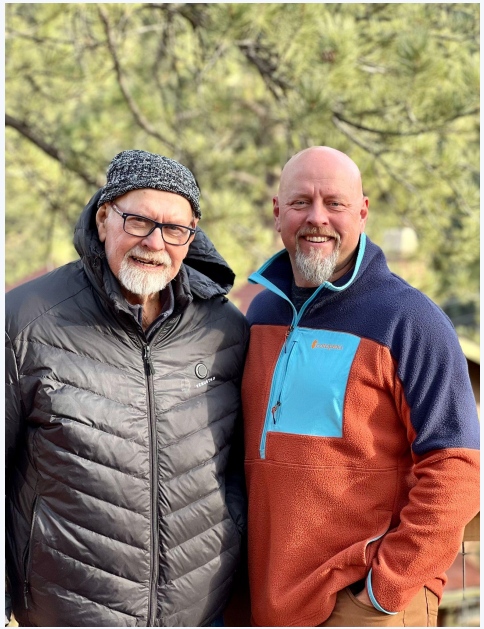
TRACEABLE

Identity In Security

Or, a problematic history in 3 acts

Richard Bird
Chief Security Officer
richard@traceable.ai

An Introduction



Just the son of a fishing boat captain – and this is The Captain and me

20 plus years in corporate roles
CISO, CIO

5 years - 3 successive C-level roles in security solutions
CCIO, CPO, CSO

Deep experience in banking, financial services, fintech and high tech manufacturing

Extensively quoted globally - Financial Times, NYT, WSJ, Dark Reading, Business Insider, etc.

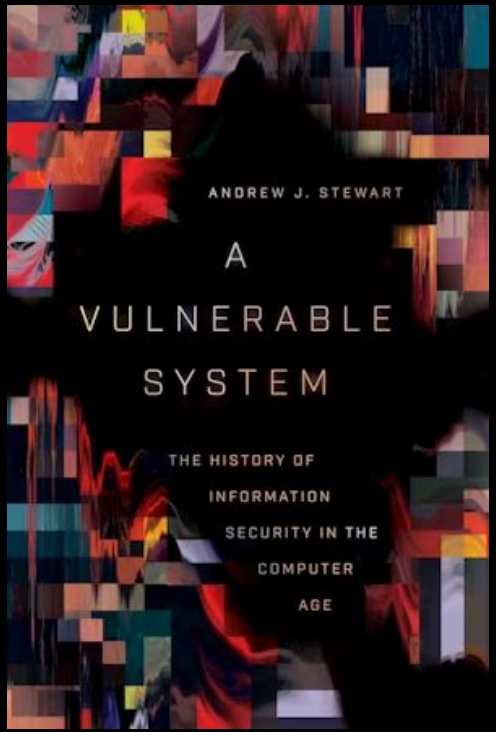
Directly involved with standards and efforts in digital identity, zero trust security, data privacy

1962

MIT & IBM

When it all started to go wrong…

“In the spring of 1962, (Alan) Scherr was looking for a way to bump up his usage time on CTSS. He had been allotted four hours per week, but it wasn't nearly enough time to run the detailed performance simulations he'd designed for the new computer system. So he simply printed out all of the passwords stored on the system.” - Wired 2012


ANDREW J. STEWART
A
VULNERABLE
SYSTEM
THE HISTORY OF
INFORMATION
SECURITY IN THE
COMPUTER
AGE

The 3 Acts

1. Access Administration
    a. Accounts and passwords
    b. Federation and SSO
    c. Directories
2. Access Control
    a. 2FA/MFA
    b. Step up/dynamic authentication
    c. Decentralized Identity/SSO
3. Layer 7 Access
    a. API access
    b. Fine grained control (AuthZ/AuthN)
    c. Elimination of implied/persistent trust

The 3 Acts

1. Access Administration
    a. Accounts and passwords
    b. Federation and SSO
    c. Directories
2. Access Control
    a. 2FA/MFA
    b. Step up/dynamic authentication
    c. Decentralized Identity/SSO
3. Layer 7 Access
    a. API access
    b. Fine grained control (AuthZ/AuthN)
    c. Elimination of implied/persistent trust

The Villains In Each Act

1. **Access Administration**
   Simply put - NOT security - but 80% to 90% of the enterprise world is still at this level. Either fully or partially.

2. **Access Control**
   Only “security” when it is implemented across the entire digital estate and is fully dependent on the quality of the implementation.

3. **Layer 7 Access Security**
   Requires major re-thinking of our current security architectures as well as substantial improvements in authorization plane security solutions.

Identity – Rarely First In Our Hearts

On its own, the open-source community does not have the leverage to enact necessary changes. This past year, two of the largest open-source registries announced that they will impose minimum security measures on "critical" projects, as defined by popularity.[312] Maintainers of "critical" projects, including hobbyists and paid developers, must secure their accounts with two-factor authentication ("2FA") to continue contributing to the project. This simple measure could prevent 99.9% of account-takeovers, a rising threat to open-source security.[313] However obvious this measure seems, the new mandate resulted in an outcry from the community—-authors of extremely popular projects threatened to abandon their posts, which could potentially break the systems of any end-user reliant on their projects.[314] With GitHub slated to roll out mandatory 2FA for all its developers by the end of 2023, we can expect more resistance.[315]

Forthcoming in the North Carolina Law Review

TRAGEDY OF THE DIGITAL COMMONS*

CHINMAYI SHARMA

Google, iPhones, the national power grid, surgical operating rooms, baby monitors, surveillance technology, and wastewater management systems all run on open-source software. Open-source software, or software that is free and publicly available, powers our day-to-day lives. As a resource, it defies economic logic; it is built by developers, many of whom are volunteers, who build projects with the altruistic intention of donating them to the digital commons. Developers use it because it saves time and money and promotes innovation. Its benefits have led to its ubiquity and indispensability. Today, over 97% of all software uses open source. Without it, our critical infrastructure would crumble. The risk of that happening is more real than ever.

In December 2021, the Log4Shell vulnerability demonstrated that the issue of open-source security can no longer be ignored. One vulnerability found in a game of Minecraft threatened to take down systems worldwide—from the Belgian government to Google. The scope of the damage is unmatched; with open source, a vulnerability in one product can be used against every other entity that uses the same code. Open source's benefits are also its burden. No one wants to pay for a resource they can get an unlimited supply of for free. Open source is not, however, truly unlimited. The open-source community is buckling under the weight of supporting over three-fourths of the world's code. Rather than share the load, its primary beneficiaries, companies that build software, add to it. By failing to take basic precautionary measures in using open-source code, they make its exploitation nearly inevitable—when it happens, they free-ride on the already overwhelmed community to fix it. This doom cycle leaves everyone worse off because it leaves our critical infrastructure dangerously vulnerable.

Since it began, open source has worked behind the scenes to make society better. Today, its struggles are going unnoticed and unaddressed. The private sector isn't willing to help—the few who are cannot carry the burden alone. So far, government

---

*Scholar in Residence at the Robert Strauss Center for International Security and Law and Lecturer at the University of Texas School of Law. I would like to thank Dean Robert Chesney, Adam Klein, Alan Rozenshtein, Charles Barzun, Orin Kerr, Peter Swire, Josh Goldfoot, Karen Copenhaver, Michael Dolan, Bruce Schneier, Jacques Chester, James Howison, and James Tomberlin for encouragement, advice, and thorough feedback on earlier drafts. This paper would not have been possible without the insight provided by members of the Linux Foundation, Apache Foundation, OSTIF, OpenSSF, OTF, Google, Microsoft, Chainguard, GitHub, GitLab, Global Cyber Alliance, and other open-source community members over the course of numerous interviews. In particular, I want to thank Allan Friedman, Trey Herr, Frank Nagle, CRob, Brian Behlendorf, David Wheeler, Jessica Wilkerson, Jacques Chester, Daniel Kahn Gillmor, Ari Schwartz, Melissa Hathaway, Amir Montazery, Madison Oliver, Tod Bearsley, Steve Weber, Justin Cappos, Bryan Nunez, Laurent Simon, William Bartholomew, Sean Brookes, Leslie Daigle, Brendan O'Leary, Jonathan Meadows, and Manoj Prasad. The largest thanks goes to Audrey Kim for excellent research assistance without whom this paper would not exist.

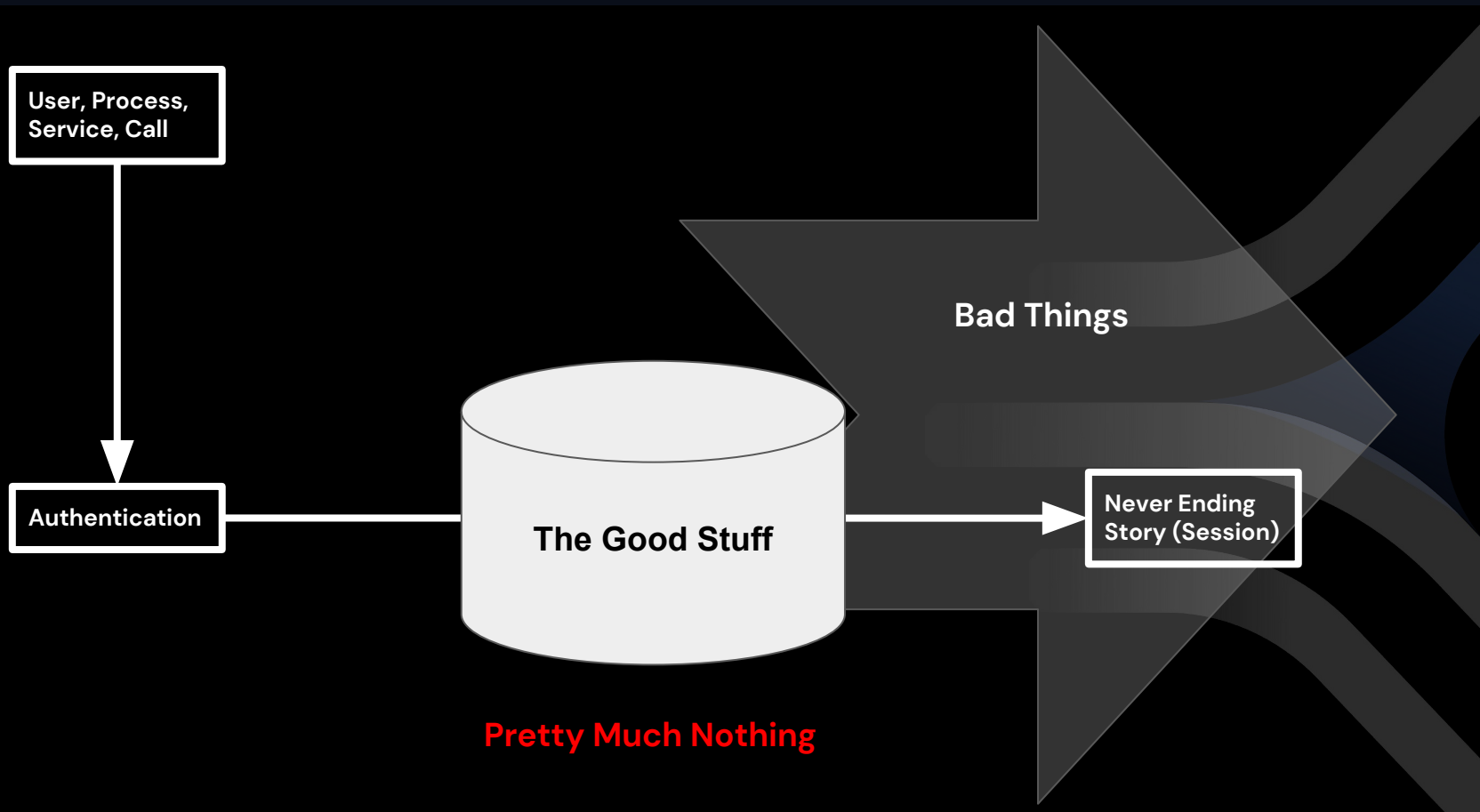
JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties. The claims in a JWT are encoded as a JSON object that is used as the payload of a JSON Web Signature (JWS) structure or as the plaintext of a JSON Web Encryption (JWE) structure, enabling the claims to be digitally signed or integrity protected with a Message Authentication Code (MAC) and/or encrypted.

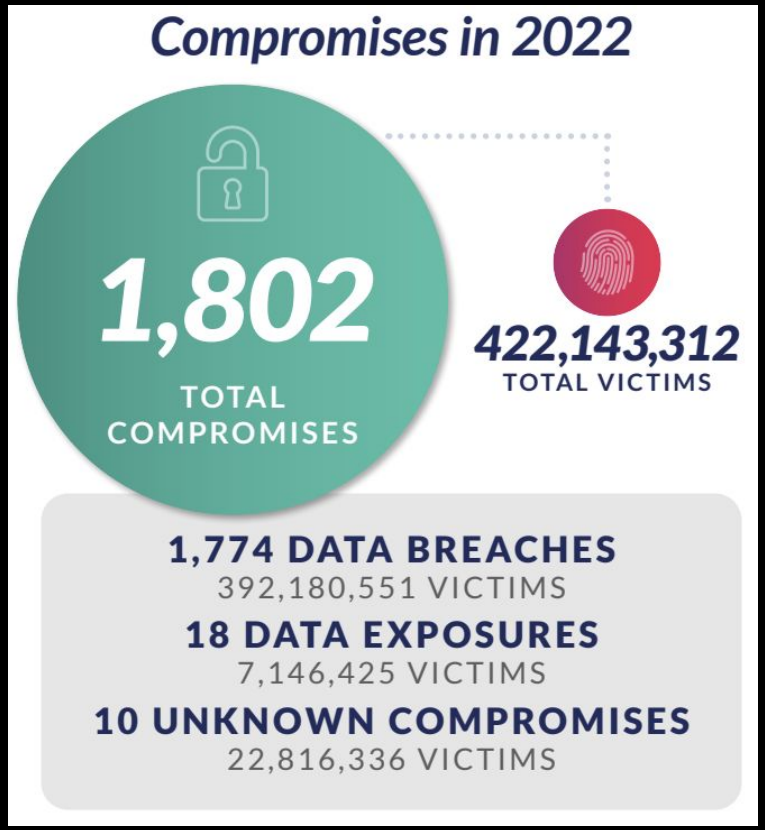
An Uncomfortable Truth

Encryption, Signatures, Claims, etc.

ARE Not Identity Security

(they are a very important PART of identity security)


User, Process, Service, Call
Authentication
The Good Stuff
Bad Things
Never Ending Story (Session)
Pretty Much Nothing



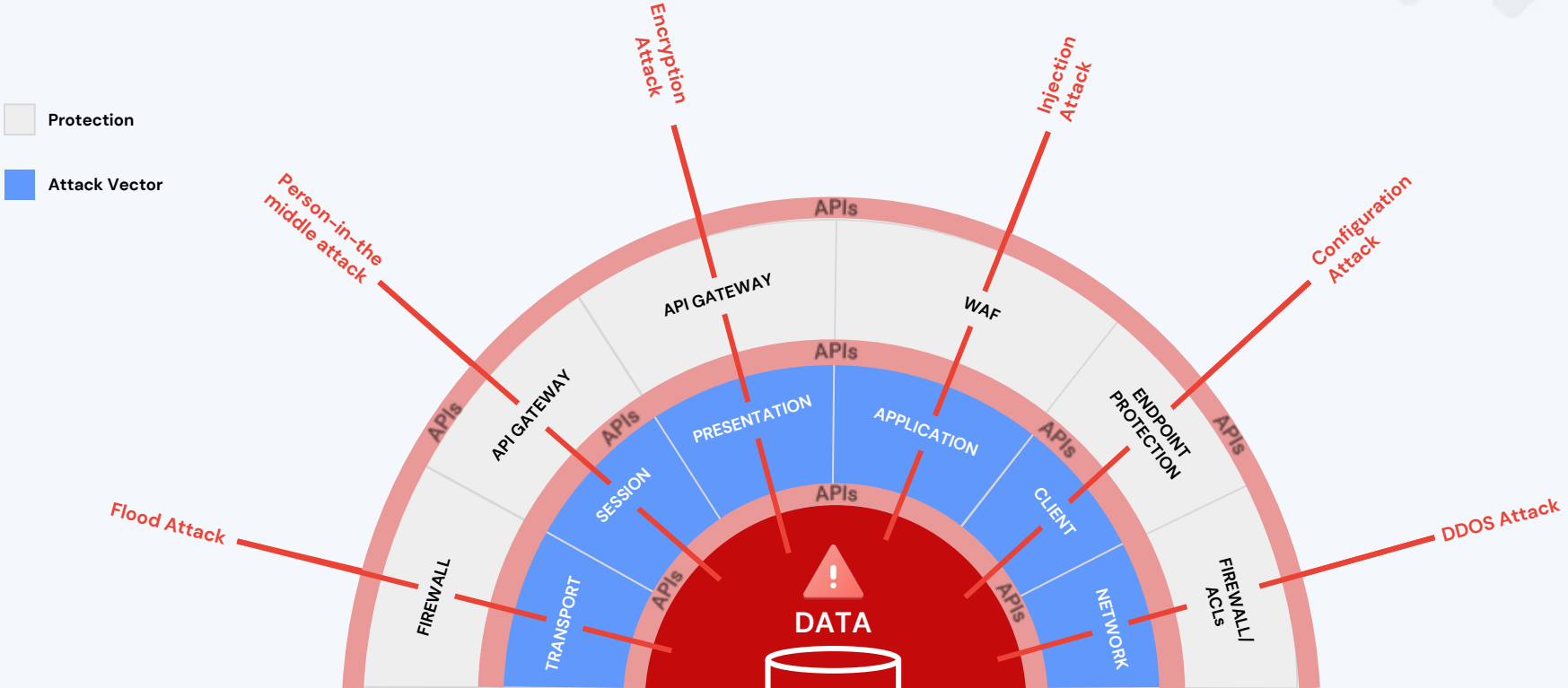
Top 10 Data Breach Attributes

Personally Identifiable Information (PII)	Compromises
Name	1,560
Full Social Security Number	1,143
Date of Birth	633
Current Home Address	565
Driver's License/State ID Number	499
Medical History/Condition/Treatment/Diagnonsis	465
Bank Account Number	443
Medical Insurance Account Number	370
Undisclosed Records	226
Medical Provider Account/Record Number	196

TRACEABLE

APIs are now the Universal Attack Vector

Every Attack Vector and Method In One Place (Layer 7)



TRACEABLE

Layer 7 API Access

Reduce Attack Surface

- API layer Adaptive Rate Limiting with insights from API Data lake
- Ensure least privilege: detect and block AuthZ, mass assignment, and excessive data exposure vulnerabilities
- Identify high risk API endpoints for priority remediation

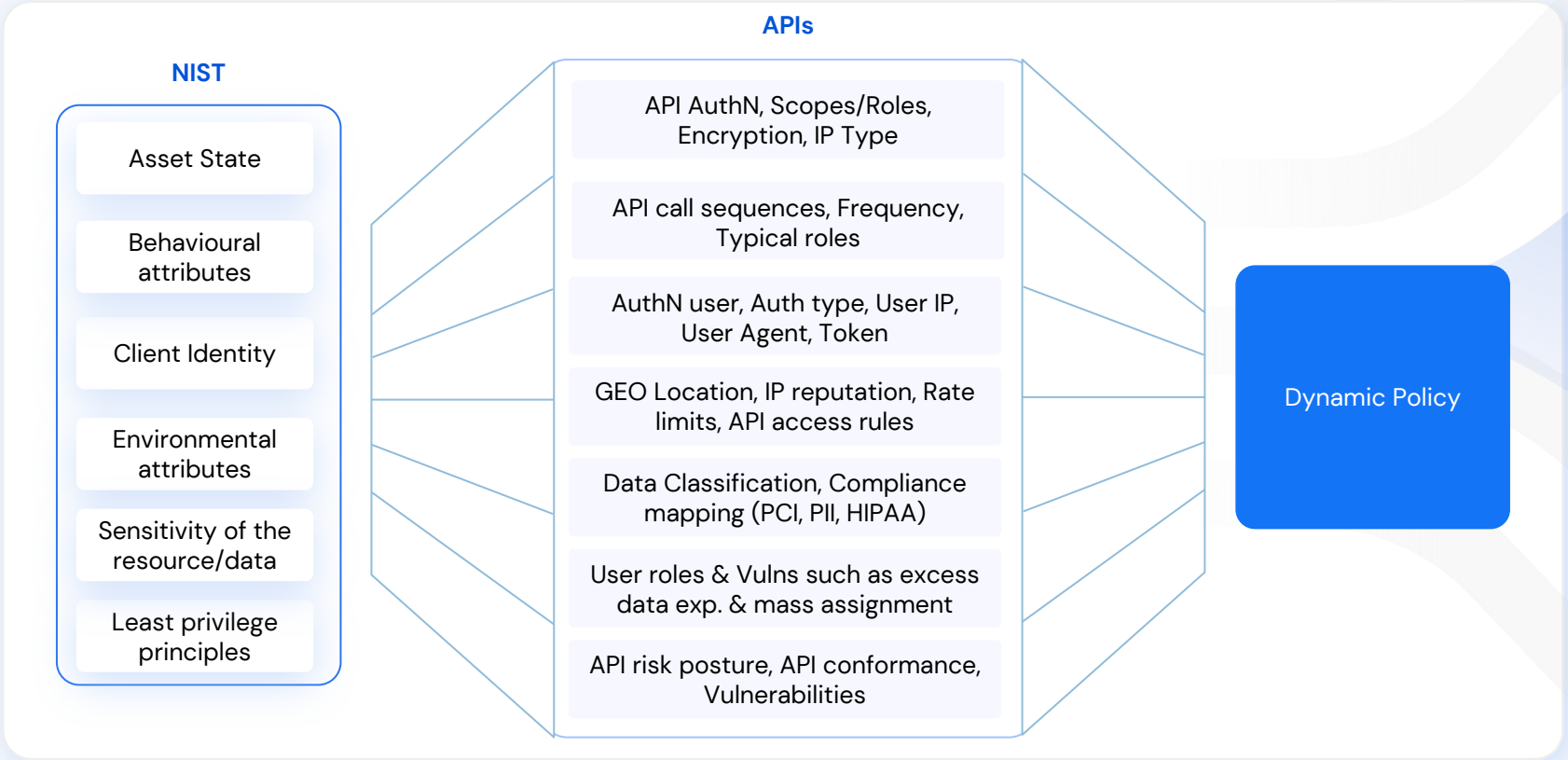
Continuous Verification

- User attribution and AuthN/AuthZ are key in determining API Access
- Source asset risk identification (IP reputation, IP type, geo-location, etc)
- Right access for right users and entities at the right time

Stop Data Breaches

- **D**ynamic data access controls
- Quickly and easily create policies with out-of-the-box templates or customize policies
- Policies catered to organization needs, compliance and data access patterns.

Dynamic Policy Considerations for Layer 7 Access



Many thanks W3C CCG!

Richard Bird
Chief Security Officer
Traceable
www.linkedin/in/rbird

Many thanks W3C CCG!

Richard Bird
CSO
Traceable