>> So hi, everyone. As Allan said, Art Manion at the CERT Coordination Center. I am one of the two co-chairs of the -- colloquially, we call it the "Framing Working Group." However, I see, officially, it is the Understanding the Problem Working Group. We'll have to deal with that.

I spent most of my efforts picking out this free-use picture of window frames to have a picture in my entire slides that are all text, otherwise. So please enjoy the only bit of presentation mastery you're going to see from me today.

>> (Inaudible)

>> Yeah. Thank you. A solid half hour, I think, that I did not need to spend on that. Oh. Sorry. I've got to play the thing where it . . . page -- good. So I've got a bunch of slides. Towards the end, they're basically just animation -- kind of cheap animation slides, so that's not going to take up too much time. But I'll try to convey some pictures towards the end that will help, hopefully, do some of the framing that we're working on here.

Actually, let me go back a minute. I mentioned I'm one of the co-chairs. Michelle Jump is the other co-chair. She sends her regrets. She's off in Europe at something that she couldn't even kind of break out to call in today.

So anyway, Michelle's not here today. I will claim that I speak for the framing group more or less and Michelle more or less, but that's not entirely true. We're a fairly loose organization. Everything in here has been through our working group, but it will be my flavor, of course, of sort of delivery. So, you know, 80, 90 percent confidence that the working group agrees with everything I'm going to say today.

I've roughly got this divided into two sections. One is what I'm calling "stable topics." So things we've been talking about for weeks and months that have congealed, sort of there's some consensus on them. I'm not saying they're solid and completely done, but we've got a class of things that are in pretty good shape and a class of things that more open questions, as Allan was mentioning earlier.

So I'm going to run through the stable stuff fairly quickly because it's stable. Again, even though it says "stable" here does not 100 percent mean that it's done. If you have questions or issues with things, I think we're going to park open issues and things to talk about for after lunch today.

So this may be a most important point, this first sentence -- actually, I'll select it here -- this first line. We've had a lot of talk about what data elements, what data needs to be in SBOM. And what I think we're coming down to is, there's some core information that we're calling "core identity," "core component identity" -- I've heard "minimum viable identification" recently" -- without which there is nothing else, right?

If I can't talk about the component or that component or the other component, I can't say that they relate to each other. I can't say who built it. I can't say how it was built. I can't say where it came from. I can't say there's a vulnerability in it. I have to have this core component identity.

So again, the consensus emerging is, you have to have that. We're getting into sort of how to create that identity. It's a big problem. There's lots of different software. Lots of ways to do versioning and naming. But the construct of namespace: name. And I have some examples to make this more concrete. We've also been talking about the tuple or triple or quadruple of supplier name, component name, a version, and possibly a hash of the component. And the hash, in this case, is more for uniqueness and less for integrity. However, you might get integrity out of it.

There are lots of different names for things. There are relationships between things. We're sort of on the edge of saying that relationships to other components might be a required part of anything we produce. And sort of aliases or other names, which could be a relationship, are also very important. You can imagine the many different ways to spell open SSL -- capital, lowercase, spaces, dashes, right? -- as one of the examples there.

Let me just stop for just a minute. Is there any sort of strong disagreement? I don't want to mess up the flow here, Allan, but any strong disagreement on this sort of core identity being a required thing? You can raise your hands or shout if you'd like. Okay.

Everything else is basically optional. That's not to say it's not important at all. In fact, the list of ingredients, the core identity, may not be super useful all by itself. You probably have to have some extra information. And you can have just about whatever you want tied back to this core identity. We've got a slide on that coming up here. Okay?

Sorry. Page down. So the Framing Working Group has written a problem statement. I don't have the text in here, but there's a Google Doc. It's linked from the slides, which if the URL works, then you can look at this yourselves. We have a list of terms which I'm calling stable, but that might be an overstatement. The four terms -- there's only four I listed here. "Supplier." That's code for vendor, developer. The person producing the SBOM or organization producing the SBOM.

"Component" is defined by the supplier unit of software, possibly hardware, that is getting the ID on it.

"SBOM" and "inventory" are not quite as stable. Conceptually, I think the issue here is, What do we call the broader collection of all the data and the processes and the way we share, and what do we call the core list of ingredients of inventory?

The temporary proposal here, at least, is, SBOM is the larger umbrella term, and inventory, or perhaps co-identification, is the subset. This is open for discussion. Even Michelle and I haven't quite come to terms and agreed on this yet. But there's at least, conceptually, an issue there that we have to deal with and pick a couple of words for.

When to produce an SBOM is reasonably stable. There are a lot of opinions on this. So, you know, that counters what I just said, but we're going to focus on what is delivered from a supplier to the downstream -- their customer, their consumer, the next hop-down stream. So something as- built, as-delivered, as-packaged. I realize those are actually technically different things in a software development lifecycle, but that's the high-level area we're aiming for. We're not collecting source code here. This is the thing as it's built and handed off. If I hand you some software and you want to know what's in the thing I just handed you, that's what the SBOM is about.

When I change -- on the supplier, when I change my software and change the component, that's a change to the SBOM, and it's a new SBOM. That's the theory here. So a patch is a new SBOM.

Sorry. Page down. We had a lot of conversation in the last several months about sort of the depth of the SBOM. There was a discussion about one hop versus multi-hop. I think, in the end, that was a bit of a false dichotomy. It's not an either/or situation. And the theory we have here in the Framing Working Group is that each supplier -- if you imagine each supplier in the giant dependency chart has direct responsibility for the things they create. They make an SBOM for that.

And since I'm making an SBOM for the stuff that I typed -- you know, my devs typed at the keyboard -- they know what they included. The tools know what they included. So one hop up the supply chain. I grabbed this open-source library. I compiled it in. I wrote some code. I hit "compile." I hit "package." I have firsthand knowledge of what I just wrote and what I included. So that's low-cost because I know exactly what's in there. I'm not guessing multiple hops back.

And when I supply this component downstream, the whole package goes with it. So that's the theory. And if you imagine everyone -- someday 100 percent, right? -- in the entire world creating software does this, the collection of SBOMs accumulates at every point in the chart, and we've all got the SBOM information at our fingertips.

But no one bears the full cost of going all the way up the chain or collecting all the information. Each supplier node is responsible for what they know best.

Sorry. Quick comment here. I'm personally pretty happy with this theory. It hasn't really been fully tested yet. I have an as-yet executed plan to get a bunch of colored index cards and some people in my office and sit down in a room and do a paper exercise, literally, of, Pretend you're a supplier. Write down some stuff, and pass cards around for an hour or two and see if this thing actually works the way I think it does. So not done yet. Going to try to do that.

So that's the end of the stable quote, unquote, topics. We're now on to the open topics. I'm going to run through these. I guess, in the interest of time, if you -- again, same with the stable topics, and these, as well -- if there's things you want to discuss or disagree with, flag them for the afternoon session, right?

>> We have a little bit of time now. So you have until 10:45. So that's --

>> Oh. Okay.

>> -- a little bit of time.

>> Sure. Sure.

>> So please feel free to ask a question. And for those of you who are watching the webcast or on the Callbridge, you hit star 1. That will raise your hand, and we'll see you in the question queue. So you can join in, and you'll speak us to from the ceiling.

>> All right. So moving on to less stable concepts and topics. There's the question of how we exchange these SBOMs, right? If all these suppliers are producing them, how do we pass them back and forth and how do we collect them all?

What I think the Framing Group will end up saying or suggesting is that there's not going to be just one way. There are different types of software. There are different types of devices. We have things with constrained resources that don't really have a dynamic file system.

So at the framing level, we're probably going to suggest you have to pass the SBOM along. Here are two or three options that you can pick from, and then technical implementation is down, you know, a couple layers further.

So it's fairly common, if you have a bigger system or more general purpose computing platform, that a package of files that's a component has the SBOM files right in it when it gets installed and delivered. We've talked about being able to look up at a supplier's website. You know, I have a unique part ID or a unique serial number, and I can look up the SBOM for that device elsewhere.

There's a couple of more fancy technical things. There's ROLIE in IETF, which is sort of a mechanism -- I think it's an extension to Atom's feed system to sort of provide information.

And this other thing, SParts has (whispering) blockchain. Sorry. I win. I know. But it has it in a not crazy way. It's irrelevant that it has blockchain, really, but there are people thinking about, technically, how to share these things at scale, is the important part. Blockchain is not important part.

>> (Inaudible)

>> But I whispered it the first time. Anyway, I will not say (whispering) artificial intelligence at all.

>> It's machine learning!

>> I won't say that, either. Sorry. Oh, yeah. Right. There's been the question of maintaining the history of SBOMs. The Framing Group has not thought about that firsthand very hard, or I missed those calls, possibly. There is nothing stopping anyone from collecting history, but I haven't -- to my knowledge, we haven't figured out a way to build it in, necessarily.

A supplier can certainly keep all the history and provide that. As a consumer, I can collect all the history and provide it. There's nothing stopping a third party from archiving a bunch of stuff and keeping that history. But there's nothing inherently, like blockchain, like in the thing that we're designing so far that has the history baked into it.

Opaqueness and a transition period. So in my dream world, when there's 100 percent compliance with this idea and it works well, we all have all the SBOM we need and it's a done deal and there's no transition period and there's no opaqueness.

In the years or decades it's going to take to get to that point -- and we'll never actually achieve it -- we're going to have areas of lack of knowledge in this big graph of suppliers and their supply chains.

So when an SBOM is not available, we need to sort of -- there's a couple things we need to be distinct about. If I'm claiming I use a part and I don't know what's further in that, I can name that thing and say, Here's a part I use from somewhere else. I can put it to use from somewhere else. I actively am telling you, I do not know what's beyond that. There may be other parts in there. There may be not. I just don't know.

Or I may also know that that is originally built software. There's nothing behind that. There's nothing upstream of that component. And conveying the difference between there's nothing behind that and there might be something behind that might be important to doing all the special graph magic that will help us process all this, our computers process all this, right?

In the end, humans might glance at formats and things, but we're going to have to have tool support to parse all this stuff. The scale of the data is going to be much more than people are going to want to stare at.

>> Art, Dave Waltermire from NIST. There's a robust ecosystem of discovery tools that do a lot of work to identify software that isn't identified by any other means on a given device. It seems like there might be an opportunity to provide some recommendations for that tooling ecosystem as far as what are some of the desirable outcomes that we would like to see there.

So for example, they could expose an SBOM that's available in a component that is included in a software, if that's available, even though an SBOM for that actual

component isn't present on the device or otherwise able to be acquired. And that would be more desirable than nothing at all, I would guess.
>> Sure. So just let me make sure I got that. So the SCA sort of area, so software composition analysis. I'm looking at binaries. I'm look at what's in there. I can detect some things. There is no SBOM magic in place yet. Would these tools then output something that lines up with this system? Is that what --
>> Yeah. As well as looking at any software dependencies, libraries, you know, other components that are pulled into the software either by installation or by downloading those external dependencies.
>> Sure. Yeah. At framing, we haven't tackled that real well. I took a note here. We haven't discussed it very much. We're aware of the SCA market. I don't think this work really would -- it might affect that market, but it's not going to remove the need for those things.

Like I said, transition is going to be more or less infinite. So good point. Thank you.
>> So this is --
>> Sorry.
>> Oh. Sorry. Go ahead.
>> JC, two, three. Yeah.
>> Gotcha.
>> And to that point -- and I say this as an SCA vendor -- I'm very careful to remind customers that ignorance is not as bad as the illusion of knowledge, right? So to Art's point about noting when you don't know, you can infer with a degree of confidence that you have to be honest about; but you also have to tell the system that it's an inference. Because we see examples of, Oh, well, yeah. It's a good package, but originally, it was taken down off a bad mirror, right?

So when you don't know something about the supply chain, you have to mark it as "not known," and then people can go ahead and use tooling to make inferences. But that has to be established as an inference.
>> Sure. Thank you. Sir? Yep?
>> Good morning. My name is Eliot Lear. I work for Cisco. I see two separate problems that I think you were mentioning there in terms of discovery. The first is sort of upstream supply chain. That is to say, I'm a supplier. I'm picking up a package. I want the SBOM for that package, and I'm going to include it in my SBOM. That's one problem of discovery. And actually, that probably is an easier problem to address because you can just put it into your supply processes. It may be a mere matter of bits, but I think I know -- Dave and I and a few others know how to deal with bits on a wire.

The second problem is much more tricky, which is, how is it that you convey the SBOM of a particular device to the end user of that information? And I don't know if you guys are thinking in those terms, but we do have some -- I have some ideas there. I'm not going to delve into the solution management right now, but I do see it as a split problem.
>> I'll say briefly -- and then you're next, yep. Sorry. I'll speak for myself and probably for the group. I'm trying to treat those, honestly, as the same problem. I don't want to have special sort of parts of the graph. So a consumer, as you're describing it, would just be the result of all the SBOMs and supply nodes kind of flowing down.

How that's delivered, again, may depend on the software, the ecosystem, the device. It might be a collection of files. But if I have devices, I may have to look things up. So unclear, technically, quite how to pull it off. But in kind of a high-level graph concept, a -- what is it? -- a leaf node is basically someone who is not providing anything further. They're just consuming SBOMs; and their set of SBOMs, ideally, would be the full set of all the things that they're running. Is that -- yeah. I'm happy to talk more, too. So --
>> Yeah. I don't want to --
>> Sure.
>> -- go into the weeds with you. I do think there are some issues, though, with that leaf node.
>> Yeah. No. Happy to discuss that. Thank you. Yes, sir?
>> Les Gray, Abbott.
>> Hey, Les.
>> So in this working group, no, we haven't discussed it. In the Proof of Concept, we have discussed tools. What we would really appreciate is some recommendations, because what we've seen so far with SCA tools is, there's lots of flaws in them. There's no one that's really that great and good, and it causes a lot of issues in how we identify this software.

And so if you can recommend some or you have some that you know are good, then we would be willing to take a look at that. And I don't think we're discussing it today in our working- group piece, but that's the group that we have looked at it.
>> Okay. Thanks. I'm going to -- just checking the time. I'm going to go a little quicker through some of these. Again, these are marked as open, so they are open for discussion.

There's only one line here, but we've recently started considering, a little more directly, awareness, adoption, sort of a simple how-to guide. How do we get tool support for these things? Ultimately, we're not going to ask people to build these by hand. We're going to have tools. Your built tools will make this, right? And we need tools on the client- consumer side to process it and give you lists of what you have.

So Open Topic 3. I had mentioned earlier -- cited this in an earlier slide, but from the Framing Group, these are four of unknown -- number, how many -- sort of pretty high-level use cases or applications of having SBOM data.

So most of these require not just the core ingredients list but additional information. I did not synchronize these in advance with the Practices Working Group. I don't know if the words match up well, but they're pretty high-level use cases. We've talked about them for a long time, so I think they're safe.

Most of the reason this is here is to really tell people, the thing you actually want to do with the your SBOM data you will be able to do, provided you have the data. We're not excluding any of these applications or anything else that might turn out to be useful. But things that we've seen either happening already or are clearly articulated use cases are licensing and intellectual property.

We have a question on, actually, the term. We've discussed briefly not using the word "license" but something different to imply entitlement. I'm allowed to use ten seats of this software versus the GPL. How am I supposed to use this software, and what am I supposed to pass on with that?

We've talked about high assurance; you know, where things came from, who touched them. Really, really, where do they come from and who touched them along the way? How are they built? Do the parts have high integrity the entire way through the chain?

The topic that got me here is vulnerability management. When a five-year-old HARP lead is noted to be in a component named open SSL 098- something, I want to search through all my SBOMs. And when I find those, guess where I'm going to go look for HARP lead patches?

Oh. An important point here. I am going to go look for that component in my list of SBOMs. Having it does not necessarily mean I'm vulnerable to HARP lead. So we need a way to convey sort of exposure or exploitability of the vulnerability, not just the fact that it is associated with a certain component.

From the Framing Working Group, we're curious, you know, if there's a framing document, what other use cases are well-understood enough or common enough that they should get a call-out here. So if folks have suggestions, please say so. Say something later or write us and let us know what else you think should be in a sort of high-level list.

We're not trying to be comprehensive. I honestly expect, if this thing actually works, we'll find new use cases that we didn't think of before because the data will be available. But we want to cover the main and well-known ones.

Oh. Well, I have awareness and adoption again here. We've talked about sort of a very easy one, two, three/ABC guide for the "crawl" stage, folks just getting into this. There's a Healthcare Proof of Concept Working Group happening right here today. We're going to hear from them. So that's a sector-specific example that may have something to show.

We have existing formats, the most common of which are SWID and SPDX. There are tools today that will help produce SBOM, but we need something -- you know, more adoption and more global use of tools. We have not thought hard about it in the Framing Working Group yet. It's a topic that came up in the last two or three calls.

Another issue is, what do we do about sort of a not-on-premises service cloud SBOM? You know, preliminary thinking is, if I am AWS and I am running systems, I'm a consumer of other people's SBOMs. I'm going to want those to know what's running and what to patch.

As an individual who has my stuff hosted in AWS, I may not really benefit much from having their SBOM. Plus, it may change daily or hourly, which may not be something I want to deal with.

Nothing in the Framing Working Group's work so far prevents doing off-premises or cloud sort of SBOM, but we're not focusing on it, is our answer for the moment.

Okay. I'm going to fairly quickly get into the fun picture part -- more fun picture part. These are somewhat toy examples, but they're as real as we've got for the moment of what an SBOM could look like in a couple of different formats.

This is the Excel spreadsheet format. If you so choose, you can create an SBOM like this. So as we touched on earlier, this has the five tuple, now, of the supplier name, the component name, a version, a hash -- which I made up -- and a relationship.

And I just picked one basic relationship here. It includes something else. What this is trying to tell you is, the supplier name, Medical Device Manufacturer 1, makes

something called FooPump, Version 4.0. It has Apache in it, and Apache has open SSL in it. And open SSL is just open SSL, right? It's a very, very small chain of three nodes. Okay? And all of the examples are going to use this same toy example.

So here's the spreadsheet version. Here's kind of just a made up namespace: name, right? And I'm just using domain names backwards here. And again, at the framing level, there's a lot of ways people do versions and software. I don't think we're ever going to say, You must name it exactly this way. I don't think that's going to fly. So we're trying to figure out a way to be flexible enough but still have sufficiently unique identifiers on component parts. This is probably the loosest example we've got.

Here's what this looks like in SWID, if my manual editing was close enough. So this is XML. This is SWID XML. It's saying all the same things, right? At the bottom here, you'll see "link." And that's saying Apache requires open SSL. I didn't put the FooPump in here. Sorry. I'm running out of space.

Again, if my editing is correct, this is the SPDX version. SPDX does have an XML or RDF version, as well, but this is more texty. I think it might be (inaudible). I'm not sure. But again, there's a relationship built into SPDX to say, I chose prerequisite of. I'm not sure that's the right relationship, but this is saying the same thing, right?

A human, you know, is going to read this maybe, and the tool might spit out something nice. But under the hood, it looks kind of like this, looks kind of like this, right?

Graphical version of it. The tree on the right is what I was just showing in the SWID and SPDX things. If you don't like super complicated pictures, don't watch this slide.

(Laughing)

>> We have a thing in the Framing Group. Some of us -- and I'm very guilty of this. Even though we're the Framing Working Group, go off and get too into the weeds, technically, sometimes. And we're trying to sort of balance the level of depth and technicality, sort of, we get into.

So there's no reason to be concerned with this drawing, this graphic. This is sort of a broad architectural picture of what things conceptually look like. It absolutely does not mean that you can't just do this on Day One. This very basic spreadsheet and very basic text-file list of what's in the product I just gave you is basically -- I'm not sure it's -- yeah, it's basically these four little boxes over here.

So we have to talk about things in a grand architectural way if this is ever going to scale. That doesn't mean we can't do very basic things on Day One.

Here's some other data SWID and SPDX produced. So this is beyond the core identity now, but we talked about needing hashes. We talked about license information. I have "license" and "entitlement" separated here. I didn't find a clear SWID version of, I use the GPL. Is that -- I'm not sure if I'm missing that, but -- okay.

So again, existing formats already support this stuff. This is where I stopped. The Standards and Formats Working Group may go into this further, but we have done some work across working groups to map what the Framing Group is saying we need to have and what's already present in the existing formats and try to have a mapping across those things. We have a Google sheet somewhere that attempts to do that mapping. We've done that homework and tried to sort of get those things sorted out.

Sorry. More pictures, hopefully. Yeah. So in the Framing Group, we've had a road and a bridge, crop circles, flower petals.

(Laughing)

>> I'm missing one at least. A lot of attempts to draw pictures in kind of -- we had trouble with words, so we went to pictures, and then we went back to words and then a new picture.

So today you are presented with a new picture. This is based on the crop circles and the flower petals, but it's going to look a little different. What we're claiming is required is this minimum identity, right? You have to have component identity. It has to be fairly unique or else we've got nothing else, okay? So -- and no one raised their hand earlier, so this is now a done deal. We have to have this. All right? Okay.

>> Just to be clear --

>> Sure.

>> -- "fairly unique" or (inaudible)?

>> I use the words "sufficiently unique," which I realize is not supper technically clear, but is every software product going to make something unique? I'm not sure I can mathematically prove that in advance. So I'm not excluding the possibility of a collision somewhere, but it has to be unique enough that the thing doesn't just completely break.

>> Hi. Duncan Sprawlis (phonetic), Roth Consulting. It has to be unique in the context of which you use it. There are different use cases. Different use cases require different amounts of information. In the context which you're using it, it has to be unique. We didn't get further than that, I don't think

>> Yeah. Sorry. One of the pieces of guidance we're working towards is that quadruple might be enough to be sufficiently unique. So supplier name, component name, version, and the hash. The hash should help with the math and the scaling part of it.

So this is required. Everything from now on is sort of optional. So if I need license information, I attach that to my component parts, my component IDs, right? If I want provenance -- I want to know who made the thing, who's touched it, who's modified it -- if I want to know how it was built, I can tag that.

If I want to get into how the thing tells me it's supposed to be being used, I can bolt that on, as well.

This is mine, vulnerability management. I want to know if vulnerability affects a component, and I want to know that a component actually exposes that vulnerability. That's what I was mentioning earlier. I know Bruce has made this point a lot, but this is very important, right?

Early on in the discussion -- so HARP lead is a good example, right? I forget the gentleman who keeps bringing this up, but there are 1500 function calls in open SSL, and two would make you vulnerable to HARP lead. So you could include open SSL and not call those two, and you are not affected by HARP lead, probably.

There's also some work by the cyber ITL thing. They've actually done real work on studying stuff and found that the fact that a vulnerability is associated with a component is not a strong indicator right off the bat that the component is actually vulnerable. In fact, it's under 50 percent, you know, subject to their measurement choices and things. So important to have it kind of go both ways there.

Also to note, externally, you know, the list of vulnerabilities, the catalog and how they map to the components, is not something NTIA here, the Framing Working Group,

is going to go solve. We know that those are required things to produce vul management, but they're external. We just need to be able to link to them.

The obvious case here would be link to CVE, right? CVE 20140160, which I think is HARP lead, maps to a component. Yes, I know my numbers, some of them.

Oh. So here's, you know, one component includes another. That's all this one is saying here, right? I can map to another one. And we've discussed, in the Framing Working Group, you know, open SSL supports -- I don't know -- triple des, right? So if there's a bug in the triple des standard or it's decided that MD5 is now not a good idea anymore, you can go find all the things that do MD5 and start deprecating them.

So this is the -- instead of the crop circles, this is -- I don't know what to call this. More of a star or something. But this is the new version of crop circles.

>> I think you should just call it "better."

>> "Better"? "Better version of crop circles"? Okay. I want a whole new word. Yeah. So real quickly here, just more touching on this diagram, if I want licensing, I need the highlighted stuff, right? I always need the core. I need the licensing piece.

If I want to do vul management, I need that stuff, right? If I want to do high assurance, at my approximation here, I need how the thing was built and who has been touching it.

With that, if you're interested in joining the Framing Working Group, we try to meet weekly. We hit most of those weekly schedules. Although, Michelle and I travel, and we cancel them from time to time or Allan covers from time to time.

Mailing list. Google Drive with way too many files in it. You'll never be able to tell what's in there. But join the Working Group, and we'll point you in the right direction. Yeah, weekly's Friday at 2:00 p.m. Eastern, if you want to join the call.

Oh. I hadn't done this ahead of time, but Allan had mentioned it in his opening remarks. What do deliverables look like? I imagine -- more than imagine. I expect from the Framing Working Group probably at least one sort of white paper laying out the problem statement, the terms and definitions, talking at a sort of requirements level about the things in these slides and pointing to further work outside of NTIA and within the other Working Groups.

We want to frame this stuff, get some of these concepts across. But roughly, we're trying to sort of be the conceptual requirements setting, documentation, and not delve into exactly how does SWID handle that or how does SPDX handle that or how do you pass the SBOMs along in a technical manner. So we're going to try to stop short of the technical implementation but do the high-level requirements.

That's it for the slide deck. Again, early on, there's a link to a Google Doc, which is the problem statement and the terms and definitions, which are -- again, I think we need those four terms, some question about inventory versus SBOM. And the definitions are very long right now. There's a bunch of notes in there. We need to clean them up and make them strong and concise at some point. So that's it for the official presentation. Thank you, everyone.

(Applause)

>> Okay. Thanks.

>> So thanks, Art. I'm going to keep you up there --

>> Yeah, sure.

>> -- for two minutes in case there are questions or comments, reactions of what you guys thought of that. We've got -- let the record reflect there was a thumbs-up.
>> Oh, yeah. Right. Thank you.
>> And, Art, I'll just say, do you think -- in terms of a deliverable, do you think you can sort of write up what you've described, maybe solving one or two of the open questions and have a draft document by, say, June? Do you think that's feasible?
>> June, you say? Probably. Again, we have a lot of stuff written. It's a question of, Is it usable or do we have to start over? But we have a couple of large documents that are half built.

But, yeah, I think -- you know, a little bit of work on some of the open questions, and that's enough material to say, you know, The world's a better place because we think these are solved. Here's our recommendations. Go forth and do better SBOM. So, yeah.
>> Great. Just also, a reminder for those of you who are listening in on the phone. We know that there's a slight delay between the webcast and the phone and realtime. So if you want to ask question, you can get on the phone and hit star 1. We'll get you in the question line to raise your hand.

With that, I want to thank Art and Michelle and everyone who has been going to those calls. I know that's been a -- they've tackled the really hard challenges. And to Diane's point, it's been an interesting mix of policy people and technical people. They've done a great job of working together. So thank you, guys, and I think you're in a great direction.