The Policy Implications of End to End

December 1, 2000
Stanford Law School Center for Internet and Society,
Stanford, CA

Introduction: Lawrence Lessig, Andy Schwartzman, Jerry Saltzer

LARRY: When I was still on the east coast — part of the east coast establishment, Hal Abelson [phonetic] from MIT and I taught a course at the Harvard Law School and at MIT. The course was titled Social Protocols. And half the students in the course were MIT undergraduates engineers and half the students in the course were lawyers. And they had to work in groups of ten — half from each school — working on particular cyber space related problems. And coming up with a solution.

Now, I think Hal and I thought this was the greatest course we ever did. The students hated the course. They hated it because — there were close to physical battles that happened in this class as lawyers and engineers tried to learn to talk to each other. And it turned out to be extremely difficult.

So when I came to Stanford and I thought what should my first conference here at Stanford be sponsored by the Law, Science and Technology Program, I thought what better conference than trying to bring together and bunch of lawyers and advocates and technologists. And trying to have them do what my students really hated to do.

So this is the one opportunity I have to have a failed conference. The first one doesn't really matter, right? So, here it is and whatever happens, happens. But that's the objective here. The objective is to have a conversation that facilitates understanding across these very different disciplines of technology. And there's an advocacy community here. And lawyers, policy makers, economists types.

So that overall objective is what we should be working for today. Now, a lot of us in this room have been involved in very specific policy battles to have to do with the issues we're going to be talking about today. In my view, that's not an important part of our conversation. It's not our job today to resolve these policy questions.

It's our job today to facilitate understanding across these groups. And so we've devised a program which is divided into

three parts, all of which are aimed at helping us talk usefully about what I consider to be a fundamental principle of the Internet we've known — the principle of the e2e argument as authored by Jerry Saltzer, David Reed and David Clark.

This is the opportunity for us to learn to have this conversation. And the Law, Science and Technology Program and the Center for Internet and Society that's just been started at Stanford hopes to have this conversation about a number of technical policy issues over the course of the next couple of years. This is just our first efforts.

Now, the day breaks up into three parts. In first section, we have four exercises. I don't know how long this is going to take. In fact, there may just be one session. But the objective of the first session is to work through four examples that help us see the particular trade-offs that are raised by the e2e argument.

And so we start with IP telephony [phonetic] and then talk about some network security issues. And then we talk about cashing and then — oh, quality of service and then cashing.

And the objective here is that by the end, all of us can do the routine of working through the e2e analysis for each of these questions. Then we'll take a break and we'll move into — depending on how long. We might have taken four or five breaks. Or somebody might have been broken by this point. But then we'll move into a discussion of cable architecture and then finally a discussion of issues related to wireless.

Perspective that have come out of the morning session will be used to analyze raised by the cable and then by the wireless question. Now, I want this conversation to happen because I'm convinced as a lawyer that the most interesting policy questions there are are technological questions. That architecture as Mitch Kapor [phonetic] started the slogan many years ago — architecture here is politics.

And the problem that we lawyers have is that we have no patience for these technical architectural questions. So I think this conversation is crucial if the values that the Internet community has given to the rest of the world are to be understood enough not to be destroyed by the rest of the world.

In particular, destroyed by my students and lawyers. So that's the reason we're here. And I'm grateful. When I sat down and put together the list of the very best people to have this conversation — all but two, and they're no longer on the list — are here.

So I'm grateful to all of you for coming for this conversation. And

just as I said last night, I would make one plea — that we work to the objective here which is not to prove the truth against darkness, but to prove that we can learn to talk to each other about these issues.

Now, one technical feature of this conversation are these cards. I got this idea from David Eisenberg's [phonetic] big hook conference where they had a little different meaning there. And that big hook meaning was you're an idiot or yes, I agree with you. That's not what these cards mean here.

What these cards mean here is okay, I'm understanding. That's fine, great. Here — I don't get it. I don't wee what you're saying. I don't — you're speaking a language that doesn't yet translate. And you're not to be really rude about it. but it's a simple way to signal that it's not translating across the communities. And if you do it subtly and quietly, then maybe people will respond. If you don't do it subtly, maybe they'll just think well, you're an idiot. But I'm hoping we can do the first rather than the second.

Okay. We're going to start first with a layout — a very brief description of the essence of this argument. But I want to ask Andy Schwartzman first to introduce something about the sponsorship of this conference.

SCHWARTZMAN: Thank you, Larry. Hi, I'm Andy Schwartzman. I'm the president of the Access Project. [no mic]

First, second and third, I want to thank Lori Reseem [phonetic] and the folks from Red Hat for making this possible. It is fitting that the open source community would see the benefit of this dialogue and the relationship between the open source principals and the [??] principals that we're talking about. And the [no mic] regulation and non-regulation and government and no government. Some of the stuff that we're going to be talking about here.

And it's a magnificent contribution. We're extremely grateful. I'm also very grateful to Larry for many things, no the least of which is his conceptual contribution to what's brought us here today.

And I also want to thank Wendy [phonetic] back there who has made this all work wonderfully efficiently. This kind of thing can be very, very difficult and she has done it with remarkable panache as well.

This is a cross-cultural conference. My contribution to that is to try to get into it as — this is, with the possible exception of an aerobics class, the first time that I've appeared in a group of more than ten people in a room without a tie.

My bias — and I don't know how this bias would be shared

through the course of the day — is that there isn't us here. There's an us that wants to see the design, the architecture of the evolving technology implemented in a manner that fulfills its technological potential.

My concern is that in ways that may not have been the case before or may have not been as important before — it is not going to happen unless the technology community understands that they're going to have to eventualize with policy makers — this is not going to happen by itself.

When I say that there's an us and a them here, this is one of the few conferences I've ever attended where I'm in the us and Peter Huber [phonetic] is in the thems. The policy makers — many of whom are in this room — understand a great deal more about technology than a lot of people around here may realize. And they want to understand a great deal more.

But — this is, of course, Larry's thesis — articulated it much better than I can, but the role of the implementation necessarily involves an interplay with international and national law regulation and policy. It is no longer possible — if it ever was — to sit back, write good code, design networks and just assume that whatever is the capacity of those networks is going to be filled out and expanded — regardless of what those policy makers do.

The bottle neck here from government and policy is as significant as any kind of technological impediment could possibly be. And unless the technology community can go and explain how the potential can be de-limited if steps are not taken to keep it open, we won't get what I hope we can get.

From my perspective, as a constitutional matter, this fits with my vision of the First Amendment — which is that if government can create an opportunity for discourse and debate, if it can enable the comments that Larry has written about to develop and function, then we can have true and effective democratic self-governments.

If the bottle necks are created, if we re-create, for example, the cable television model which favors leveraging value added content over distribution of the maximum amount of content in different ways — and which favors one way rather than two-way multiple discourse, we will be much the worse for it.

And I think that the growth and innovation that has really put America in the forefront of the international economy — this week on Wall Street notwithstanding — is to continue. This is an important component. Now, as I said, this is all my bias and I

hope this part of the dialogue from my end that I hope will transpire.

Obviously, I need to hear what the technology community has to say and if I'm misunderstanding or getting it wrong, we need to know that. So that's what I hope to get out of it. I'm just thrilled that some of the names that I've been reading for years and people whose work I have been handing and slapping on tables and saying here, read this. Professor Saltzer [??] to these people. And I look forward to the day. Thank you.

LARRY: Thank you. Okay. So we're going to start with Jerry Saltzer give us a very brief baseline on the idea of e2e. Most of you here have received papers and read the paper, but there are members in the audience who didn't have that opportunity. So Jerry, if you'd just like to lay this out in as simple and straightforward way as possible.

JERRY: I'm Jerry Saltzer. I am a retired professor of computer science at MIT. And the background here is kind of interesting. Digital technology gives us an opportunity to build systems of unimaginable complexity. And also unmanageable complexity.

And you have to do something about that. and so the issue here is really to come up with techniques that can somehow conquer and deal with this complexity. There are several such techniques.

One of the key techniques is the old rule of divide and conquer which goes with the modern buzz word modularity [phonetic]. If you divide the thing up into pieces, then you can work on each of the individual pieces. And you can exchange the pieces and you can replace them.

But it turns out that this is sort of only the beginning. And the reason it's only the beginning is it's not at all obvious ever how to divide the lines into the modules. Getting the right modularity is the hard part. So you have to apply some additional things to what you're doing beyond just saying well, we have modularity.

One of the additional things that has proven to be very powerful is to arrange modules in what are called layers. And most modern systems are organized in layers. If you open up a television set, you'll find that there are layers or chips in it.

There are connected chips and then there's an entire operation. And your home system consists of a next higher layer of these modules put together, namely the VCR and the TV and the video hi-fi system that connects behind them.

So things are organized in layers which makes it a lot easier to see what you're doing. Finally, that by itself also is not sufficient

to be able to keep the complexity of your control. In addition to having modules and arranging the modules in layers, you really need to design systems in such a way that you can replace and change. You have to plan to iterate.

That is, you design things and then you expect you're going to have to replace them with newer and better versions and move the modular boundaries around. The principal here is fairly straightforward. You're almost certainly not going to design it right the first time. You should plan to have to replace the pieces. And plan to do it over again several times.

So in light of all of that which are the techniques that computer scientists tells their students that's how you go about dealing with large complex digital systems, you need some designed rules to help you decide where in these systems to place individual functions. And this is where the e2e class of argument comes into play.

It's a line of reasoning that simply says that in the lower layers in your system, you are going to be supporting function that you cannot predict at a higher level. And therefore, you should be very conservative about imbedding function in the lower layers because if you embed a function down there, everybody up there above you has to live with it and has to work with it.

And if you get it just a little bit wrong, then it's not going to work well for at least some applications up above. And therefore, you should — in the interest of being conservative, in the interest of allowing future iteration, you should push function up. you should keep the bottom layers as general and straightforward as possible.

So, for example, if you were designing a television set today, without the knowledge of fifty years of previous experience of people designing television sets and you are sort of clueless as to how you ought to divide the thing up, you would say gee, at the bottom layer where we choose the chips, we shouldn't be putting in specific functions and dedicating chips to various things. We should buy some memory. A memory chip is a good thing to put down there because it's very general.

And that leads to designers who hooks the chips together up above us or maybe to the user who programs it in some [??] elaborate television set. The opportunity to make the design decisions as late as possible in the sequence of producing the system.

The name e2e comes about from applying this line of argument to communications systems where the layers have not only a

building up, but also a communication across in which at the very bottom there is a layer which moves bytes across wires. And then in between, there is a layer that decides which wires to use for this particular communication.

And above that is a layer which figures out how to get a font [phonetic] loaded from there over to here. And these various layers have the same property. Namely, you would prefer to see the lower layers have the minimum amount of design decision wired into them. And allow the upper layers, the guy who was actually responsible for actually moving a file from here to there or responsible for getting a telephone communication, voice-over IP [phonetic] from here to there — figure out the best way to make that happen.

So that in essence is what the e2e argument — or the line of reasoning called the e2e argument is all about.

LARRY: Thank you. Okay. So we're going to start with Jerry Saltzer give us a very brief baseline on the idea of e2e. Most of you here have received papers and read the paper, but there are members in the audience who didn't have that opportunity. So Jerry, if you'd just like to lay this out in as simple and straightforward way as possible.

JERRY: I'm Jerry Saltzer. I am a retired professor of computer science at MIT. And the background here is kind of interesting. Digital technology gives us an opportunity to build systems of unimaginable complexity. And also unmanageable complexity.

And you have to do something about that. and so the issue here is really to come up with techniques that can somehow conquer and deal with this complexity. There are several such techniques.

One of the key techniques is the old rule of divide and conquer which goes with the modern buzz word modularity [phonetic]. If you divide the thing up into pieces, then you can work on each of the individual pieces. And you can exchange the pieces and you can replace them.

But it turns out that this is sort of only the beginning. And the reason it's only the beginning is it's not at all obvious ever how to divide the lines into the modules. Getting the right modularity is the hard part. So you have to apply some additional things to what you're doing beyond just saying well, we have modularity.

One of the additional things that has proven to be very powerful is to arrange modules in what are called layers. And most modern systems are organized in layers. If you open up a television set, you'll find that there are layers or chips in it.

There are connected chips and then there's an entire operation. And your home system consists of a next higher layer of these modules put together, namely the VCR and the TV and the video hi-fi system that connects behind them.

So things are organized in layers which makes it a lot easier to see what you're doing. Finally, that by itself also is not sufficient to be able to keep the complexity of your control. In addition to having modules and arranging the modules in layers, you really need to design systems in such a way that you can replace and change. You have to plan to iterate.

That is, you design things and then you expect you're going to have to replace them with newer and better versions and move the modular boundaries around. The principal here is fairly straightforward. You're almost certainly not going to design it right the first time. You should plan to have to replace the pieces. And plan to do it over again several times.

So in light of all of that which are the techniques that computer scientists tells their students that's how you go about dealing with large complex digital systems, you need some designed rules to help you decide where in these systems to place individual functions. And this is where the e2e class of argument comes into play.

It's a line of reasoning that simply says that in the lower layers in your system, you are going to be supporting function that you cannot predict at a higher level. And therefore, you should be very conservative about imbedding function in the lower layers because if you embed a function down there, everybody up there above you has to live with it and has to work with it.

And if you get it just a little bit wrong, then it's not going to work well for at least some applications up above. And therefore, you should — in the interest of being conservative, in the interest of allowing future iteration, you should push function up. you should keep the bottom layers as general and straightforward as possible.

So, for example, if you were designing a television set today, without the knowledge of fifty years of previous experience of people designing television sets and you are sort of clueless as to how you ought to divide the thing up, you would say gee, at the bottom layer where we choose the chips, we shouldn't be putting in specific functions and dedicating chips to various things. We should buy some memory. A memory chip is a good thing to put down there because it's very general.

And that leads to designers who hooks the chips together up

above us or maybe to the user who programs it in some [??] elaborate television set. The opportunity to make the design decisions as late as possible in the sequence of producing the system.

The name e2e comes about from applying this line of argument to communications systems where the layers have not only a building up, but also a communication across in which at the very bottom there is a layer which moves bytes across wires. And then in between, there is a layer that decides which wires to use for this particular communication.

And above that is a layer which figures out how to get a font [phonetic] loaded from there over to here. And these various layers have the same property. Namely, you would prefer to see the lower layers have the minimum amount of design decision wired into them. And allow the upper layers, the guy who was actually responsible for actually moving a file from here to there or responsible for getting a telephone communication, voice-over IP [phonetic] from here to there — figure out the best way to make that happen.

So that in essence is what the e2e argument — or the line of reasoning called the e2e argument is all about.