Selected Submission – Black Hat USA 2023

Title Smashing the State Machine: The True Potential of Web Race Conditions
Speaker Name James Kettle
Tracks Application Security: Offense; Exploit Development
Format 40-Minute Briefings

Why Did You Select the Above Track(s) for your Submission?

- This talk introduces new classes of web race-condition attack, and explores how to detect and exploit them. As such, I think it's a good fit for the 'Application Security: Offense' track.
- For the secondary track I choose 'Exploit Development' as techniques like the single-packet attack are valuable for exploit reliability under variable network conditions, and may be adapted for memory-safety attacks in future too. 'Application Security: Defense' would be a viable alternative secondary track.

Abstract

- For too long, web race-condition attacks have focused on a tiny handful of scenarios. Their true potential has been masked thanks to tricky workflows, missing tooling, and simple network jitter hiding all but the most trivial, obvious examples. In this session, I'll introduce multiple new classes of race condition that go far beyond the limit-overrun exploits you're probably already familiar with.
- Inside every website lurks a state machine: a delicately balanced system of states and transitions that each user, session, and object can flow through. I'll show how to fire salvos of conflicting inputs to make state machines collapse, enabling you to forge trusted data, misroute tokens, and mask backdoors. These exploits will be demonstrated across multiple high-profile websites, and a certain popular authentication framework.
- These techniques unveil so much fresh attack-surface, it can be hard to know where to focus your testing. To help, I'll share a polished methodology for efficiently pursuing leads, automating complex attacks, and quickly ruling out dead ends. You'll learn to recognise high-risk patterns and eke out subtle tell-tale clues to scent blood long before sacrificing anything to the RNG gods.
- To defeat jitter and make these attacks reproducible, I've taken lore amassed over years of research into HTTP Desync Attacks and applied it to develop precision tooling. You'll learn how to adapt your attacks to different HTTP versions and target architectures, abusing protocol-level design decisions and obscure implementation quirks in popular servers. This includes a strategy that can squeeze 30 requests sent from Melbourne to Dublin into a sub-1ms execution window. Alongside the open source tool, we'll also release a full complement of free online labs to the Web Security Academy, so you can try out your new skillset immediately.

Presentation Outline

<u>Brief outline</u>

- Introduction
- Methodology
    - Fundamentals & attack potential
    - Tackling jitter with the single-packet attack
    - Tackling scope explosion
- Real-world application, with case studies and a live demo
    - Object masking attacks
    - Multi-step style exploits on single-step processes
    - Parameter-split attacks on single endpoints
    - Single-request attacks
- Promising research avenues
    - Attack scenarios
    - Tooling enhancements
- Defence
- Takeaways + Q&A

Detailed outline

- I'll open with a brief story of how this research was inspired by a blog post from 2016 that initially defied comprehension, and a technique from an academic talk in 2020 that I suspected had hidden potential.
- To lay the foundation, I'll give a brief overview of race condition fundamentals, using an unpatched vulnerability in Google's ReCAPTCHA as a case study. This will show how virtually all documented attacks have a 'limit-overrun' objective - with one exception from 2016.
- I'll analyse the 2016 attack - an accidental discovery that could cause email confirmation tokens to be routed to the wrong address - and draw out the key implication: everyone knows multi-step sequences are a hotbed for business logic flaws, but with race conditions, everything is multi-step.
- Next, I'll show how to overcome two barriers that have left advanced race condition vulnerabilities overlooked for decades.
- The first barrier is network jitter, which I've overcome by developing the single-packet attack. I'll explain how it works in depth, then use a benchmark attack from Melbourne to Dublin to show how it seriously outperforms alternatives like last-byte sync. I'll also introduce server-side connection warming; a technique for overcoming heavy delays caused by a common style of reverse-proxy implementation, which can easily be mistaken for an impervious defence.
- The second barrier is the scope explosion problem; testing a single pair of endpoints often requires multiple attempts, custom automation and attack tuning so it's simply not practical to exhaustively test every possible combination of endpoints in a moderately sized application. I'll share a high-level overview of a practical, clue-based, black-box methodology designed to tackle this.
- In the next section, I'll use case studies to demonstrate the methodology being applied and adapted for three new classes of attack. These examples will also show how to apply different aspects of the methodology such as staggering requests, automating exploitation attempts, and coaxing out clues.
- I'll start with object masking attacks, which I'll demonstrate with a little vulnerability in Gitlab, where a race condition makes it possible to create high-privilege invitations that are invisible to administrators.
- The second attack class is hitting single-step processes with multi-step-style exploits, by invoking mutiple endpoints simultaneously. I'm hoping to illustrate this with a racy variation on the classic edit-basket-during-checkout exploit. I'll also use an exploration of Gitlab's email confirmation process to demonstrate the importance of staggering request salvos to compensate for different endpoints' internal workloads. This is a significant departure from classic race condition exploitation, which is always about making requests arrive as close to each-other as possible.
- The third attack class is the most powerful and counterintuitive. I'll show how you can exploit a single endpoint in isolation by feeding it simultaneous requests with different parameters. Using the conference WiFi, I'll live-demo the discovery of a spectacular finding in Gitlab, which tricks it into misrouting email confirmation codes, enabling attackers to gain admin access and also compromise third-party applications via OpenID.
- For the final case study, I'll use a token-misrouting vulnerability in a xxx (redacted) to demonstrate a subtle variant of the single-endpoint attack that's fully self-contained, so you can exploit it using a single request.
- From there, I'll explore and analyse a range of other attacks that I wasn't able to find live examples of during this research. I'll explain the reasoning that makes me confident they're out there somewhere, and where and how the audience might find them. This includes 'glimpse' attacks where an app briefly travels through an insecure middle state. One place this can happen is during object construction, such as when a user account is being registered and the password field hasn't been set yet. Other attack types include races based on data stored in the user session, and session-fixation style races where an attacker action coincides with a victim action.
- After that, I'll lay out some ideas for further tooling enhancements. These include finding ways to fit even more requests into a single TCP packet, and triggering server-side delays to make the single-packet attack compatible with staggered requests. Time permitting, I'll also share key lessons learned from a research perspective; what I'd do differently if I tackled this research again.
- Then, I'll evaluate a broad range of defences and mitigations, from locking and database-transactions to pushing state client-side with JWT.
- Finally, I'll wrap up with key takeaways, leaving 5 minutes for questions.

What new research, concept, technique, or approach is included in your submission?

- New classes of race-condition attack that use conflicting endpoints & inputs to break websites' state machines, with consequences such as security tokens being emailed to the wrong address.
- An efficient methodology for discovering these vulnerabilities without exhaustively testing every possible combination of endpoints in the entire application. One core insight within is a focus on provoking clues that racy behaviour is present, rather than going straight for a successful exploit. On one target, a clue was evoked after 3 probes but a successful exploit took 348 attempts.
- The single-packet attack; a technique that combines request fragmentation with HTTP/2's concurrency features to complete up to 30 requests with a single TCP packet, eliminating the effect of network jitter and reliably squeezing them into a 1ms execution window on cross-continent attacks.

Takeaways

1. Everyone knows multi-step sequences are a hotbed for business logic flaws, but with race conditions, everything is multi-step.
2. Treating everything as multi-step creates an overwhelming scope explosion, but this can be tamed with an efficient, clue-focused methodology.
3. The single-packet attack means that as HTTP/2 adoption grows, there is no 'remote' when it comes to race conditions.

If applicable, what problem does your research solve?

- The single-packet attack solves the problem of people missing race condition vulnerabilities due to network jitter. This is especially cool because it makes this attack class vastly more accessible to people in countries with poor internet infrastructure.
- The methodology partially solves the scope-explosion problem (people failing to find non limit-overrun vulnerabilities, due to exhaustively testing all possible endpoint combinations being impractical)
- Finally, I believe people are nervous about looking for non-trivial web race conditions because they're notoriously hard to identify, understand and replicate. Through the combination of methodology, tooling, labs and proven examples, I'm aiming to inspire attendees to overcome this fear and find these bugs.

Will you be releasing a new tool? If yes, is the tool open source or commercial?

- Open-source. The single-packet attack will be implemented in a major update to Turbo Intruder, which is an open-source Burp Suite extension. This update also enhances other features to better support the race condition hunting methodology. Turbo Intruder is fully compatible with the community edition of Burp, and already the most popular extension.
- https://github.com/PortSwigger/turbo-intruder
- The single-packet attack is also straightforward to reimplement using the details I'll be sharing - please refer to the 'message for review board only' for further info.

If selected, who will present the session?

James Kettle

If the speaker(s) listed above are previous Black Hat Briefings speakers, when/where did they speak?

Black Hat USA 2022, 2021, 2020, 2019, 2018, 2017, 2015; Black Hat Europe 2021, 2019, 2016

So the Review Board is able to get a sense of their presentation style(s), please provide a link to a video sample.

https://www.youtube.com/watch?v=N54VevVxrgQ (live, 2022)
https://www.youtube.com/watch?v=w-eJM2Pc0KI (live, 2019)
https://www.youtube.com/watch?v=rHxVVeM9R-M (pre-recorded, 2021)

Message for Review Board Only

- As mentioned in the introduction, this research was inspired by a blog post from 2016. I've never seen a similar finding published in the 7 years since, possibly because it was an accidental discovery that didn't even make

sense to the author, leading to people, including myself, writing it off as a freak occurrence and focusing on the other techniques in the post. You can find it here: https://www.josipfranjkovic.com/blog/race-conditions-on-web

- The single-packet attack was inspired by the 2020 USENIX presentation 'Timeless Timing Attacks'. In that presentation, they place two entire HTTP/2 requests into a single TCP packet, then look at the response order to compare the server-side processing time of the two requests. The single-packet attack fragments and transmits up to 30 requests, withholding a zero-sized DATA frame from each, then issues a single TCP packet to complete them all simultaneously. This makes it possible to launch race condition attacks that are unaffected by network jitter. The sheer number of simultaneous requests this enables works really well with the 'clue-based' approach used in the methodology.
- I've implemented the single-packet attack in the open source tool Turbo Intruder. I'll also share all essential implementation details, including a trick to make the operating system's network stack batch the correct frames via Nagle's algorithm, which means it doesn't require a custom TCP or TLS stack. This means that anyone who prefers not to use Turbo Intruder can easily make their own implementation in a high-level language of their choice.
- The email-confirmation vulnerability in Gitlab partly originates from Devise, which is a very popular Ruby on Rails module for authentication. Whether individual systems built with Devise are exploitable depends on their implementation. I hope to see this patched properly in Devise itself by the time the presentation lands.
- This presentation has fewer case studies than usual for me because automated detection of these vulnerabilities is not practical. On the plus side, this leaves a large number of vulnerabilities on the table that the audience can find simply by applying the methodology. I've just found a way to create an account-data inconsistency in a major social network, making different endpoints disagree about what my email address is, and I'm exploring ways to turn this into useful exploit for a case study.
- This talk will come with a printable whitepaper. We'll also help attendees gain practical experience applying the techniques shared in this presentation by releasing free online labs. For reference, you can find our labs for last year's browser-powered desync attacks research here: https://portswigger.net/web-security/request-smuggling/browser
- https://portswigger.net/web-security/request-smuggling/browser/client-side-desync
- Please let me know if you have any questions. I'd be happy to share more details on the methodology or anything else.

Selected Submission – Black Hat USA 2023

Title Nothing but Net: Leveraging macOS's Networking Frameworks to Heuristically Detect Malware

Speaker Name Patrick Wardle

Tracks Network Security; Malware

Format 40-Minute Briefings

Why Did You Select the Above Track(s) for your Submission?

- The network security track makes sense for a talk about creating network monitoring tools.
- Also, as a secondary track "malware" makes sense, as the goal of the networking tools is to detect malware.

Abstract

As the majority of malware contains networking capabilities, it is well understood that detecting unauthorized network access is a powerful detection heuristic. However, while the concepts of network traffic analysis and monitoring to detect malicious code are well established and widely implemented on platforms such as Windows, there remains a dearth of such capabilities on macOS.

This talk aims to remedy this situation by delving deeply into a myriad of programmatic approaches capable of enumerating network state, statistics, and traffic, directly on a macOS host. We will showcase open-source implementations of relatively overlooked low-level APIs, private frameworks, and user-mode extensions that provide insight into all networking activity. And, by leveraging these techniques, you will learn how to efficiently and generically detect both known and unknown threats targeting macOS!

Presentation Outline

/* Part 0x0: Introduction */

In this (brief) part of the talk, we will highlight:
A. The importance of detecting unauthorized network access in the context of malware detection (especially noting the fact that host-based (vs. on the network) approaches provide the responsible process).

We'll also give specific examples of recent macOS malware that accesses the network …ranging from payloads used in 3CX supply chain attack, to previously undetected implant updaters.

B. Explanation of the lack of network traffic analysis and monitoring capabilities on macOS. Showing this is due to a few reasons, such as changes at the OS level (e.g. deprecation of network kernel extensions), lack of documentation of newer approaches, or even the fact that Apple has chosen to keep certain frameworks and APIs private.

/* Part 0x1: Network State and Statistics */
We'll then dive in, here, in this part of the talk focusing on methods that provide a snapshot of the network (including listening sockets, network connections, traffic stats, and more).

A. Enumerating Network State (per process) via proc_pid APIs.
Starting with proc_pidinfo & PROC_PIDLISTFDS to get a list of all file descriptors currently opened by a process, then focusing on file descriptors whose type are PROX_FDTYPE_SOCKET.

B. Enumerating Network State and Statistics via the private NetworkStatistics framework
We'll show how to extract APIs and symbols from the framework, and then via (private) APIs such as NStatManagerCreate, NStatManagerQueryAllSourcesDescriptions, show how globally enumeration the network state and statistics such as bytes transmitted down/up.

/* Part 0x1: Network Monitoring */
In this part of the talk, we'll focus on approaches that provide real-time monitoring capabilities of the network, on macOS.

A. Introduction to System Extensions and the Network Extension Framework
…including the entitlements required, how to craft an application bundle (containing an extension), and then how to install & activate your extension.

B. Creating a DNS monitor (com.apple.networkextension.dns-proxy)
After highlighting the benefits of host-based DNS monitoring (e.g. it provides a light-weight approach to detecting malware resolving domains, etc etc), we'll dive into building a DNS monitor, atop Apple's Network Extension Framework.

(Fun fact, I've been working on an open-source implementation of a DNS monitor, see https://github.com/objective-see/DNSMonitor), for the last year or so. AFAIK this is (was?) the first open-source implementation using macOS' new(ish) Network Extension Framework. Lots of lessons learned and comms w/ Apple engineers as their documentation had some inconsistencies and was incomplete. This open-source tool will act as a comprehensive case-study for this section of the talk).

As DNS is predominately sent over UDP, in this talk we'll focus solely on UDP flows, whose type will be NEAppProxyUDPFlow. We'll show how to our DNS monitor must:

1. Open the flow, with the local endpoint
2. Read from the flow
3. Open the remote endpoint
4. Send the read datagrams (from step #2) to the remote endpoint
5. Read any response from the remote endpoint
6. Write any response from the remote endpoint to flow

At step two, we'll have a list of DNS packets (likely a DNS "question") while at step five, we'll have another DNS packet (likely a DNS "answer" to the "question").

We'll also show how the process responsible for the DNS request (e.g. malware) can be identified from the flow's audit token.

C. Comprehensive Network Monitoring (com.apple.networkextension.filter-data)
Next, we'll show how to build a full network monitor or filter, (again, atop the Network Extension Framework). We'll show how to filter on all protocols and either only outgoing traffic (or incoming as well).

Using this (and information from the NEFilterSocketFlow), we'll show how a fully featured firewall can be built.

D. Tools vs. Malware
We'll then pit various malware specimens against the tools and techniques presented in this talk. For example, we'll show how the simpler "snapshot"-based approaches can reveal listening sockets tied to reverse-shells. Of course, we'll also show how the network monitors (DNS and full protocol) can flag implant exfiltration, C&C tasking, and more.

Note that for each tool, we'll provide a full open-source implementation!

(Fun fact two, I've submitted the networking tools I've been building and covered in this talk, to BH Arsenal. If this talk and the Arsenal session get accepted, attendees could be able to both come to this talk to get a conceptual understanding of the tools and their approaches, as well as then come to the Arsenal session to try them out in a hands-on manner).

E. Conclusions / Takeaways
We'll wrap up the talk, with a brief discussion of the importance of incorporating these capabilities into cybersecurity strategies for macOS environments

We'll also summarize the talk's key points and the tools presented.

What new research, concept, technique, or approach is included in your submission?

- Surprisingly there hasn't been a lot of talk / research on host-based network monitoring on macOS (especially in the context of malware analysis). This is maybe because Windows has long been the dominant OS in the enterprise and/or a lot of network monitoring is done off host (e.g. on the network). Now Macs are super prevalent in the enterprise, and host-based monitoring has many advantages (access to unencrypted data, ability to identify the responsible process, etc etc). Thus, the majority of the content in this talk is new …or at the very least, has not been widely discussed nor understood.

Takeaways

1. Host-based network monitoring is a very efficient and powerful approach to detecting malware on macOS.

2. macOS supports a wide range of mechanisms to build host-based network tools (though some are found within private frameworks/APIs or are not particularly well documented).

3. The approaches and open-source tools presented in this talk can (and should be) leveraged to build enterprise-grade EDR products for macOS …bringing parity with Windows tooling.

If applicable, what problem does your research solve?

- The dearth of information regarding networking monitoring / detection tools on macOS.

Will you be releasing a new tool? If yes, is the tool open source or commercial?

- Yes(ish). I've been working on the DNS monitor - and will be releasing a new (and fist non-beta) version with a host (ha!) of new features / capabilities such as DNS filtering / blocking, etc. etc.
- It's open-source (GPL v3)

If the speaker(s) listed above are previous Black Hat Briefings speakers, when/where did they speak?

- BH USA 2020 https://www.blackhat.com/us-20/briefings/schedule/#office-drama-on-macos-20854
- BH USA 2021: https://www.blackhat.com/us-21/briefings/schedule/#armd-and-dangerous-23772
- BH USA 2022: https://www.blackhat.com/us-22/briefings/schedule/#dj-vu-uncovering-stolen-algorithms-in-commercial-products-27673

So the Review Board is able to get a sense of their presentation style(s), please provide a link to a video sample.
My talk on exploiting Zoom 0days at DefCon last year was recently posted: https://youtu.be/jk1OMo8Gcsk?t=18

Message for Review Board Only

- Thank you for considering my submission!
- I'm a passionate Mac malware analyst (virologist?) and have recently spent a lot of time diving into network-based detection, directly on the host.
- I'd be stoked to share my findings with the BlackHat attendees and provide actionable takeaways so we're all better prepared to both detect & thwart malware targeting macOS!

Selected Submission – Black Hat USA 2023

Title	Making and Breaking NSA's Codebreaker Challenge
Speaker Name	Rita Doerr (Academic Outreach Lead, NSA)
Track	Community & Career
Secondary	N/A
Format	40-Minute Briefings

Why Did You Select the Above Track(s) for your Submission?
Since its inception in 2013, NSA's Codebreaker Challenge has influenced the post-secondary and secondary cybersecurity education communities as both a curriculum-influencing activity as well as a hiring and recruiting event for NSA. Given this, the Community and Career track seems quite fitting for this presentation.

Abstract
For the past 10 years, the National Security Agency (NSA) has hosted the Codebreaker Challenge. This competition, written and run by NSA's technical experts, includes a new theme and challenge each year. The challenges are custom designed by NSA to mimic a real-world problem NSA faces in its mission and include up to 10 rounds of increasing difficulty including code analysis and reverse engineering. Starting with only 5 schools in 2013, the program has grown exponentially. In 2022, more than 4,800 participants from 449 schools across the United States competed for bragging rights. Only 104 people (about 2%) successfully completed all 9 rounds of the challenge this past year. For the very first time at Black Hat, the NSA will publicly share insights and lessons from running the Codebreaker Challenge. We will present details about the design of the 2022 challenge that focused on NSA assisting FBI agents in responding to a US company that had been crippled by a ransomware attack.

Presentation Outline
INTRODUCTION --
Speaker Bio. Dr. Rita Doerr is the NSA Cybersecurity Lead for Academic Outreach. She has more than 39 years' experience at NSA, is an adjunct professor, and a Ph.D. in computer science, both from the University of Maryland, Baltimore County. She has a passion for cybersecurity education and outreach.
Why is NSA doing a cyber competition? NSA makes and breaks codes. We want to educate and excite students about this field to advance the field and the next generation of professionals. Recruitment is a secondary goal.

What outcomes have come from hosting Codebreaker? Schools are using Codebreaker as mandatory and optional course content.

GOALS OF CODEBREAKER CHALLENGE –
How is this different than other Capture the Flag (CTF) challenges such as CISA President's Cup and the National Collegiate Cyber Defense Competition (NCCDC)? NSA is unique among U.S. government agencies for its mission and authorities in cybersecurity and foreign signals intelligence. As part of the Codebreaker Challenge, NSA desires its participants to use Ghidra, NSA's software reverse engineering tool publicly released in 2019.

Who are the participants? U.S.-based academic institutions. Initial launch in 2013 had 5 participating post-secondary schools and 2022 had 449 (2021 had 632). 2017 marked the first year that Codebreaker saw the participation of high school students. In 2022, 40+ high schools/districts participated and in 2021, there were 60+. Georgia Tech was recently declared the champion for 2022.

DESIGNING THE CHALLENGE --
Each year, NSA experts engage in open-ended brainstorming that iteratively coalesces towards a more cohesive story over a week or two to create the "backstory" and tasks. Then, a team of volunteers implements the challenges over several months for a fall semester kickoff. Aligning themes with real-world and NSA care-abouts, the challenge is designed to show how NSA works collaboratively across government and industry to secure critical networks.

The 2022 Event. The 2022 event challenged participants to investigate a fictitious compromise of a U.S. company that had been crippled by a ransomware attack. NSA was standing by to provide technical assistance to FBI agents responding. Participants had to find the attacker's identity, identify the tools that they used to carry out their attack, and investigate a Ransomware-as-a-Service (RaaS) website used by the attacker which involved finding and exploiting vulnerabilities to recover the victim's files. Participants were asked to investigate the victim's network (e.g., Which user account was compromised?) and to recover the attacker's tools and discover their identity. Next, they had to investigate and gain access to the RaaS site. Lastly, they had to recover the victim's keys by escalating privileges to an administrator account, finding the key-encrypting-key, and then finally recovering the victim's files.

Evolution over time. Every year uses a fictitious story to provide a realistic context. Tasks have included blockchain analysis, smart contract development, message spoofing, user masquerades, and message decryption. Past challenges have also required participants to reverse engineer and develop exploitation capabilities against a custom Android secure messaging app (TerrorTime -- fabricated for the challenge) to discover and thwart future attack plans. Over the years, NSA has increased the number and complexity of challenges from 4 to 10. In 2021, participants spent a full 5 months working on the challenge. During that time NSA offered virtual lectures, tips, and tools to help solve the problems, and regularly engaged with participants using Discord chat.

LESSONS LEARNED FROM PARTICIPANTS –
Participants have developed sharper technical skills in reverse engineering, exploit development, software development, forensics, and analysis.

"The Codebreaker Challenge had a really big impact on my career path." said one computer science undergraduate who went on to the Scholarship for Service (SFS) program.

Successful solvers have later gone on to work at NSA, after also successfully completing NSA's competitive hiring process. Since its inception in 2013, 200+ final job offers have been extended to Codebreaker Challenge participants.

FUTURE OF CODEBREAKER --
No end in sight ...this session will present the backstory and introduce the 2023 challenge.

Will this content be 100% new, never before published/presented by the time of Black Hat USA? NO

If no, what's new and what's already been published/presented and where?
In June 2022, Dr. Doerr presented an abbreviated version of this talk to the National Initiative for Cybersecurity Education (NICE) Conference in Atlanta, GA. That audience is primarily educators and not cybersecurity practitioners.

What new research, concept, technique, or approach is included in your submission?
This is the only cyber challenge designed and run by the NSA. Codebreaker is unlike any other cyber competition given its focus on reverse engineering and code breaking.

Takeaways
1. Attendees can use results from reverse engineering, forensics, and software development to build their own education programs and to capitalize on topics that generate excitement.
2. Attendees will takeaway lessons that NSA has learned from running Codebreaker for 10 years and apply that knowledge to optimizing their own internal training and education programs.
3. Attendees will be able to describe how and why to participate in Codebreaker 2023 as a teacher, participant, or advocate.

SAMPLE SUBMISSION

Title: Government-Mandated Front Doors?: A Global Assessment of Legalized Government Access to Data
Submitted by: Andrea Little Limbago, Interos (Black Hat USA 2021)

Abstract (Provide a concise, yet detailed description of your presentation - 300 word maximum):
Who needs a backdoor when front door access is required? From Tesla to the U.S. tech giants, there has been growing focus on whether private sector companies are obliged to turn over data to a foreign government in exchange for market access. This can take the form of source code reviews to unfettered access upon request and increasingly may pose a risk to intellectual property and personal data as digital authoritarian frameworks proliferate.

This comes at a time when significant supply chain disruptions have prompted many in the private sector to reassess their global footprint, with cybersecurity a top priority and motivator when exploring greener pastures elsewhere. Integrating government data access policies must become core to these considerations as corporations reshore and transform their global footprint.

But how do these policies compare from one country to the next? Has the GDPR inspired more progeny or is the Chinese model spreading faster as many contend? To address these questions, this presentation will introduce a new global index of countries based on government-mandated data access requirements. We will discuss the data and factors driving the index, as well as elicit community recommendations for improving the model. With such significant global transformations underway, government-mandated data access warrants greater attention when exploring the full range of global cyber risks.

Notes:
•Detailed, yet concise abstract
•Defines a problem and offers a solution(s) that will be examined during the session.

Presentation Outline (Show the progression of your presentation.)
I. Introduction
a. Common adage that tech outpaces policy, but there are significant policy shifts across the globe
b. With the growing Splinternet/Balkanization of the internet and digital and physical supply chains, I was curious how this was playing out with regard to government-mandated data access. Are companies reshoring from one high security risk location to another when it comes to government access to data?
c. Are Russian mandatory source code reviews and Chinese local data storage and security reviews an anomaly or spreading to other regimes? What about Australian backdoor legislation or data protection similar to that found in the GDPR?
d. Building on and integrating the growing literature on encryption, cross-border data flows, and the techno-dictator/digital democracy archetypes, I created a global ranking of government access to data. This talk will share the findings and seek collaboration and feedback to continue to help improve the index and highlight this shifting landscape.

II. Shifting regulatory landscapes in context -
a. The rise of data protection laws and individual data rights - GDPR and its global offspring
b. Data sovereignty - growing requirements to store data locally and/or provide government access to data
c. The various motivations -> national security largely the underlying justification achieved through various means
d. Variations in implementation with examples - ranging from mandated certificates and unfettered access to transparent and limited access with a warrant

III. Why this matters for supply chain resilience
a. Quick overview of the stats and growth of reshoring and onshoring over the last few years

IV. The Model Framework
a. Rankings coded and based on a range of factors:
1) Degree of data localization and storage requirements
2) Degree of legalized government access -> from no access to unfettered access
3) Transparency of government access
4) Joint venture requirements for a local presence
5) Source code access requirements
6) The existence of a federal data protection law and independent data protection authorities

V. The Findings
a. Global country rankings
b. Interesting trends or non-intuitive results
c. All summarized in a white paper

VI. Next Steps
a. Gather feedback from the community for improvement
b. Iterate on the analysis by end of the year
c. Motivate broader thinking about the range of cyber risks and data protection considerations as globalization transforms

Notes:
•Clearly conveys progression of talk.
•Helps the Review Board visualize the presentation in its entirety.
•Gives an explanation of each area of the presentation.

Attendee Takeaways (What are three actionable takeaways included in your presentation?)
1) Government-mandated access to data poses a growing risk across the globe
2) Rethinking cyber risk to include government policies is increasingly necessary and introduces new risks to IP and PII
3) As supply chains transform and reshore, government data policies should be a top cyber consideration when assessing locations

Notes:
•Submitter fulfilled requirement of providing 3 takeaways
•Explains relevance to the audience
•Cleary emphasizes the participant benefits

SAMPLE SUBMISSION

Title: Can You Hear Me Now? Remote Eavesdropping Vulnerabilities in Mobile Messaging Applications

Submitted by: Natalie Silvanovich, Security Engineer, Google (Black Hat USA 2021)

Abstract:
On January 29, 2019, a serious vulnerability was discovered by multiple parties in Group FaceTime which allowed an attacker to call a target and force the call to connect without user interaction from the target, allowing the attacker to listen to the target's surroundings without their knowledge or consent. While this remarkable bug was soon fixed, it presented a new and unresearched attack surface in mobile applications that support video conferencing. This presentation covers my attempts to find similar bugs in other messaging applications, including Signal, JioChat, Mocha, Google Duo, and Facebook Messenger.

Notes:

- *Detailed, yet concise abstract*
- *Defines a problem and offers a solution(s) that will be examined during the session.*

Presentation Outline

- Introduction
  - Explain FaceTime bug that caused call to be answered without user interaction
  - Goal: determine how these bugs happen and could they occur anywhere else
- WebRTC
  - What is WebRTC? (the video conferencing library used by most apps other than WhatsApp and Facetime)
  - How does WebRTC signaling work? What needs to happen for a microphone or camera to send content
- Analysis techniques (How to determine whether an app uses WebRTC and what the state machine looks like)
  - Documentation
  - Frida
  - Reversing with IDA or apktool
- Results
  - Signal and Facebook Messenger bugs
  - Both of these bugs allow audio to be transmitted without user consent
  - Both of these bugs occur due to not checking whether the software is in the correct state before moving to the next one
  - Mocha and JioChat bugs
  - Both bugs occur due to developers not understanding WebRTC functionality
  - Both bugs allow both audio and video to be transmitted without user consent
  - Google Duo bug
  - Occurred due to bad threading
  - Allowed the camera to take photos and transmit them with no user interaction
- Conclusions
  - Basically, everything I looked at had a similar bug to the Group FaceTime bug

- Root causes are lack of attention to security in design, lack of understanding of video conferencing bugs and lack of understanding of this type of vulnerability
- A lot more research is needed

Notes:

- *Clearly conveys progression of talk.*
- *Helps the Review Board visualize the presentation in its entirety.*
- *Gives an explanation of each area of the presentation.*

Attendee Takeaways (What are three actionable takeaways included in your presentation?)

1. Security researchers should investigate calling state machines when looking for bugs
2. Developers should be careful when implementing calling state machines as they are very vulnerability prone
3. Application design is very important to how these bugs work, and small changes can go a long way in preventing these types of bugs

Notes:

- *Submitter fulfilled requirement of providing 3 takeaways*
- *Explains relevance to the audience*
- *Cleary emphasizes the participant benefits*