Intel Supply Chain Overview

Matthew C Areno, PhD
Senior Principal Engineer
Nighthorse Lake and C3D Chief Architect
Office of the CTO, Intel Systems Architecture and Engineering (SAE)
Notices and Disclaimers

Intel provides these materials as-is, with no express or implied warranties. All products, dates, and figures specified are preliminary, based on current expectations, and are subject to change without notice.

Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No product or component can be absolutely secure. Check with your system manufacturer or retailer or learn more at http://intel.com.

Some results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Intel and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

*Other names and brands may be claimed as the property of others.

© Intel Corporation
AGENDA

- Who’s this guy
- Overview of Intel’s supply chain
- Tools and Processes used
- What gaps exists
WHO AM I?

• Began career at Sandia National Labs doing nation-state level, red teaming activities.
  • Focused on reverse engineering and vulnerability exploitation work against embedded systems

• Worked for Raytheon SI-Gov/Cyber Security Innovations group.
  • SME on PRoT technologies for COTS equipment
  • Graduate of Raytheon and US Navy anti-tamper courses

• Joined Intel in 2019 and now chief architect of new high-security processing solutions
  • Former Sr Director of Security Assurance and Cryptography
  • Authored internal and external Intel documents for supply chain threat models
  • Assisting in the development of supply chain specification for ISO and USG
  • Primary interface to USG for security-related engagements
LET’S LEVEL SET

- Intel Compute Lifecycle Assurance (CLA) initiative identifies four primary stages of product lifecycle:
  1. Build
  2. Transfer
  3. Operate
  4. Retire

- What standards and efforts exist today are primarily focused on Transfer and Operate phases, with little education or definition on the Build phase.

- In this presentation, we’ll focus on the Build phase and how it can be assessed.
THE SUPPLY CHAIN
1. Define customer requirements
2. Register product into SDLe database
3. Begin product definition
4. Establish execution team and define product timeline

1. Create hardware and/or software design for product
2. Identify external components and providers
3. Complete internal program and security reviews

1. Integrate custom and 3rd party components into overall design
2. Complete full system synthesis and initial testing
3. Tape-out final product and send to manufacturing

1. Create product masks, typically one per layer of IC
2. Begin mass production of silicon wafers
3. Conduct wafer sort, validating structural and electrical characteristics
4. Package wafers for transfer to ATMs

1. Products are assembled and enclosed in product packaging
2. Packaged product begins first stage testing
3. After thorough testing, product are transitioned into high volume manufacturing (HVM) stage

1. Minimal provision perform after assembly to support early product tests
2. Functional product are fully provisioned with pre-generated device data and settings
3. Final testing is performed to validate provisioning

1. Final products are recorded in internal databases and sorted for destinations
2. Coordinate distribution of products to partners and customers
3. Manage disposal of failed products

* Thanks to depositphotos.com for images
Intel Global Manufacturing Footprint

Robust, geo-diverse and expanding supply for wafer and packaging

- Oregon: $3B
- Arizona: $20B
- New Mexico: $3.5B
- Ohio: $20B

- Leixlip, Ireland
- Magdeburg, Germany
- Kiryat Gat, Israel
- Chengdu, China
- Ho Chi Minh City, Vietnam
- Kulim, Malaysia

New site under construction
# IC Supply Chain Threats (10K Ft View)

<table>
<thead>
<tr>
<th>Conceptual</th>
<th>Design</th>
<th>Integration</th>
<th>Manufacturing</th>
<th>Testing</th>
<th>Provisioning/Configuring</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insider Threat</td>
<td></td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
</tr>
<tr>
<td>Design Tools</td>
<td></td>
<td>Malicious Hardware</td>
<td>Insertion of Trojan Circuitry</td>
<td>Design Alteration</td>
<td>Falsefication of Test Results</td>
<td>Unauthorized Disclosure</td>
</tr>
<tr>
<td>Third-Party Plugins</td>
<td></td>
<td>Malicious Firmware</td>
<td>Insertion of Trojan Components</td>
<td>Component Replacement</td>
<td>Unauthorized Disclosure</td>
<td>Physical Alteration in Transit</td>
</tr>
<tr>
<td>Attack on Design Networks</td>
<td></td>
<td>Design Alteration</td>
<td>Reverse Engineering of Design</td>
<td>Unauthorized Disclosure</td>
<td>Component Replacement</td>
<td>Replacement of Valid Firmware</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unauthorized Disclosure</td>
<td>Attack on Design Networks</td>
<td></td>
<td>Improper Device Settings</td>
<td></td>
</tr>
</tbody>
</table>
### SW Supply Chain Threats (10K Ft View)

<table>
<thead>
<tr>
<th>Concept</th>
<th>Design</th>
<th>Implementation</th>
<th>Build</th>
<th>Integration</th>
<th>Test</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
<td>Insider Threat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Falsification/ Compromise of User Credentials</td>
<td>Compromise of Build System</td>
<td>Compromise of Build System</td>
<td>Compromise of Build System</td>
<td>Compromise of Deployment System</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modification of Submission Logs</td>
<td>Injection of Malicious/ Vulnerable Library</td>
<td>Injection of Malicious/ Vulnerable Library</td>
<td>Injection of Malicious/ Vulnerable Library</td>
<td>Compromise of Update System</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compromise of Design Tools</td>
<td>Compromise of Signing Keys</td>
<td>Compromise of Signing Keys</td>
<td>Compromise of Signing Keys</td>
<td>Malicious Insertion of Unauthorized Code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Malicious Plugin for Design Tools</td>
<td>Malicious Use of Signing Keys</td>
<td>Malicious Use of Signing Keys</td>
<td>Malicious Use of Signing Keys</td>
<td>Replacement of Valid Binaries/ Patches</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exfiltration of Source Code or Sensitive Data</td>
<td>Impersonate Library Repository</td>
<td>Impersonate Library Repository</td>
<td>Impersonate Library Repository</td>
<td>Extraction of Customer Information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Deletion of Data</td>
<td>Deletion of Data</td>
<td>Deletion of Data</td>
<td>Deletion of Data</td>
<td></td>
</tr>
</tbody>
</table>
INTERNAL TOOLS
Challenge: Threats & Threat Model

Top 25 CICD Threats

<table>
<thead>
<tr>
<th>Threat ID</th>
<th>Threat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Modification of Concept/Design content</td>
</tr>
<tr>
<td>2</td>
<td>Modification of Requirements content</td>
</tr>
<tr>
<td>3</td>
<td>Exfil of Concept/Reqs/Design content</td>
</tr>
<tr>
<td>4</td>
<td>Modification of Source code</td>
</tr>
<tr>
<td>5</td>
<td>Read/Modification of Defects/Vulns</td>
</tr>
<tr>
<td>6</td>
<td>Compromise of Design/Dev tools</td>
</tr>
<tr>
<td>7</td>
<td>Exfil of Source code &amp; Defect/Vulns</td>
</tr>
<tr>
<td>8</td>
<td>Modification of Source code</td>
</tr>
<tr>
<td>9</td>
<td>Compromise of Build system</td>
</tr>
<tr>
<td>10</td>
<td>Modification of binaries</td>
</tr>
<tr>
<td>11</td>
<td>Malicious 3PIP, Bad dependency</td>
</tr>
<tr>
<td>12</td>
<td>Modification/Compromise of Keys</td>
</tr>
<tr>
<td>13</td>
<td>Exfil of source code, 3PIP, binaries, keys</td>
</tr>
<tr>
<td>14</td>
<td>Modification of Test content</td>
</tr>
<tr>
<td>15</td>
<td>Modification/Bypass of Test results</td>
</tr>
<tr>
<td>16</td>
<td>Compromise of Test tools</td>
</tr>
<tr>
<td>17</td>
<td>Exfil of Test content &amp; results</td>
</tr>
<tr>
<td>18</td>
<td>Modification of Release content</td>
</tr>
<tr>
<td>19</td>
<td>Read/Modification of Customer content</td>
</tr>
<tr>
<td>20</td>
<td>Compromise of Deploy/Update systems</td>
</tr>
<tr>
<td>21</td>
<td>Exfil of release content, customer content</td>
</tr>
<tr>
<td>22</td>
<td>Compromise of Deployed product</td>
</tr>
<tr>
<td>23</td>
<td>Read/modification of Defects/Vulns</td>
</tr>
<tr>
<td>24</td>
<td>Read/modification of Customer content</td>
</tr>
<tr>
<td>25</td>
<td>Exfil of Defects/vulns, customer content</td>
</tr>
</tbody>
</table>

Source: Intel Supply Chain Security Experts Team
Threat Model / Attack Tree

- Use centralized log servers (with append only, encryption, etc).
- Use fully scripted or automated build process that generates metadata and enables provenance to be created.
- Use version control and a hosted build service that generates authenticated provenance. All build steps run using same build service, not on a developer’s workstation.
- Ensure that the build platform meets specific requirements to guarantee the auditability of the source and the integrity of the provenance.
- Apply hardening procedures on the environment hosts (logs in place, integrity, etc).
- Deploy and use a SEM (Security Information and Event Management) system.
- Use intrusion detection and protection systems.
- Deploy and use a SEM (Security Information and Event Management) system.
- Use cryptography (in transit and at rest).
- Apply hardening procedures on the environment hosts (CIS Benchmarks – just as a starting point).
- Monitor / perform integrity checks of executables, libraries, configuration files, etc.
- Use centralized log servers (with append only, encryption, etc).

Compliance of Build
- Centralize building process to ensure consistent results.
- Ensure logs are maintained for audit purposes.
- Use centralized log servers (append only, encryption, etc).
- Use fully automated, reproducible build process.
- Verify the build logs and ensure all dependencies are met.
- Use intrusion detection and protection systems.
- Deploy and use a SEM (Security Information and Event Management) system.
- Use cryptography (in transit and at rest).
- Apply hardening procedures on the environment hosts (CIS Benchmarks – just as a starting point).
- Monitor / perform integrity checks of executables, libraries, configuration files, etc.
- Use centralized log servers (with append only, encryption, etc).

Compromise of Code Repository
- Compute checksums for all artifacts.
- Store checksums securely and verify on-demand.
- Use version control to track changes and ensure integrity.
- Ensure that all builds are signed and verified.
- Use centralized log servers (append only, encryption, etc).
- Use intrusion detection and protection systems.
- Deploy and use a SEM (Security Information and Event Management) system.
- Use cryptography (in transit and at rest).
- Apply hardening procedures on the environment hosts (CIS Benchmarks – just as a starting point).
- Monitor / perform integrity checks of executables, libraries, configuration files, etc.
- Use centralized log servers (with append only, encryption, etc).

Threat Model
- Identify potential threats to the system.
- Classify threats based on likelihood and impact.
- Develop strategies to mitigate threats.
- Test plans to ensure effectiveness.
- Monitor and adjust strategies as needed.
- Use intrusion detection and protection systems.
- Deploy and use a SEM (Security Information and Event Management) system.
- Use cryptography (in transit and at rest).
- Apply hardening procedures on the environment hosts (CIS Benchmarks – just as a starting point).
- Monitor / perform integrity checks of executables, libraries, configuration files, etc.
- Use centralized log servers (with append only, encryption, etc).

Attacker Tree
- Identify potential attackers and their motivations.
- Classify attackers based on their capabilities and access.
- Develop strategies to prevent attacks.
- Test plans to ensure effectiveness.
- Monitor and adjust strategies as needed.
- Use intrusion detection and protection systems.
- Deploy and use a SEM (Security Information and Event Management) system.
- Use cryptography (in transit and at rest).
- Apply hardening procedures on the environment hosts (CIS Benchmarks – just as a starting point).
- Monitor / perform integrity checks of executables, libraries, configuration files, etc.
- Use centralized log servers (with append only, encryption, etc).
INTEL THREAT MODELING TOOL (TMT)

- Centralized tool and threat database to assist users in the creation of robust/complete threat models
- Threat models are all stored in one location, making vulnerability triage easy and fast
- Created from the ground up at Intel
USG/DoD – Problem/Challenge – need to grow supplier landscape while increasing security/assurance of intellectual property (IP)

In response to NDAA section 224, USG/DoD are defining a new framework to govern the acquisition of microelectronics products & services from commercial factories with additional assurance.

MQA Framework foundation:
- Applies to custom ICs.
- Built on zero trust.
- Designed as a supplier risk assessment model.
- Allows for scalability – levels of assurance.

Designed for integration with USG/DoD programs – RAMP, RAMP-C, SHIP
MQA RESULTS

- Intel is supporting the framework development with feedback on practical implementation and pilot studies to demonstrate feasibility.
- MQA pilot demonstration helped to evaluate feasibility of MQA process in commercial foundry
- Intel comprehended MQA as part of continuous improvement foundation into its broader operations
- MQA standard can facilitate achievement of USG NDAA for quantitative assurance framework for microelectronics
INTEL® TRANSPARENT SUPPLY CHAIN HIGH LEVEL PROCESS

- **Source**: Data gathered and communicated on sourced components
- **Build**: Creation of digital certificate for platform
- **Integrate**: Digital certificate extended to cover full system
- **Validate**: System authenticity verified against digital certificates

**Component Data**
- **Suppliers**
- **Manufactures**
- **Integrators**
- **Deployers**

**Trusted Computing Group Platform Certificate**

**"As Built" Data**

**Trusted Computing Group Delta Certificate**

**"As Built" Data**

**System Snapshot**
What is Project Amber?

An Intel Service to remotely verify and assert trustworthiness of compute assets. (TEEs, devices, Roots of Trust, etc.)

Operationally independent from the Cloud/Edge infrastructure provider that is hosting confidential compute customer workloads.
## Relevant Standards: NIST, DFARS, CHIPS

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
<th>TSC Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFARS 246.870-2</td>
<td>Contractors’ Counterfeit Electronic Part Detection and Avoidance</td>
<td>• TSC aligns to DFARS</td>
</tr>
</tbody>
</table>
| NIST SP800-161 r1                                           | Cybersecurity Supply Chain Risk Management Practices for Systems and Organization | • TSC aligns to NIST SP  
• TSC highlighted in Intel/Coalfire paper                   |
| Trusted Computing Group Platform Certificate Specification v1.1 | Trusted device manufacturing, traceability, and transparency               | • TSC Delivers TCG Platform Certificate  
• V1.1 improved (April 2020) and gaining momentum                      |
| NIST NCCOE SP1800-34. (A, B, C)                             | Validating the Integrity of Computing Devices                                | • Version C (12/2022) TSC specified within “How To”                         |
| US Government CHIPS Act                                      | Mandates plan to identify and mitigate relevant semiconductor supply chain security risks; access, availability, confidentiality, integrity, and a lack of geographic diversification in the covered entity’s supply chain | • TSC aligns to SEC 103. SEMICONDUCTOR INCENTIVES  
Sections iii & iv page 16                                      |
| NIST NCCoE Manufacturing Supply Chain Traceability with Blockchain Related Technology | Introduces the concept of a manufacturing supply chain “traceability chain,” which is comprised of a series of immutable manufacturing traceability records written to industry- specific ecosystem blockchain-related technologies. | • TSC BC aligns to the manufacturing traceability with an immutable ledger technology |
EXISTING GAPS/CHALLENGES

- Consistent standards for security between integrators and suppliers
- Independent verifier of vendor HW/FW authenticity
- Data model specifications for AI engines to assess supply chain security
- Tools/specifications for validating HW design compliance with security requirements
INTEL CONTACTS

- Transparent Supply Chain: Tom Dodson tom.dodson@intel.com
- Threat Modeling Tool: Jonny Valamehr jonathan.k.valamehr@intel.com
- Intel Foundry Services: Jeff Josiah jeff.josiah@intel.com
- Intel Trust Authority: Raghu Yeluri raghuram.yeluri@intel.com
Thank You!!